中国科学: 信息科学 2021 年 第 51 卷 第 8 期: 1233–1254
SCIENTIA SINICA Informationis
论文

方舱计算

蒋昌俊[1,2*], 丁志军[1,2], 喻剑[1,2], 章昭辉[1,2], 闫春钢[1,2], 张亚英[1,2], 王鹏伟[1,2]

1. 同济大学嵌入式系统与服务计算教育部重点实验室, 上海 201804
2. 同济大学网络信息服务上海市工程研究中心, 上海 201804
* 通信作者. E-mail: cjjiang@tongji.edu.cn

收稿日期: 2020–06–11; 修回日期: 2020–12–30; 接受日期: 2021–02–24; 网络出版日期: 2021–08–05

国家重点研发计划 (批准号:2018YFB2100800) 资助项目

摘要 随着信息技术创新日新月异, 数字化、网络化、智能化深入发展, 新应用层出不穷、新业态蓬勃发展, 对业务系统的敏捷构造和持续运维提出了更高的要求. 为此, 本文提出了一种新的计算模式: 方舱计算. 所谓方舱计算, 是通过网络访问的, 面向 IT 任务全生命周期的跨域资源配置和协同的计算集成环境. 其核心是 "方舱专用机动、资源跨域伸缩、系统运维自治". 本文给出了方舱计算系统结构及其工作原理, 方舱计算系统主要由方舱生成与管理系统、跨域资源管理系统、虚拟数据中心系统和若干网关 (方舱网关、虚拟数据中心网关、跨域资源网关等) 组成. 进而, 给出了适于方舱计算的资源分配最优化问题定义, 在兼顾数据资源、计算资源和存储资源等的基础上, 实现最小化资源成本. 通过规划求解器与近似优化算法仿真实验, 表明本文的方法能够兼顾数据资源和物理资源的分布, 实现方舱资源的优化配置.

关键词 方舱, 虚拟数据中心, 跨域资源管理, 资源分配, 资源分布图, 资源目录

1 引言

信息技术是当今世界经济和社会发展的重要驱动力, 推动了全球产业结构转型和优化升级, 带来了人类生产生活方式的深刻变化. 通信、网络等新技术的飞速发展, 极大地拓展了信息服务业的发展空间, 也带来了新的发展机遇. 新应用层出不穷、新业态蓬勃发展. 与此同时, 需求的多样性、环境的多变性对信息服务技术提出了更高的要求.

第一, 应用正在变得越来越复杂, 需要应对多变的需求、需要支持更多的用户、需要更强的计算能力、需要更加稳定安全等, 而为了支撑这些不断增长的需求, 应用提供商不得不投入更大的人力、物力和财力去支持各类硬件设备 (服务器、存储、带宽等) 和软件 (数据库、中间件等) 的运维等, 业务系统的敏捷构造与持续运维成为当前热点.

引用格式: 蒋昌俊, 丁志军, 喻剑, 等. 方舱计算. 中国科学: 信息科学, 2021, 51: 1233–1254, doi: 10.1360/SSI-2020-0173
Jiang C J, Ding Z J, Yu J, et al. Cabin computing (in Chinese). Sci Sin Inform, 2021, 51: 1233–1254, doi: 10.1360/SSI-2020-0173

第二, 智能处理、互联设备、数字服务和云应用的快速增长正在推动着数据的爆发式增长. 尽管大数据中几乎包含了所有我们需要的信息, 但是由于大数据在数量、类型、动态特征等方面已大大超出了人类的认知, 如何高效获取和处理这么多的动态数据成为一个公认的难题.

第三, 信息产品和资源的不断丰富, 使得资源的优化配置成为可能和必须, 在多种可以相互替代的资源使用方式中, 如何选择较优一种, 以达到业务处理的最高效率, 实现消费者、企业及社会利益的最大满足.

信息技术的不断发展, 也无一不是在解决上述重大问题和挑战. 特别是计算机网络和互联网的出现和普及, 出现了一种新的网络结构 —— 网格. 网格是把地理位置上分散的资源集成起来的一种基础设施. 资源共享是网格的根本特征, 消除资源孤岛是网格的最终目标 [1,2]. 网格的出现更好地利用了分散的计算资源, 并且提供了较高的可伸缩性. 为了保护共享信息和对特定资源的有效共享, 在网格的基础上提出了虚拟组织. 虚拟组织是由个人和自治域按照一定的资源共享的规则形成的集合, 自治域或个人通过虚拟组织共享资源和进行问题求解. 虚拟组织要求用户必须共享出自己的资源, 同一虚拟组织内的用户既可以提供资源, 也可以使用资源. 由于虚拟组织是在网格基础上构建, 所以, 虚拟组织内用户提交的执行作业需要网格中心进行调度 [3,4] 等.

虚拟组织实现了更强的信息共享, 但是在资源使用上缺乏通用性. 云计算恰好提供了一个统一的使用平台. 在系统平台方面, 云计算是物理的服务器或虚拟的服务器; 在应用方面, 它描述了一种可以通过互联网访问的可扩展的应用程序. 云计算的出现, 带来了以下好处: 降低了计算资源的使用成本、减少了软件使用成本、提供了按需的资源使用模式、高可用性和扩展性等 [5~7]. 虽然, 现有的云计算已经成为主流的计算模式, 但是也会存在以下缺点: 实时性不够、不利于数据安全和隐私等. 为了保护用户数据的安全性和实时响应用户需求, 边缘计算模式被提出. 边缘计算是在靠近物或数据源头的网络边缘侧, 融合网络、计算、存储、应用核心能力的计算模式, 就近提供边缘智能服务 [8]. 边缘计算追求的目标就是更快响应. 边缘计算相对于云计算具有较多的优点, 例如, 在网络边缘处理大量临时数据, 不再全部上传至云端, 减少了网络泄露的风险、在靠近数据生产者处做数据处理, 可以获得较快的响应. 然而, 边缘计算也存在明显的缺陷, 即计算资源的有限性, 无法承担大量的数据计算任务.

受边缘计算的启发, 采用多个不同的云中心处理用户请求, 既可以确保计算能力, 又可以获取较近的云资源. 多云是满足上述需求的一种模式. 多云是指组织在单个网络架构中使用来自 AWS, Azure, Google 等全球主要云计算提供商的多个云计算和存储服务. 多云通常具有多个身份识别基础设施, 适用于多个环境, 能够混合来自 SaaS, IaaS 和 PaaS 的交付机制. 多云还支持独立于用于访问这些服务的 API 和接口的多个云存储服务 [9~12]. 云际计算 (JointCloud computing) [12,13] 是以多个云服务实体之间开放协作为基础, 通过多方云资源深度融合, 方便开发者通过软件定义方式定制云服务, 创造云价值的新一代云计算模式. 其中重点开展了跨云多利益主体价值交换支撑技术的支撑, 为云际计算商业模式创新提供了有力支撑. 多云的使用增加了应用程序的灵活性, 提高了应用的可靠性, 同时, 避免了被供应商锁定. 但是, 多云环境下的业务运行复杂, 需要同时考虑多个供应商的价格、容错等, 增加了管理成本.

此外, 1988 年施乐公司 PARC 实验室首席科学家马克 · 威瑟 (Mark Weiser) 首次提出 “泛在计算” (ubiquitous computing) 的概念, 描述了任何人无论何时、何地都可通过合适的终端设备与网络进行连接, 获取个性化信息服务的全新信息社会. 因此, 间断连接与轻量计算 (即计算资源相对有限) 是泛在计算最重要的两个特征, 其更加强调设备与设备之间通信的主动性 [14,15].

针对现有计算模式和技术的问题和不足, 本文提出了一种新的计算模式: 方舱计算, 所谓方舱计算, 是通过网络访问的, 面向 IT 任务全生命周期的跨域资源配置和协同的计算集成环境. 与现有计算

模式和技术相比, 方舱计算的特点包括: (1) 应 IT 任务的需要而机动搭建、应 IT 任务的执行而伸缩管理、应 IT 任务的结束而动态消亡; (2) 从 IT 任务全生命周期的纵向维度, 完成 “识别需求、资源配置、任务执行、结束任务” 4 大功能; (3) 从 IT 任务所需资源的横向维度, 实现数据资源和物理资源的统筹配置和协调运行.

具体地, 本文的创新有以下 3 点:

(1) 提出了一种新的计算模式, 方舱计算, 其核心是 “方舱专用机动、资源跨域伸缩、系统运维自治”.

(2) 给出了方舱计算系统结构及其工作原理, 方舱计算系统主要由方舱生成与管理系统、跨域资源管理系统、虚拟数据中心系统和若干网关 (方舱网关、虚拟数据中心网关、跨域资源网关等) 组成. 特别地, 我们定义了两种不同的方舱计算模式, 集中式和自治式, 并对二者的差别进行了阐述和分析.

(3) 给出了适于方舱计算的资源分配最优化问题定义, 在兼顾数据资源、计算资源和存储资源等的基础上, 实现最小化资源成本.

全文内容组织如下: 第 2 节介绍了方舱计算系统体系结构, 定义了集中式和自治式方舱计算模式, 给出了方舱计算的分析比较和应用场景示例; 第 3 节介绍了方舱生成与管理系统, 在系统结构的基础上, 给出了方舱资源需求描述和分配模型; 第 4 节介绍了跨域资源管理系统, 给出系统结构、跨域资源目录模型及其相关算法; 第 5 节介绍了虚拟数据中心系统, 给出系统结构和数据资源分布图模型等; 第 6 节进行了方舱资源配置方案的实验与分析; 第 7 节最后结语部分总结全文, 并对未来工作给出展望.

2 方舱计算系统架构

方舱计算系统其核心就是根据用户的业务需求, 灵活机动地优化配置和组织互联网上的数据资源和物理资源, 搭建专用定制的业务集成环境, 即方舱. 显然, 要实现高效的数据资源和物理资源的优化配置, 就需要能够获取和把握互联网上的相关资源. 在方舱计算系统中, 这部分功能是由跨域资源管理系统和虚拟数据中心系统实现完成的.

跨域资源管理系统是一个独立的 7×24 小时运行的公共服务系统, 提供资源注册服务. 互联网上的 IT 资源可以通过注册系统进行注册. 注册服务需要资源提供方提供关于资源的地理位置、资源的描述、资源获取方式接口、资源使用资费、实时资源状态查询接口等信息. 资源通过注册以后, 就会形成一个资源目录. 这一目录通过资源状态更新模块定期进行更新. 跨域资源管理系统同时还提供了方舱资源请求服务, 用于接收方舱的资源请求. 当接收到一个资源请求清单后, 方舱资源申请模块就会根据资源清单中的要求, 使用对应资源申请接口向资源提供方申请资源, 当资源申请成功以后, 则会更新跨域资源目录, 同时将申请成功的信息返回给资源申请人.

虚拟数据中心系统也是一个独立的 7×24 小时运行的公共服务系统. 其数据来源主要有两种途径, 一个是开放的互联网数据, 另一个是协议数据提供方. 网络数据勘探器使用勘探算法对两种数据资源进行勘探, 生成数据资源分布图. 数据资源分布图主要描述了数据资源的地理位置、数据资源类型、数据资源更新速度、数据资源量等特征. 外部应用可以通过虚拟数据中心的公共服务获取到所需要数据资源信息.

2.1 方舱计算系统架构

具体实施时, 方舱计算既可以采用集中模式, 其系统结构如图 1 所示, 也可以采用自治模式, 系统

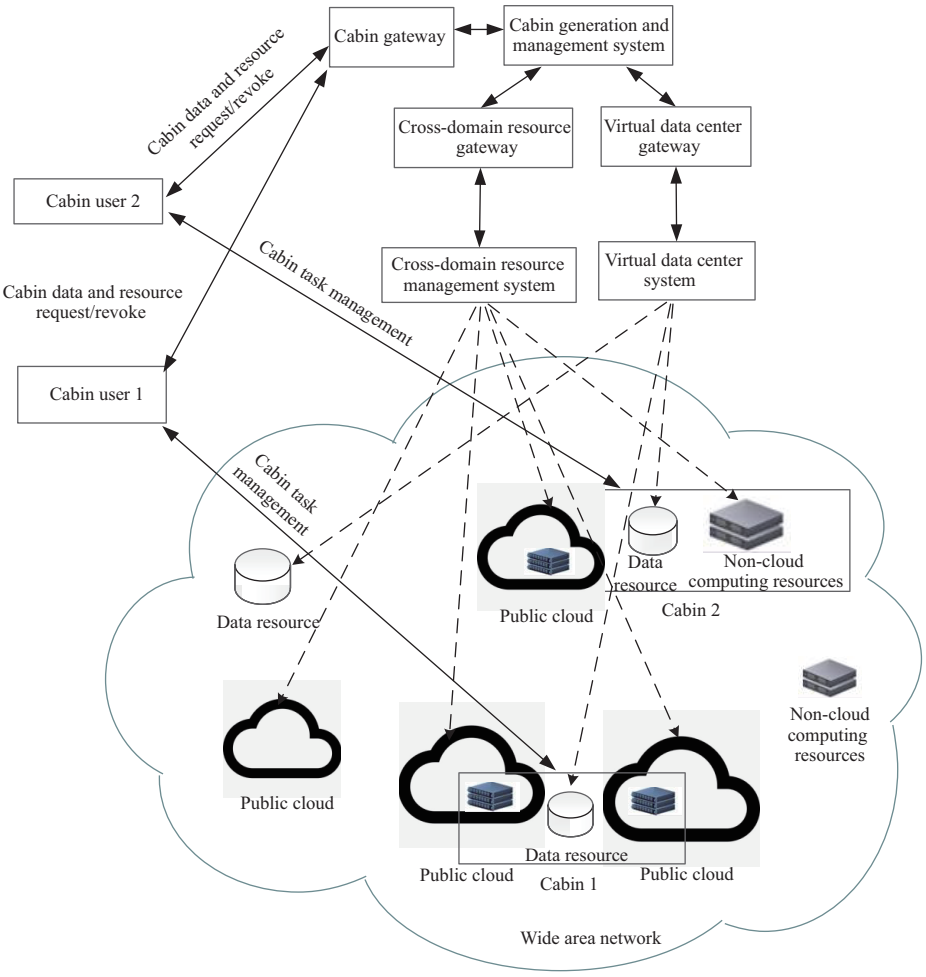


图 1　(网络版彩图) 集中式方舱计算结构

Figure 1　(Color online) The architecture of centralized cabin computing

结构如图 2 所示.

如图 1 所示的集中模式, 当用户有一个业务任务时, 用户通过方舱网关登录方舱生成与管理系统, 通过身份认证后提交任务的需求. 需求分为两类, 一类是总的方舱任务需求, 主要包括数据传输和计算处理的成本、时间约束等. 另一类是计算数据要求 (数据类型、数据内容的关键字、数据量要求等) 信息.

方舱生成与管理系统通过跨域资源网关从跨域资源管理系统获取到跨域资源目录信息, 同时通过虚拟数据中心网关从虚拟数据中心系统获取到所需的数据资源分布图信息. 方舱生成与管理系统根据用户的需求信息, 结合跨域资源目录与数据资源分布图, 进行资源分配, 并生成方舱的资源需求清单.

方舱生成与管理系统将生成的资源需求清单提交给跨域资源管理系统. 跨域资源管理系统向各资源域中心发出资源请求, 由各资源域中心实施具体物理资源配置. 等所有的资源配置都完成后, 跨域资源管理系统向方舱生成与管理系统返回方舱资源配置信息, 完成方舱资源分配与配置. 进而用户根据业务需要, 在配置的资源上进行方舱具体业务计算环境的搭建, 并自主进行业务运行.

方舱的计算任务完成后, 方舱用户可通过方舱生成与管理系统关闭所生成的方舱. 一旦用户决定

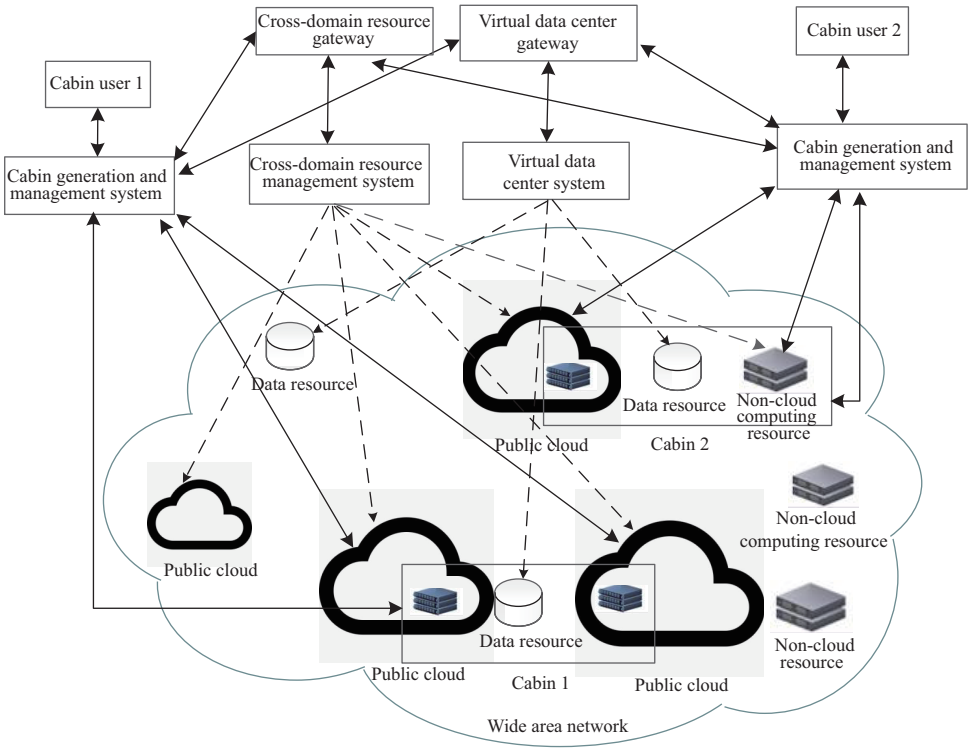


图 2 (网络版彩图) 自治式方舱计算系统结构

Figure 2 (Color online) The architecture of autonomous cabin computing

撤消方舱, 方舱生成与管理系统会通过跨域资源管理系统将所申请的资源全部返还给各资源提供方, 完成方舱资源的释放.

图 2 所示的自治模式是当用户有一个业务任务时, 可以在本地安装一个方舱生成与管理系统. 在方舱生成与管理系统中提交方舱需求后, 方舱生成与管理系统也会访问跨域资源管理系统与虚拟数据中心系统, 获取跨域资源目录与数据分布图, 进而执行分配算法生成方舱资源分配清单. 之后, 用户可以选择不通过跨域资源管理系统直接向互联网上的资源域中心申请资源, 获得资源后的管理方式与集中模式一样.

相比于集中模式, 自治模式的区别主要在于: (1) 方舱生成与管理系统的部署方式不同. 集中模式中, 存在一个集中式的后台方舱生成与管理系统, 所有用户访问该系统, 提交方舱需求; 自治模式中, 每个用户都安装部署独立的方舱生成与管理系统, 由其自主提交方舱需求; (2) 资源分配存在差别. 在集中模式中, 由于有集中的后台方舱生成与管理系统, 所以可以对所有用户的方舱需求实施统一分配, 可以实现全局资源优化配置; 而自治模式中, 由于每个用户的方舱生成与管理系统自主分配资源, 所以无法实现全局资源优化配置, 并且可能存在多个用户间资源分配的冲突, 因此需要进行冲突消解, 可能会带来额外的系统代价; (3) 资源配置的方式不同. 集中模式中是由跨域资源管理系统作为代理, 向各资源域中心申请物理资源; 而在自治模式中, 用户不再通过跨域资源管理中心, 而是自主向各资源域中心申请物理资源.

由图 1 和 2 可以看到, 方舱计算系统中不仅包括方舱生成与管理系统、虚拟数据中心系统和跨域资源管理系统, 而且还包括了若干网关 (方舱网关、虚拟数据中心网关、跨域资源网关等).

方舱网关, 主要由业务网关、身份认证模块、API 监控、智能路由和限流模块、用户注册服务等模块组成. 其中业务网关为业务请求的统一接入点, 可实现服务级流控、服务过滤、服务聚合与发现等功能; 身份认证模块实现用户的注册、管理及身份认证; API 监控实现对外部 API 服务的监控; 限流模块主要实现对业务访问流量的限制; 智能路由实现服务访问的路由.

跨域资源网关, 除了具有限流、身份认证 (可选)、API 监控、智能路由等模块外, 还具有用户注册、跨域资源目录服务、任务需求提交服务等内部服务编排能力.

虚拟数据中心网关, 除了具有限流、身份认证 (可选)、API 监控、智能路由等模块外, 还具有用户注册、数据资源分布图服务等内部服务编排能力, 可以实现业务服务的快速组合.

2.2 方舱计算的分析比较

如上所述, 方舱计算从资源使用的原理来说, 可以视为网格计算和云计算的综合体, 下面给出其定义:

定义1 设网格计算 G 是从分散资源 R 到虚拟组织 O 的聚合函数, 即 G: $R \to O$; 云计算 C 是从物理资源 P 到虚拟资源 V 的虚拟化函数, 即 $C : P \to V$; 则方舱计算 M 就是网格计算 G 和云计算 C 的复合函数, 即 $M = G \cdot C$. 方舱计算 M 既可以是网格计算或者云计算, 又可以是二者的复合, 表现为虚拟资源的聚合化.

具体来说, 跨域资源管理系统通过实虚映射 (聚合和划分的混合) 技术, 将底层资源域中心的资源映射为虚拟资源, 形成跨域资源目录, 然后方舱生成与管理系统根据 IT 任务需求, 结合数据资源分配方案, 基于跨域资源目录, 输出方舱资源分配方案和实现方舱资源聚合.

方舱计算的技术特点可以从与网格计算、云计算的对比分析中更清楚的总结和归纳, 表 1 是对方舱计算和网格计算、云计算进行的对比分析.

其中, 网格计算和虚拟组织侧重于计算密集型的科学计算, 针对特定领域和问题形成虚拟组织, 强调资源聚合, 通过开放网格服务体系结构/Web 服务资源框架 (open grid services architecture(OGSA)/Web service resource framework(WSRF)), 将跨域分散的 “小” 资源聚合成为 “大” 的资源, 以解决原先 “小” 资源无法解决的 “大” 计算问题; 云计算则侧重于数据密集型的企业计算, 强调资源的划分和虚拟化, 通过将 “大” 的物理资源虚拟化为 “小” 的虚机资源, 基于分布式资源管理和 MapReduce 等计算框架, 在实现资源共享的同时, 降低了资源的使用成本. 可以看到, 网格计算、云计算较少考虑数据资源和物理资源的统筹配置, 在面对数据和计算混合密集型任务上有所不足. 而方舱计算, 综合网格计算、云计算等优点, 通过实虚映射 (聚合和划分的混合), 形成以方舱为单位的虚舱, 采用混合部署和异构计算框架, 实现数据资源和物理资源的统筹配置和协调运行. 此外, 泛在计算与方舱计算都强调了面向 IT 任务的服务便捷性, 但是泛在计算更加强调设备与设备之间通信的主动性, 而方舱计算则侧重数据与计算的统筹配置和协调运行.

2.3 方舱计算的应用场景示例

如上所述, 方舱计算适用于包含数据密集和计算密集的混合计算 IT 任务, 如城市智能交通信息服务就是此类系统的典型代表. 下面就以城市智能交通信息服务为例介绍方舱计算的使用场景:

如图 3 所示, 交通信息服务中典型的服务或计算任务有数据采集 (包括流动 GPS 数据采集 f_1、车载视频数据采集 f_2、道路视频数据采集 f_3、路口流量数据采集 f_4 等)、实时流量处理 (包括基于流动 GPS 的路段实时交通流计算 f_5、基于车载视频的路段实时交通流计算 f_6、基于道路视频的路段实时交通流计算 f_7、基于路口流量的路段实时交通流计算 f_8); 全区域实时交通流量融合计算 g_1; 流

表 1 方舱计算与网格计算、云计算的分析比较

Table 1 Analytical comparison of cabin computing, grid computing and cloud computing

Computing mode	Feature					
	Resource organization			Applicable type	Application scenario	Core technologies	Business mode
Grid computing	Cross-domain	$\xrightarrow[\text{Heterogeneous}]{\text{aggregate}}$	Virtual organization	Computing-intensive	Scientific computing	Middleware OGSA/WSRF HPC computing framework (shared memory and message passing) Inter-domain flexibility	Null
Cloud computing	Same domain	$\xrightarrow[\text{partition}]{\text{Isomorphic}}$	Virtual machine	Data-intensive	Enterprise computing	Virtualization distributed resource management MapReduce framework Intra domain flexibility	Charge in terms of quantity
Cabin computing	Cross (same) domain	$\xrightarrow[\text{Mapping}]{\text{Hybrid construction}}$	Virtual cabin	Hybrid-intensive	Hybrid computing	Real and virtual mapping(combination of aggregation and partition) Hybrid deployment Heterogeneous computing framework Interdomain and intradomain flexibility	Charge in terms of quantity

量图谱生成 (包括推演区域划分 f_9、区域流量推演 g_2、跨区域流量图谱融合推演 g_3); 滤波带控制方案 f_{10}、滤波带控制需求 f_{11}、滤波带配时计算 g_4; 最短时间路径点播服务 f_{13}、最短时间路径跨域寻优 g_5、路段路况预测等计算 f_{12}. 其中 f_i 是数据密集型任务, 即涉及单域内的计算资源就可完成计算, g_j 是计算密集型任务, 即一次计算需要跨域计算资源聚合才能完成的计算.

例如, 在带有滤波带控制的基于流量图谱的最短时间的路径服务中, 完成一次最短时间路径点播服务需涉及以上计算任务共同协作完成, 即

$$((f_1 \cdot f_5) \parallel (f_2 \cdot f_6) \parallel (f_3 \cdot f_7) \parallel (f_4 \cdot f_8)) \cdot g_1 \cdot (f_9 \cdot g_2 \cdot g_3) \cdot ((f_{10} \cdot g_4 \cdot f_{11} \cdot g_2 \cdot g_3)^*) \cdot (f_{12} \cdot g_5 \cdot f_{13}),$$

其中, $\cdot$ 表示顺序计算关系, $\parallel$ 表示并行计算关系, $()^*$ 表示循环计算关系.

方舱在为以上服务序列提供计算环境时, 既需要考虑数据密集型的资源需求, 也需要考虑计算密集型的资源需求. 显然, 对于数据密集型任务 f_i 尽量提供域内的计算资源, 如云内的虚拟计算资源 C; 对于计算密集型任务 g_j, 可能域内资源不满足计算需求, 需要方舱提供跨域的计算资源, 如将云内资源和云外资源虚拟化后的聚合计算资源 G, 从而满足大规模跨域计算任务的需求.

3 方舱生成与管理系统

方舱生成与管理系统负责方舱的生成、组织与管理. 具体而言, 在方舱生成阶段, 负责接受用户发出的方舱搭建请求, 并根据请求分别调用虚拟数据中心系统和跨域资源管理系统中的数据分布服务

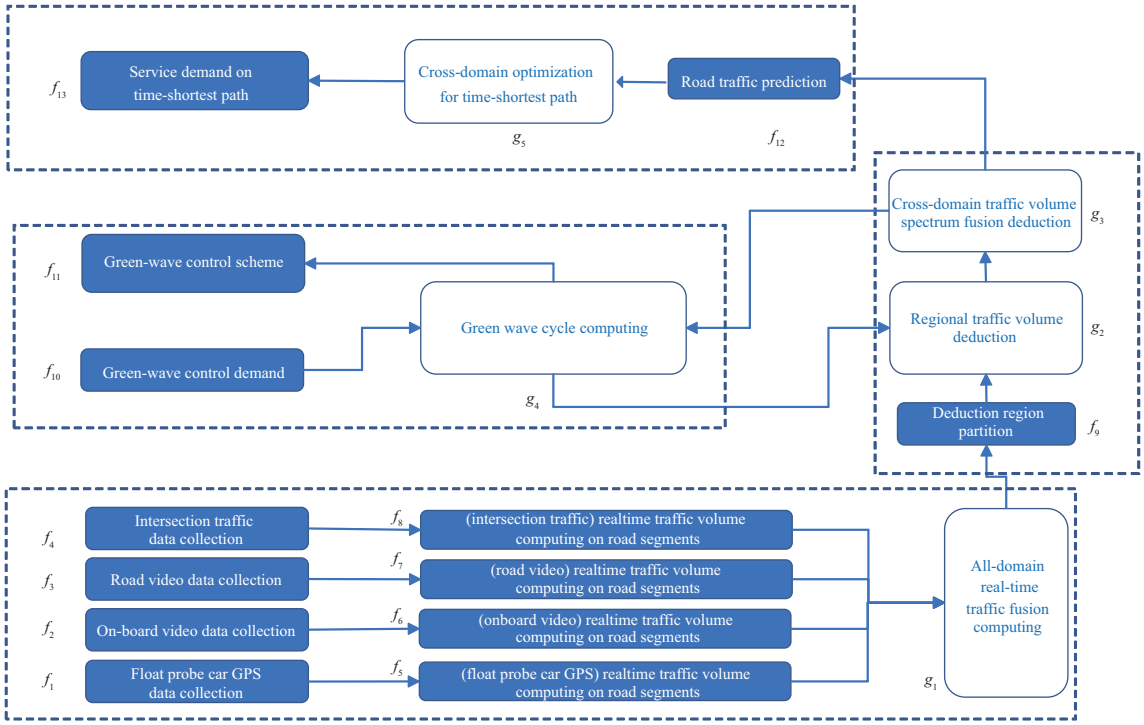


图 3　(网络版彩图) 城市智能交通信息服务

Figure 3　(Color online) Urban intelligent traffic information service

图和跨域资源目录, 生成数据资源和物理资源 (计算、存储和网络等) 的分配方案. 在搭建和运行阶段, 根据分配方案, 通过各资源域中心, 实现物理资源的实际配置, 最终得到方舱的实际数据资源和物理资源的配置方案. 进而用户根据业务需要, 在配置的资源上进行方舱具体业务计算环境的搭建, 并自主进行业务运行. 业务运行完成, 用户向方舱生成与管理系统提出方舱撤销请求, 系统相应执行方舱资源的释放操作, 完成整个方舱的全生命周期管理.

3.1　方舱生成与管理系统结构

如图 4 和 5 所示, 方舱生成与管理系统由用户需求管理模块、资源分配模块、方舱搭建与运行模型、方舱撤销等模块组成.

用户需求管理模块提供用户方舱资源需求填写和提交功能, 接受用户按照方舱资源需求规范填写和提交的需求信息, 并将其转发给资源分配模块.

资源分配模块接收到用户需求, 首先将其分解为数据资源请求和物理资源请求, 根据数据资源请求访问虚拟数据中心, 从数据分布图服务获取到数据的分布情况, 并返回数据资源分配方案. 然后分配模块再根据返回的数据资源分配方案, 综合用户的物理资源请求, 访问跨域资源管理系统, 获取跨域资源目录, 利用资源调度算法计算输出方舱资源分配方案.

方舱搭建与运行模块根据分配方案, 实现物理资源的实际配置, 最终得到方舱的实际数据资源和物理资源的配置方案. 进而用户根据业务需要, 在配置的资源上进行方舱具体业务计算环境的搭建, 并自主进行业务运行.

这其中, 集中式和自治式方舱在物理资源的配置操作上存在不同, 如图 4 所示, 集中式方舱模式

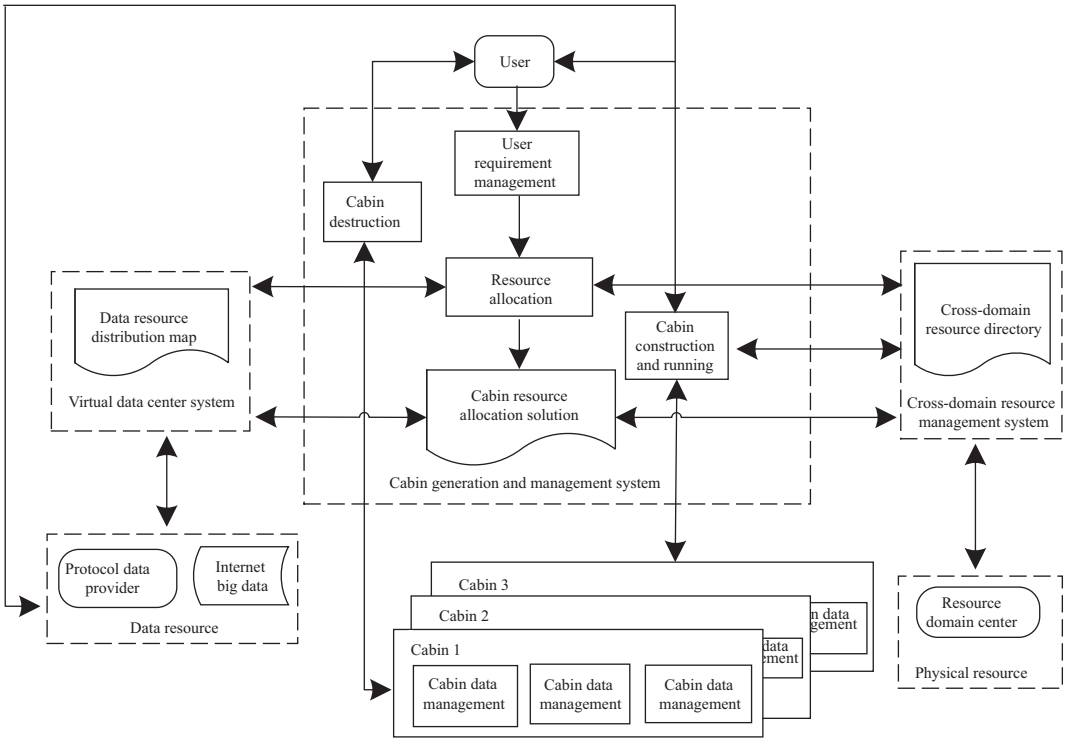


图 4 集中式方舱生成与管理系统

Figure 4 Centralized cabin generation and management system

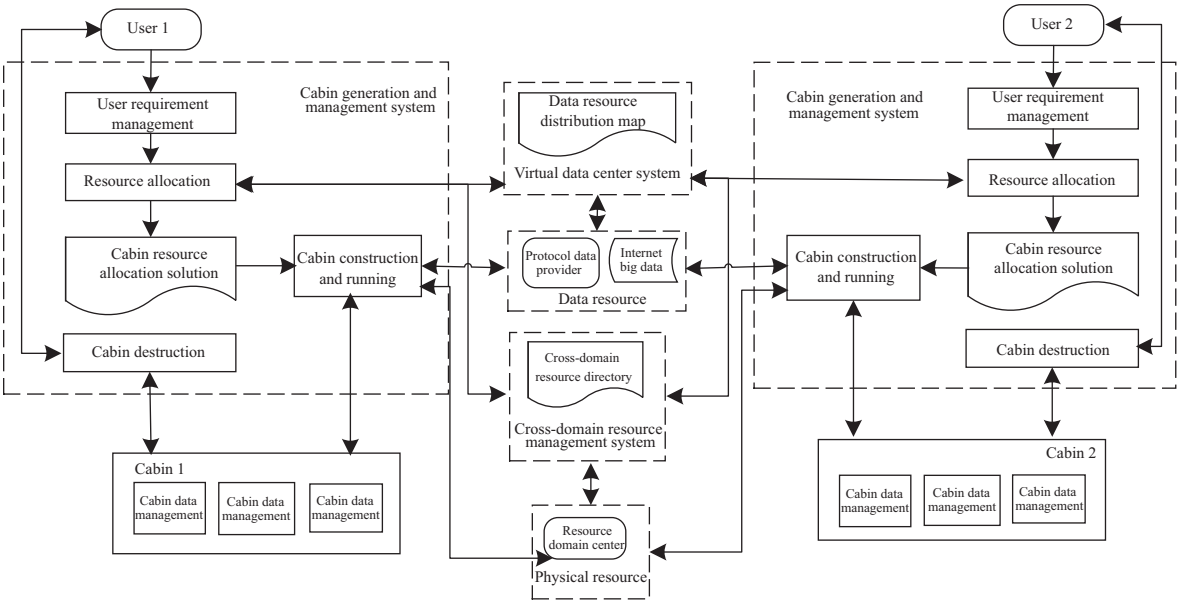


图 5 自治式方舱生成与管理系统

Figure 5 Autonomous cabin generation and management system

表 2　方舱需求相关符号和含义

Table 2　Symbols and their meaning related to the cabin requirement

Symbol	Meaning
D	User requirements for the total time of data processing
T	Data types users processed
N	Number of keywords required for data resources
K_i	i-th keyword of data resource requirement
S	Data volume of the data set, unit: GB

中方舱搭建与运行模块将分配方案发送给跨域资源管理系统, 由跨域资源管理系统统一向各资源域中心发出资源配置请求, 在得到各资源域中心返回的具体物理资源配置方案后, 汇总后返回方舱搭建与运行模块; 而在图 5 所示的自治式方舱模式中方舱搭建与运行模块将自行根据分配方案向各资源域中心发起资源配置请求, 并汇总配置方案. 需要注意的是, 自治式方舱模式由于各用户自主使用方舱生成与管理系统生成资源分配方案, 缺少集中式的全局资源分配管控能力, 所以可能各用户的资源方配方案存在资源冲突, 导致某些用户无法申请到所需的物理资源. 这种情况下, 系统将再次运行资源分配模块, 根据跨域资源目录最新状态, 重新执行分配算法进行资源分配, 直到成功为止. 方舱撤销模块根据用户的方舱撤销请求, 终止方舱运行, 执行方舱资源的释放操作.

3.2　方舱资源需求描述和分配模型

3.2.1　方舱资源需求描述模型

用户向方舱生成与管理系统提交方舱资源需求, 该需求描述如下: (1) 方舱任务相关的需求: 任务完成截止时间约束 D; (2) 方舱任务相关的数据资源需求: 数据类型、数据内容的关键字、数据量要求等. 方舱需求相关符号和含义如表 2 所示.

3.2.2　物理资源分配模型

本小节将构建方舱物理资源的分配模型, 其中需要用到的符号及其含义如表 3 所示. 方舱物理资源最优化分配问题为在已知各资源域中 VM 的数量、处理能力、存储能力、网络带宽等, 所需要使用的数据资源块的数量和大小, 以及各数据块到各资源域的物理距离及其网络延迟的情况下, 满足用户任务处理的截止时间约束下, 求解物理资源的分配方案, 使得任务处理的总费用最小.

资源分配问题的约束条件为:

(1) 数据处理总时间小于截止时间:

$$\sum_{j=1}^{N}\left[\left(\frac{S_j}{N_i}+L_{F_{i,j}}\right)+\frac{S_j}{n_{i,j}\times V_i}\right]\leqslant D.$$

(2) 分配的 VM 的个数为整数:

$$n_{i,j}\in\mathbb{N}^{+}.$$

(3) 分配的 VM 的个数小于或等于第 i 个资源域中心的 VM 的数量:

$$\sum_{j=1}^{N}n_{i,j}\times d_{i,j}\leqslant R_i.$$

表 3 资源分配模型相关符号和含义

Table 3 Symbols and their meaning for the resource allocation model

Symbol	Meaning
N	Number of data blocks users need to process
S_j	Size of user's j-th data block, unit: GB
D	User requirements for the total time of data processing
V_i	VM's processing speed of i-th resource domain center, unit: GB/s
P_i	Rental unit price of VM in the i-th resource domain center, unit: yuan/s
U_i	Unit price of uplink bandwidth corresponding to the VM in the i-th resource domain center, unit, yuan/GB
R_i	Number of VMS in the i-th resource domain center
N_i	Corresponding bandwidth of the VM of the i-th resource domain center, unit: GB/s
M_i	Storage capacity of the VM of the i-th resource domain center, unit: GB
$F_{i,j}$	Distance between the physical location of the i-th resource domain center and the physical location of the j-th data block, unit: km
$L_{Fi,j}$	Network propagation delay with distance length of $F_{i,j}$, unit: s
$n_{i,j}$	Number of VMS allocated to the j-th block by the i-th resource domain center
$d_{i,j}$	Whether the i-th resource domain center allocate VM to the j-th data block, with value 0 or 1

(4) 所分配 VM 的存储容量需大于或等于对应数据块的大小:

$$n_{i,j} \times M_i \times d_{i,j} \geqslant S_j.$$

方舱物理资源的分配需要满足如上约束, 其中需要说明的是约束 (1) 中, 数据总处理时间由数据传输时间和方舱计算时间总计得到, 数据传输时间根据文献 [16,17], 由网络传输加网络延迟 $L_{Fi,j}$ 组成. 在满足上述约束条件基础上, 其目标函数定义为

$$\min\ C = C_T + C_H.$$

其中, (1) 数据传输总成本:

$$C_T = \sum_{j=1}^{N} S_j \times U_i \times d_{i,j}.$$

(2) 数据处理总成本:

$$C_H = \sum_{j=1}^{N} \frac{S_j}{n_{i,j} \times V_i} \times P_i \times d_{i,j},$$

即方舱物理资源分配的目标函数为资源使用总成本最小:

$$\min\ C = \sum_{j=1}^{N} \left(U_i + \frac{P_i}{n_{i,j} \times V_i} \right) \times S_i \times d_{i,j}.$$

4 跨域资源管理系统

跨域资源管理系统 [18,19]1) 主要负责建立、管理和动态维护跨域资源目录.

4.1 跨域资源管理系统结构

跨域资源管理系统结构, 如图 6 所示, 主要由跨域资源目录、资源注册服务模块、资源状态更新模块、资源目录管理器模块、跨域资源目录服务模块、方舱资源请求服务模块、方舱资源申请模块等构成. 具体包括: 跨域资源目录, 跨域资源目录的主要功能是向方舱生成与管理系统提供可用资源域中心的具体情况. 它是跨域资源管理系统的核心数据结构组件, 反映了互联网资源的整体统计情况, 包括资源位置、资源种类、资源量、资源特征等信息. 方舱生成与管理系统, 通过读取用户请求和跨域资源目录构建以成本最小为目标的最优资源分配方案; 资源注册服务模块, 将资源域中心的资源信息注册至资源目录中. 其主要是通过资源注册算法实现. 通过动态注册、发现, 最终维护跨域资源的逻辑视图 (目录); 资源状态更新模块, 基于统一资源访问接口, 提供跨域资源目录的维护、更新、删减等管理操作, 同步跨域资源目录与实际资源域中心的资源信息; 资源目录管理器管理跨域资源目录, 接受注册服务和状态更新模块提交的物理资源信息, 实现目录数据的增删改查; 跨域资源目录服务提供对外的跨域资源目录服务; 方舱资源请求服务接受来自方舱生成与管理系统的方舱搭建请求; 方舱资源申请模块接受方舱生成与管理系统的资源分配方案请求, 并将请求进行拆分, 相应发给各资源域中心. 接收各资源域返回的资源分配信息, 汇总上报给方舱生成与管理系统.

4.2 跨域资源目录模型

跨域资源管理系统的核心在于对跨域资源目录的管理和更新. 跨域资源目录结构主要用于资源子系统的数据管理. 跨域资源目录结构可以采用关系型或非关系型数据库存储, 其逻辑结构为树形结构.

跨域资源的结构如图 7 所示, 主要包括第 0 层节点 (根节点)、第 1 层节点 (资源供应商)、第 2 层节点 (地理位置)、第 3 层节点 (资源域中心). 具体结构及其构造描述如下:

第 0 层节点 (根节点) 主要包括: 供应商数量、访问限制、子类别 1 指针、子类别 2 指针、……、扩展项等描述. 其中, 供应商数量用于记录不用的资源供应商厂商的数量; 类别指针用于指向不同的资源供应商的具体内容.

第 1 层节点主要包括: 厂商名称、访问限制、位置指针 1、位置指针 2、……、扩展项等描述. 厂商名称是指具体的资源供应商; 访问限制表明当前资源是否可用; 位置指针分别指向分布于不同地区的资源中心.

第 2 层节点主要包括: 位置名称、网络带宽、资源域中心数量、资源域指针 1、资源域指针 2 等. 位置名称指的是资源所分布的地区; 网络带宽是指该地区的上行网络带宽; 资源域中心数量表明该地区资源域的数量; 位置指针分别指向不同的资源域中心.

第 3 层节点为资源域中心节点, 主要包括: 资源配置、资源价格、访问形式、资源处理速度、扩展项等. 资源配置是指本资源域中心的资源的具体配置, 比如 CPU、MEM 和硬盘容量等; 资源价格表明了本资源域中心的单个资源的租用成本; 访问形式通常是指用户访问资源域中心资源的具体方式.

4.3 跨域资源管理系统的算法

跨域资源管理系统主要进行跨域资源目录的建立、管理、动态维护和对外访问, 实现跨域资源的

1) 蒋昌俊, 陈闵中, 闫春钢, 等. 基于虚拟超市的大规模网络资源组织与管理的关键技术及应用. 国家技术发明奖二等奖, 2010.

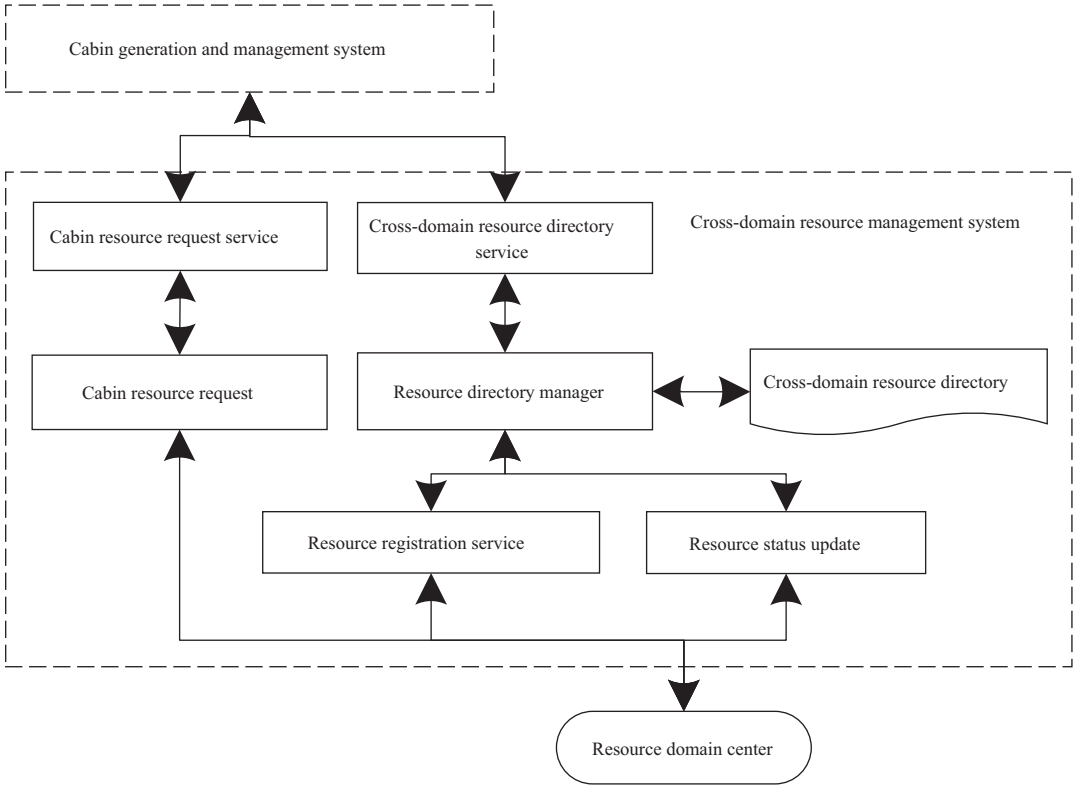


图 6 跨域资源管理系统结构

Figure 6 Cross-domain resource management system architecture

动态注册、状态更新和对外交互等. 此外, 针对集中式方舱模式, 还负责接受方舱生成与管理系统的资源分配方案请求, 实现跨域资源的配置. 涉及的算法如下所示.

4.3.1 资源注册算法

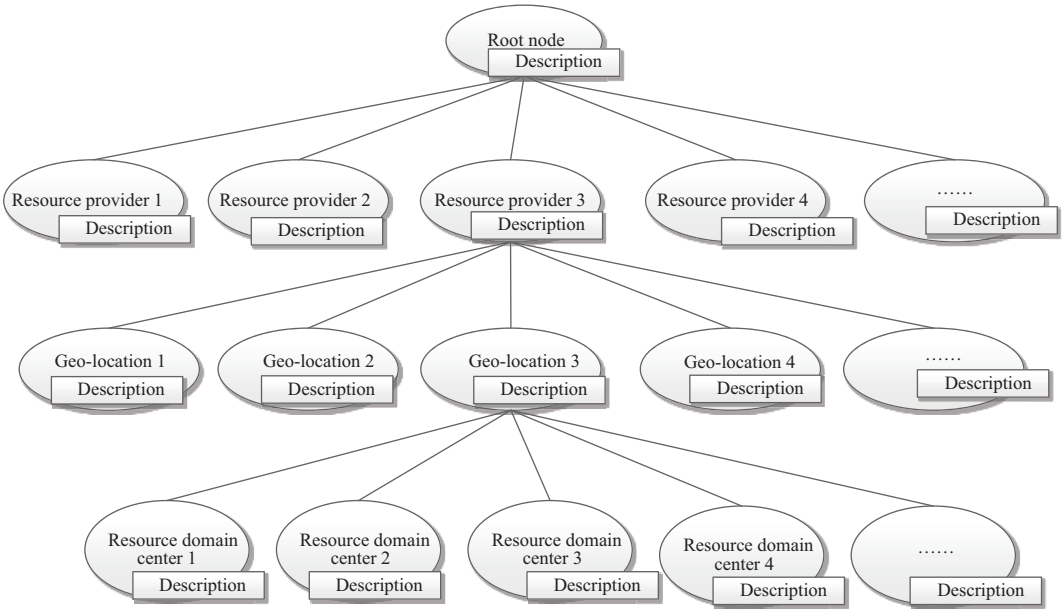
资源注册算法如算法 1 所示, 包括以下步骤: 首先访问资源域中心目录, 读取资源域中心下的资源状态文件 status.txt, 并提取其中的 CPU、MEM、硬盘容量、价格和地理位置等信息, 生成相应资源配置的资源条目, 查询跨域资源目录, 若该条目不在当前跨域资源目录中, 则将其插入到跨域资源目录的相应位置, 完成资源注册.

Algorithm 1 Resource registration algorithm

Input: Resource directory in resource domain center.

1: Read resource status file in the resource domain center;
2: Extract the CPU, MEM, hard disk storage, price and geographic location information of resources in the resource status file;
3: Generate resource entry of corresponding resource configuration;
4: **if** the resource entry is not in the cross-domain resource directory **then**
5: the entry is added to the cross-domain resource directory;
6: **end if**

Output: Cross-domain resource directory.



Node on level 0 (root node)	The number of providers	Access control	Subtype 1	Subtype 2	Extension
Nodes on level 1	Provider name	Access control	Location pointer 1	Location pointer 2	Extension
Nodes on level 2	Location name	Network bandwidth	The numbers of resource domain center	Resource domain pointer 1	Resource domain pointer 2
Nodes on level 3	Resouce configuration	Resource price	Way of access	Resource processing speed	Extension

图 7　跨域资源目录结构

Figure 7　Cross-domain resource directory structure

4.3.2　资源状态更新算法

资源状态更新算法如算法 2 所示, 首先访问跨域资源目录和资源域中心状态文件, 依次读取跨域资源目录中的目录, 并与资源域中心的状态文件进行对比. 如果跨域资源目录中的内容与状态文件中的信息一致, 则继续匹配下一条目录内容; 否则按照状态文件中的内容修订跨域资源目录中的内容, 实现资源状态的更新.

4.3.3　资源配置算法

资源配置算法如算法 3 所示, 根据方舱生成与管理系统的资源分配结果列表, 按照资源域进行拆分, 将拆分后的资源要求分组发送至不同的资源域中心. 资源域中心按照分组中的要求返回具体的资源配置信息, 比如: CPU、IP 和价格等. 当所有的资源域中心返回了具体配置信息后, 再将整合后的资源配置返回至方舱生成与管理系统.

5　虚拟数据中心系统

虚拟数据中心系统主要负责互联网大数据的注册、勘探、组织与对外访问. 其数据来源主要有两

Algorithm 2 Resource status update algorithm

Input: Cross-domain resource directory and resource domain center status file.

Read cross-domain resource directory list;
Read the state file in resource domain center;
for all resource configuration type in resource domain center status file **do**
    Cross-domain resource directory matching;
    if it is consistent **then**
        Go further to the next entry information;
    else
        The directory information in the cross-domain resource directory will be updated according to the content of the resource domain center status file;
    end if
end for

Output: Updated cross-domain resource directory.

Algorithm 3 Resource allocation algorithm

Input: Resource allocation results of cabin generation and management system.

Read the resource allocation result list of cabin generation and management system;
Group the results in the resource allocation result list by different resource domain centers;
Send different groups to different resource domain centers;
The resource domain center returns the specific resource information according to the requirements in the grouping;
Integrate all the results returned by the resource domain center, and return the integration results to the cabin generation and management system;

Output: Resource configuration information.

种途径, 一个是开放的互联网数据, 另一个是协议数据提供方. 本节重点介绍虚拟数据中心系统的体系结构和数据资源分布图模型, 有关数据勘探、数据组织和访问等具体算法可以参见文献 [20, 21], 本文不再赘述.

5.1 虚拟数据中心系统结构

虚拟数据中心的体系结构如图 8 所示, 主要由网络数据勘探器、互联网虚拟资源库、数据资源分布图管理、数据资源获取制导服务、数据协议生成与管理、数据安全可信管理等子系统构成. 具体包括: 数据资源分布图是互联网虚拟数据中心系统的核心数据结构组件, 反映了互联网数据的整体分布情况, 包括数据位置、数据量、数据特征等信息, 是大规模数据采集的指导信息表, 数据资源分布图的访问按照树形结构进行访问; 勘探样本库内保存网络数据勘探器对数据进行勘探时获取的勘探样本; 网络数据勘探器对互联网大数据及协议数据提供方的数据进行勘探, 生成数据资源分布图, 同时将勘探样本放入勘探样本库; 数据资源分布图管理器维护数据分布图数据, 实现对分布图的增删改查; 数据资源分布图服务对外提供数据资源分布图数据的访问服务, 根据资源分布图向数据需求方提供数据采集与挖掘的指导服务, 以保证数据需求用户能高效地、有序地采集挖掘互联网数据及其进一步的分析.

5.2 数据资源分布图模型

数据资源分布图的结构如图 9 所示, 主要包括第 0 层节点 (根节点)、第 1 层节点、第 2 层节点、第 3 层节点 (数据节点). 其中, 第 0 层节点 (根节点)、第 1 层节点、第 2 层节点为初始化层节点, 第 3 层

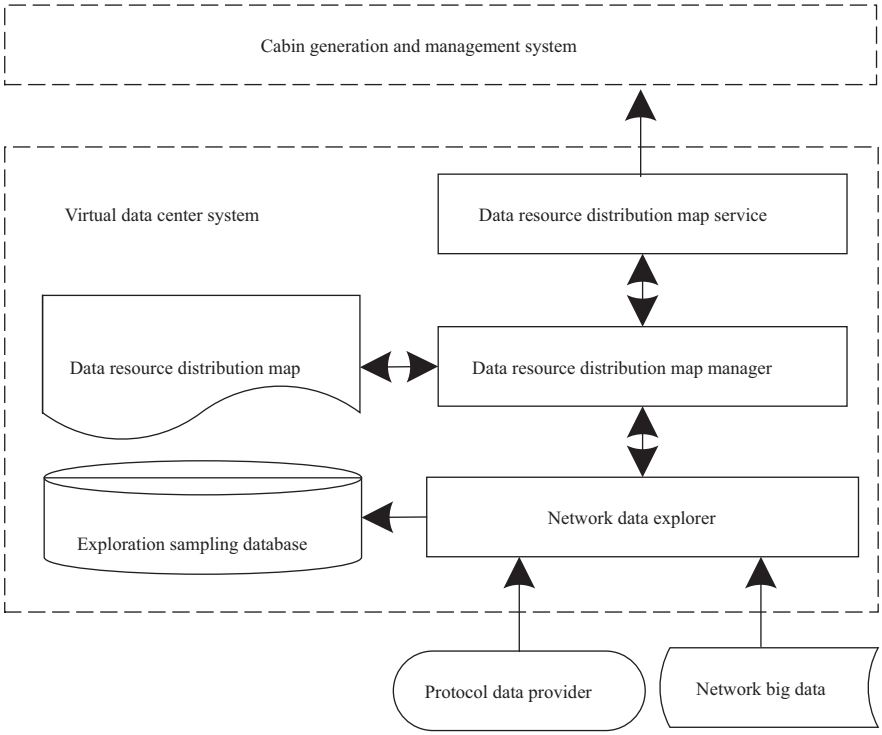


图 8　虚拟数据中心系统的体系结构

Figure 8　The architecture of virtual data center

节点 (数据节点) 为扩展层节点, 4 层节点构成树形结构. 具体结构及其构造描述如下:

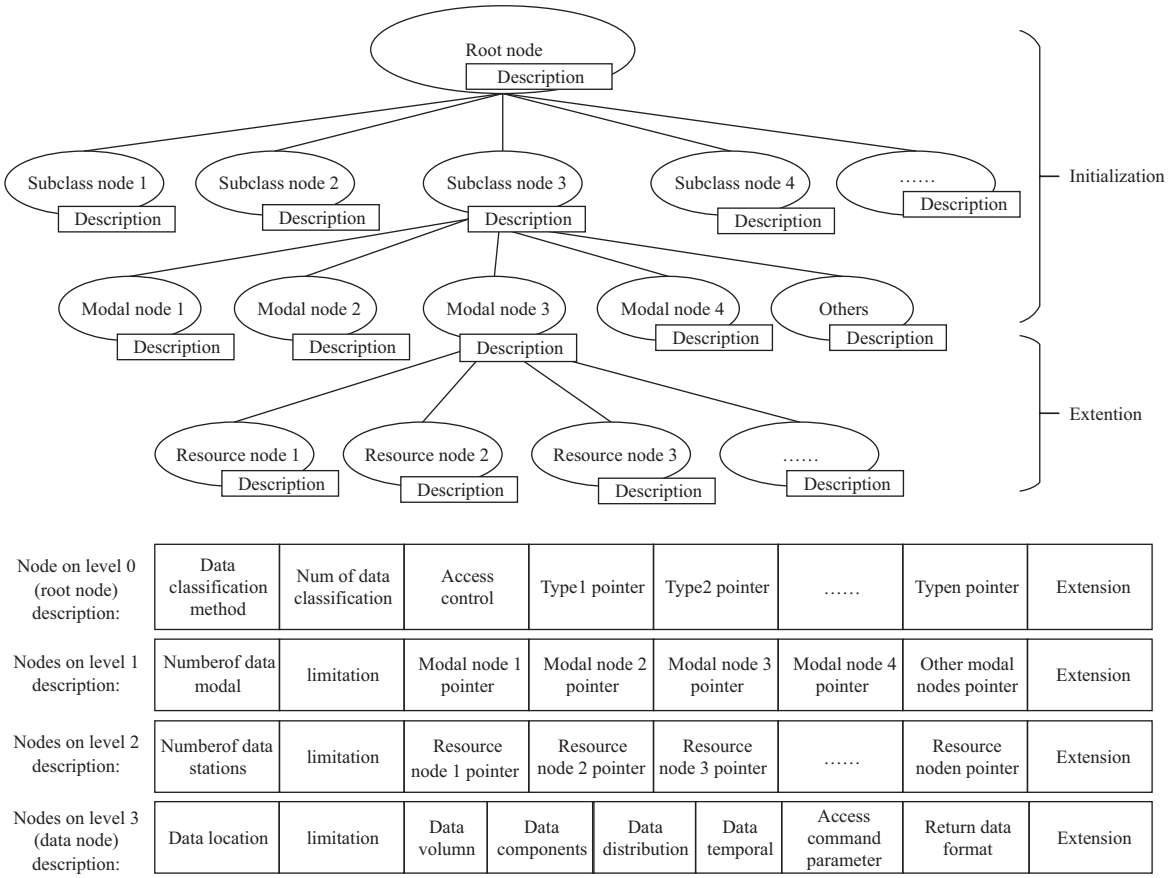
第 0 层节点 (根节点) 主要包括: 数据分类方法、数据分类数量、访问限制、子类别 1 指针、子类别 2 指针、……、子类别 n 指针、扩展项等描述. 其中, 数据分类方法项记录用于数据分类模型或方法; 类别指针用于指向类别节点, 即根节点的每个子节点为一个类别, 扩展项用于信息扩充.

第 1 层节点主要包括: 数据模态数、限制命令、模态节点类指针、扩展项等描述. 数据模态数是指数据模态的分类数, 一般情况指文本、图像、视频、语音, 以及其他等共 5 种数据; 文本类指针、图像类指针、视频类指针、语音类指针、其他类指针是记录指向子节点的链接指针, 其子节点为某种数据模态的节点.

第 2 层节点主要包括: 数据站点数、限制命令、资源节点 1 指针、资源节点 2 指针、……、资源节点 m 指针、扩展项等描述. 数据站点数是指某类某种数据模态下的数据源站点的总个数, 该数量同时表明其孩子的节点数; 资源节点指针记录了其每个子节点.

第 3 层节点为数据节点, 主要包括: 数据位置、限制命令、数据量、数据成分、数据分布、数据时序性、访问命令及参数、返回数据格式、扩展项等描述. 数据位置记录了该数据源的站点位置; 限制命令为访问该数据源的限制访问描述; 数据量为该站点的数据数量, 由数据提供方提供 (也可为空); 数据成分表明数据的组成元素; 数据分布是数据的基本特征及其分布情况; 数据时序性表明数据之间是否为时间序列关系; 访问命令及参数记录访问该数据源的命令及其参数 (也可为空); 返回数据格式是指所获取的数据的格式.

数据资源分布图的管理主要包括数据资源分布图的存储、访问, 以及更新等. 数据资源分布图可以采用关系型或非关系数据库存储, 其逻辑结构为树形结构. 数据资源分布图的访问应当按照树形结



Node on level 0 (root node) description:	Data classification method	Num of data classification	Access control	Type1 pointer	Type2 pointer	……	Typen pointer	Extension
Nodes on level 1 description:	Numberof data modal	limitation	Modal node 1 pointer	Modal node 2 pointer	Modal node 3 pointer	Modal node 4 pointer	Other modal nodes pointer	Extension
Nodes on level 2 description:	Numberof data stations	limitation	Resource node 1 pointer	Resource node 2 pointer	Resource node 3 pointer	……	Resource noden pointer	Extension
Nodes on level 3 (data node) description:	Data location	limitation	Data volumn	Data components	Data distribution	Data temporal	Access command parameter	Return data format	Extension

图 9 数据资源分布图结构

Figure 9 The structure of data resource distribution map

构进行访问. 虚拟数据中心系统的核心在于数据资源分布图的动态更新.

6 方舱实验与分析

方舱计算实现了机动性与定制性的统一、数据资源与物理资源的统筹以及跨域资源的优化协同. 其物理资源配置方案会根据 IT 任务所需的数据资源量和存取位置实现资源动态分配和聚合, 减少数据传输网络延迟, 降低系统运行成本.

为此针对方舱计算核心的资源配置模型, 本文进行了数值实验, 分别使用了线性规划器 Lingo, 用于求解模型的最优解; 使用遗传算法 (GA), 用于快速求解大规模网络资源环境下的方舱资源配置方案. 数值实验运行在 Window 10 操作系统中, CPU 的型号为 AMD R5-2500u, 内存为 8 G 的物理机. 每次实验时, 应用程序独占该物理机.

6.1 实验参数设置

实验参数设置分为两个部分, 首先是模型参数取值: N 的取值范围为 $[1, 100]$; S_j 的取值范围为 $[1, 384]$; W 的取值范围为 $[1, 50]$; V_i 的取值为 $1, 2, 4, 8, 16, 32, 40, 48, 64, 96$; P_i 的取值范围为 $[0.081, 45.696]$; U_i 的取值范围为 $[0.3, 3]$; R_i 的取值范围为 $[40, 80]$; N_i 的取值范围为 $[0.3, 19]$; M_i 的取值范围为 $[75, 600]$;

表 4 实验规模

Table 4 Experiment size

Number	Solution method	Number of data blocks (N)	Number of resource domain centers (W)
1	Lingo GA	5	10
2	Lingo GA	10	20
3	Lingo GA	14	20
4	Lingo GA	20	20
5	Lingo GA	30	28
6	Lingo GA	40	36
7	Lingo GA	60	48
8	Lingo GA	80	65
9	Lingo GA	100	82

$F_{i,j}$ 的取值范围为 $[0, 10000]$;

$$L_{F_{i,j}} = \begin{cases} 0, & \text{if } d_{F_{i,j}} < \varepsilon, \\ 200 + 400 \times d_{F_{i,j}}, & \text{otherwise}, \end{cases}$$

其中, $d_{Fi,j}$ 是根据 min-max 方法将进行归一化的结果.

其次是遗传算法参数设置: 种群大小 400; 终止进化代数 500; 交叉概率 0.4; 变异概率 0.0001.

6.2 实验设置

本次实验共设置了 9 组实验. 针对表 4 设定的前 4 组实验 (后 5 组实验限于篇幅, 就不在此展示具体资源配置方案了) , 可得出表 5 中的详细资源配置方案, 其中 $D_i : C_j(N)$ 表示第 i 个数据块选取了第 j 个资源域中心下的 N 台虚拟机. 前 3 组实验中对比了 Lingo 的最优解和遗传算法的求解结果; 后 6 组实验中, 由于问题规模变大, Lingo 在规定时间内未能得到解, 所以仅获得了遗传算法的求解结果. 由于遗传算法求解具有一定的波动性, 在实验时, 我们对不同规模的问题, 进行了 10 次求解, 以中位数作为本次规模的求解结果.

图 10 对比的是资源域及其资源数量相同, 但是数据块数及其大小不同的第 2 组和第 3 组两组的 GA 实验结果. 从图中可以看出, 对应不同的数据块资源, 根据其所处的位置和所需要的计算资源量, 方舱资源配置算法所选择的资源域及其资源数量也是动态变化和弹性伸缩的.

进一步地, 对两种算法求取的方舱资源配置方案分别计算了它们的资源成本, 以及方案求解所需的算法执行时间, 结果如图 11 和 12 所示, 图中横坐标表示不同的问题规模, 采用 "$N \times W$" 的形式表示. 纵坐标表示不同的指标值.

图 11 所示的是不同资源规模下分配方案的资源成本. 可以看出, 随着问题规模的不断增加, 资源

表 5 资源配置方案

Table 5 Resource configuration scheme

Number	Solution method	Resource configuration scheme
1	Lingo	$D_1 : C_{10}(19)$; $D_2 : C_2(52)$; $D_3 : C_{10}(24)$; $D_4 : C_6(50)$; $D_5 : C_{10}(21)$
	GA	$D_1 : C_{10}(12)$; $D_2 : C_6(24)$; $D_3 : C_{10}(19)$; $D_4 : C_{10}(12)$; $D_5 : C_{10}(13)$
2	Lingo	$D_1 : C_{20}(52)$; $D_2 : C_{18}(7)$; $D_3 : C_{18}(7)$; $D_4 : C_{18}(6)$; $D_5 : C_{10}(32)$ $D_6 : C_{17}(13)$; $D_7 : C_{17}(6)$; $D_8 : C_{17}(4)$; $D_9 : C_{10}(27)$; $D_{10} : C_{18}(16)$
	GA	$D_1 : C_{18}(14)$; $D_2 : C_{10}(27)$; $D_3 : C_9(25)$; $D_4 : C_{15}(13)$; $D_5 : C_{18}(6)$ $D_6 : C_{18}(8)$; $D_7 : C_{10}(10)$; $D_8 : C_2(12)$; $D_9 : C_{18}(8)$; $D_{10} : C_{18}(15)$
3	Lingo	$D_1 : C_{10}(22)$; $D_2 : C_{14}(29)$; $D_3 : C_{10}(21)$; $D_4 : C_{16}(27)$; $D_5 : C_{14}(29)$ $D_6 : C_5(46)$; $D_7 : C_{12}(54)$; $D_8 : C_{20}(27)$; $D_9 : C_{10}(16)$; $D_{10} : C_{16}(18)$ $D_{11} : C_5(25)$; $D_{12} : C_{11}(59)$; $D_{13} : C_{20}(25)$; $D_{14} : C_{17}(62)$
	GA	$D_1 : C_{10}(12)$; $D_2 : C_{14}(3)$; $D_3 : C_{10}(2)$; $D_4 : C_{10}(22)$; $D_5 : C_{20}(18)$ $D_6 : C_{16}(14)$; $D_7 : C_{10}(14)$; $D_8 : C_{14}(6)$; $D_9 : C_{16}(16)$; $D_{10} : C_{20}(12)$ $D_{11} : C_{14}(13)$; $D_{12} : C_5(23)$; $D_{13} : C_5(10)$; $D_{14} : C_{10}(8)$
4	GA	$D_1 : C_{15}(5)$; $D_2 : C_9(16)$; $D_3 : C_{14}(4)$; $D_4 : C_{16}(22)$; $D_5 : C_{10}(20)$ $D_6 : C_6(8)$; $D_7 : C_{20}(7)$; $D_8 : C_{14}(18)$; $D_9 : C_{10}(17)$; $D_{10} : C_{16}(14)$ $D_{11} : C_{10}(5)$; $D_{12} : C_{14}(18)$; $D_{13} : C_{10}(10)$; $D_{14} : C_{20}(17)$; $D_{15} : C_5(18)$; $D_{16} : C_{14}(4)$; $D_{17} : C_{13}(3)$; $D_{18} : C_2(16)$; $D_{19} : C_{14}(8)$; $D_{20} : C_{20}(11)$

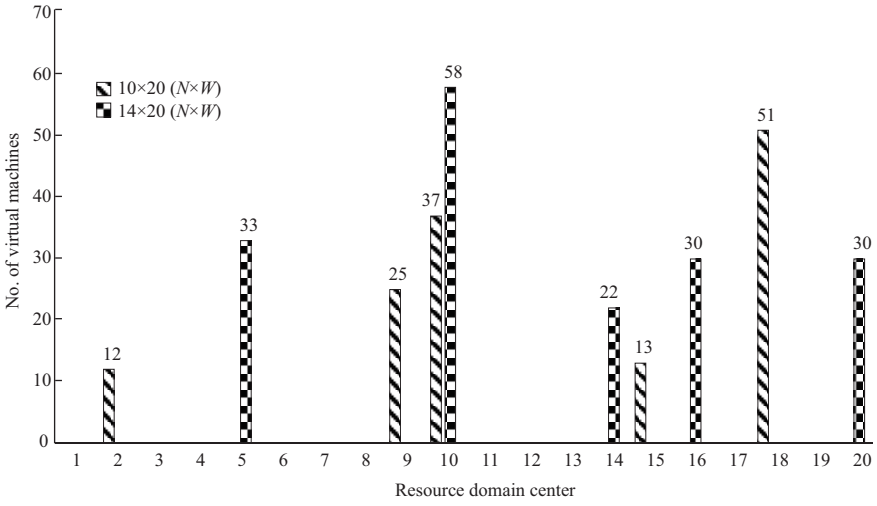


图 10 方舱数据资源配置方案对比分析

Figure 10 Comparative analysis of cabin resource allocation scheme (N: the number of data blocks; W: the number of resource domain centers)

成本也随之增加. 前 3 组实验结果表明 GA 计算得出的成本与 Lingo 得出成本的误差在 10% 之内. 因此, GA 获得的近似最优解可以满足业界认可的精度. 当规模超过 20×20 时, 由于 Lingo 运行时间较长, 因此, 中断了求解过程, 没有获取到最优解. 所以, 在图 11 中只展示了遗传算法求取的方舱资源配置方案的资源成本.

图 12 所示的是不同资源规模下方舱资源配置问题的求解时间. 在前 3 组实验中, 可以看出随着问题规模的不断增大, Lingo 的求解时间也在不断增加. 后 6 组实验展示的是 Lingo 算法程序中断前

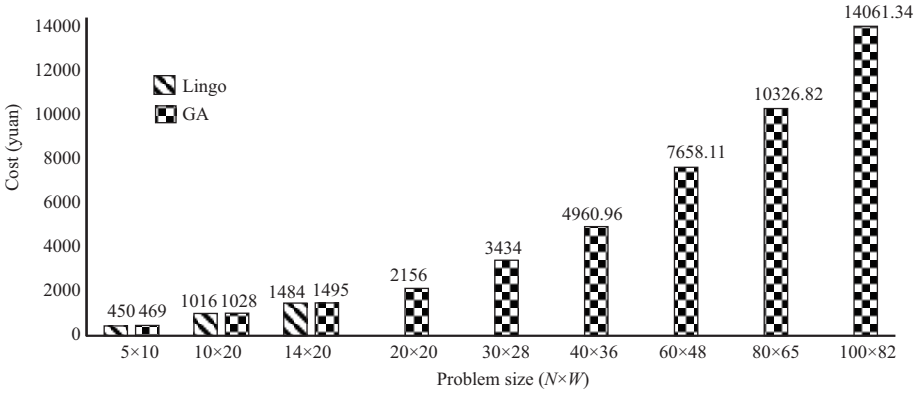


图 11　不同资源规模下的分配方案资源成本

Figure 11　Resource cost of allocation schemes under different resource scales

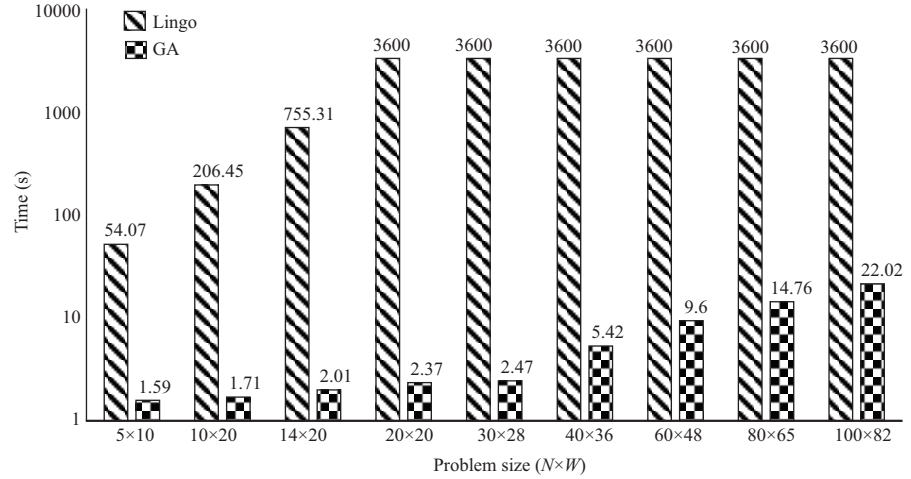


图 12　不同资源规模下的问题求解时间

Figure 12　Problem solving time under different resource scales

的运行时间 (此时还未获得求解结果). 不同于 Lingo 算法随着问题规模的增大, 算法求取时间也呈快速增长, 当在小规模时, 遗传算法的运行时间从 1.59 s 增加至 2.47 s, 求解时间波动较小; 当在较大规模时, 遗传算法求解时间增加较为明显, 这是由于随着规模的增大出现非法个体的数量也增大, 所以在遗传算法的迭代过程中对非法个体进行替换以保证取得最优解的时间开销也相应增加.

从图 11 和 12 中, 可以看出, 不同规模下, 遗传算法可以获得接近于 Lingo 最优解的结果, 同时, 求解时间也大幅度减小. 基于遗传算法的方舱资源配置方案求解方法是有效且可行的.

7　结语

方舱计算本质是针对网络环境下的业务对于自身敏捷构造需求和外部大数据应用需求所设计的新的计算模式和解决方案. 相较于网格计算、虚拟组织、云计算、边缘计算等计算模式, 方舱计算实现了机动性与定制性的统一、数据资源与物理资源的统筹以及跨域资源的优化协同.

下一步, 我们将对方舱计算模式进一步细化和优化, 以支持细分领域和不同业务的个性化和差异

化. 同时, 我们将研发实现方舱计算的各类系统, 实现方舱的按需搭建、资源配置与动态消亡, 并在网络金融、O2O、智慧城市等领域开展示范性应用.

参考文献

1 Xu Z W, Feng B M, Li W. Grid Computing Technology. Beijing: Publishing House of Electronics Industry, 2004 [徐志伟, 冯百明, 李伟. 网格计算技术. 北京: 电子工业出版社, 2004]

2 Liu L. Grid resource management and positioning. Electron Technol Softw Eng, 2017, 24: 10–11 [刘磊. 网格资源管理与定位. 电子技术与软件工程, 2017, 24: 10–11]

3 Toporkov V, Yemelyanov D, Bobchenkov A, et al. Preference-based economic scheduling in grid virtual organizations. Procedia Comput Sci, 2016, 80: 1071–1082

4 Mashayekhy L, Grosu D. A merge-and-split mechanism for dynamic virtual organization formation in grids. IEEE Trans Parallel Distrib Syst, 2014, 25: 540–549

5 Shen Y, Bao Z F, Qin X L, et al. Adaptive task scheduling strategy in cloud: when energy consumption meets performance guarantee. World Wide Web, 2017, 20: 155–173

6 Khazaei H, Misic J, Misic V B. Performance analysis of cloud computing centers using M/G/m/m+r queuing systems. IEEE Trans Parallel Distrib Syst, 2012, 23: 936–943

7 Liu Y, Li A, Liu S, et al. Autonomic self-testing of regression and internationalization based on cloud computing. In: Proceedings of the 11th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Qiqihar, 2019. 141–144

8 Ding C T, Cao J N, Yang L, et al. Edge computing: applications, state-of-the-art and challenges. ZTE Technol, 2019, 025: 1–7 [丁春涛, 曹建农, 杨磊, 等. 边缘计算综述: 应用、现状及挑战. 中兴通讯技术, 2019, 025: 1–7]

9 Guo W Z, Lin B, Chen G L, et al. Cost-driven scheduling for deadline-based workflow across multiple clouds. IEEE Trans Netw Serv Manage, 2018, 15: 1571–1585

10 Silva P, Perez C, Desprez F. Efficient heuristics for placing large-scale distributed applications on multiple clouds. In: Proceedings of the 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2016. 483–492

11 Ferry N, Chauvel F, Song H, et al. CloudMF: model-driven management of multi-cloud applications. ACM Trans Inte Technol, 2018, 18: 1–24

12 Shi P C, Wang H M, Zheng Z B, et al. Collaboration environment for JointCloud computing. Sci Sin Inform, 2017, 47: 1129–1148 [史佩昌, 王怀民, 郑子彬, 等. 面向云际计算的自主对等协作环境. 中国科学: 信息科学, 2017, 47: 1129–1148]

13 Shi P C, Yin H, Wo T Y, et al. Research progress on basic theory and method of cloud computing defined by software. Basic Sci China, 2019, 6: 54–60 [史佩昌, 尹浩, 沃天宇, 等. 软件定义的云际计算基础理论和方法研究进展. 中国基础科学, 2019, 6: 54–60]

14 Xu G D, Shi Y C, Xie W K. Pervasive computing. Chinese J Comput, 2003, 26: 1042–1050 [徐光档, 史元春, 谢伟凯. 普适计算. 计算机学报, 2003, 26: 1042–1050]

15 Jiang C J, Li Z. Mobile Information Service for Networks. Berlin: Springer, 2020

16 Wang X Y, Zhu J K, Shen Y H. Network-aware QoS prediction for service composition using geolocation. IEEE Trans Serv Comput, 2015, 8: 630–643

17 Bao L, Wu C Q, Bu X X, et al. Performance modeling and workflow scheduling of microservice-based applications in clouds. IEEE Trans Parallel Distrib Syst, 2019, 30: 2114–2129

18 Guo Y F, Lama P, Jiang C J, et al. Automated and agile server parameter tuning by coordinated learning and control. IEEE Trans Parallel Distrib Syst, 2014, 25: 876–886

19 Cheng D Z, Guo Y F, Jiang C J, et al. Self-tuning batching with DVFS for performance improvement and energy efficiency in internet servers. ACM Trans Auton Adapt Syst, 2015, 10: 1–32

20 Jiang C J, Zhang Z H, Wang P W, et al. China Patent, 2019109266982, 2019 [蒋昌俊, 章昭辉, 王鹏伟, 等. 中国专利, 2019109266982, 2019]

21 He Y, Wang C, Jiang C J. Discovering canonical correlations between topical and topological information in document networks. IEEE Trans Knowl Data Eng, 2018, 30: 460–473

Cabin computing

Changjun JIANG[1,2*], Zhijun DING[1,2], Jian YU[1,2], Zhaohui ZHANG[1,2], Chungang YAN[1,2], Yaying ZHANG[1,2] & Pengwei WANG[1,2]

1. *Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai* 201804*, China*;
2. *Network Information Service Shanghai Engineering Research Center, Tongji University, Shanghai* 201804*, China*
* Corresponding author. E-mail: cjjiang@tongji.edu.cn

Abstract With the rapid development of information technology innovation, digitalization, networking and intelligence are deeply developing. New applications are emerging and new industries are flourishing, which puts forward higher requirements for the agile construction and continuous operation and maintenance of business systems. This paper proposes a new computing model called cabin computing. Cabin computing is the cross-domain resource configuration and collaborative computing integration environment for the full life cycle of IT tasks accessed through the Internet. Its core idea is "mobile special cabins, flexible cross-domain resource, and autonomous system operation and maintenance". This paper presents the architecture and working principle of the cabin computing system. It is mainly composed of cabin generation and management system, cross-domain resource management system, virtual data center system and several gateways (cabin gateway, virtual data center gateway, cross-domain resource gateway, etc.). Furthermore, the definition of resource allocation optimization in cabin computing is stated. Taking data resources, computing resources and storage resources into account, the resource cost is minimized. The simulation experiments of the programming solver and approximate optimization algorithm demonstrate that the proposed approach can realize the optimal allocation of cabin resources considering the distribution of both data resources and physical resources.

Keywords cabin, virtual data center, cross-domain resource management, resource allocation, resource distribution map, resource directory

Changjun JIANG was born in 1962. He received his Ph.D. degree from Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1995. Currently, he is a professor at the Department of Computer Science and Technology, Tongji University, Shanghai, China. His current research interests include network computing, concurrency theory, formal verification of software, intelligent systems, and service-oriented computing.

Zhijun DING was born in 1974. He received his Ph.D. degree from Tongji University, Shanghai, China, in 2007. Currently, he is a professor at the Department of Computer Science and Technology, Tongji University. His research interests are in formal engineering, Petri nets, services computing, and mobile internet.

Jian YU was born in 1975. He received his Ph.D. degree in circuits and systems from South China University of Technology, Guangzhou, in 2010. Currently, he is an engineer at Tongji University. His research interests include the Internet of things, embedded systems, and cloud computing.

Zhaohui ZHANG was born in 1971. He received his Ph.D. degree in computer science and technology from Tongji University, Shanghai, China. He is a professor at the School of Computer Science and Technology, Donghua University, Shanghai. His research interests include networks computing, big data processing, and intelligent systems.