基于约束的空间众包多阶段任务分配

范泽军 沈立炜 彭 鑫 赵文耘

(复旦大学计算机科学技术学院 上海 200433)

(复旦大学上海市数据科学重点实验室 上海 200433)

摘 要 在人机物融合的背景下,空间众包可被视为一种新型的软件服务.空间众包是针对于物理世界中与地理位置和时间要素相关的众包任务,通常要求参与者真实地移动到指定位置执行相应操作,承担数据收集与感知等任务,并通过移动终端反馈操作结果.任务分配是空间众包中的关键技术,其目的是在满足时空约束条件的前提下选取一个或一组合适的参与者承担任务的执行.在具有空间位置访问权限、线下服务使用权限并考虑参与者可用时间与移动范围约束的空间众包场景中,单个参与者可能无法完成整个任务的执行,转而需要一组参与者进行协同合作后才能完成.针对该需求,本文提出了一种基于约束的空间众包多阶段任务分配方法.该方法的核心算法将首先根据两种不同的优化目标获取对应的任务路径集合,然后针对每一条路径采用从起点和终点双向选取局部最优的方法递归地选择承担阶段性任务的参与者.通过上述步骤,可高效地将任务分解为一组由不同参与者在符合约束条件时能够承担的阶段性子任务,以此提高任务完成的概率.最后,我们使用上述算法分别进行了模拟实验和真实场景实验.模拟实验面向随机生成的参与者数据集,并与基于动态规划取得全局最优解的算法进行对比.实验结果表明本文所提算法相比于对比算法具有更好的计算效率.真实场景实验依赖自主开发的校园空间众包任务平台开展.实验结果表明本文算法能够在真实任务环境下推导得到参与者可接受的子任务分配决策.

关键词 空间众包;任务分配;多阶段;时空约束;空间拓扑

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2019.02722

Multi Stage Task Allocation on Constrained Spatial Crowdsourcing

FAN Ze-Jun SHEN Li-Wei PENG Xin ZHAO Wen-Yun

(*School of Computer Science, Fudan University, Shanghai* 200433)

(*Shanghai Key Laboratory of Data Science, Fudan University, Shanghai* 200433)

Abstract In the context of the fusion of the ternary human-machine-thing, spatial crowdsourcing is regarded as a new type of software service. Spatial crowdsourcing works for crowdsourcing tasks which are related to the geographic and temporal elements in the physical world. The involved workers are usually required to move to the specified location to perform the corresponding operations, to undertake tasks such as data collection and perception, and to feedback the operation results through the mobile terminal. Task allocation has been a key technique in the field of spatial crowdsourcing. It aims to select one or a group of suitable workers to undertake the execution of the task while satisfying the constraints of time and space. The existing spatial crowdsourcing research work mostly works for simple tasks, which can usually be undertaken by one worker alone. Even in a multi-tasking scenario, each simple task is basically independent. A single worker may undertake one or more tasks, and the execution of a single simple task does

收稿日期:2018-06-05;在线出版日期:2019-03-29.本课题得到国家重点研发计划"人机物融合的云计算架构与平台"(2018YFB1004800)资助.**范泽军**,硕士研究生,主要研究方向为人机物融合、移动云计算软件系统、边缘计算.E-mail:16212010007@fudan.edu.cn.**沈立炜**(通信作者),博士,副教授,中国计算机学会(CCF)会员,主要研究方向为人机物融合、移动应用开发与分析.Email:shenliwei@fudan.edu.cn.**彭 鑫**,博士,教授,中国计算机学会(CCF)会员,主要研究领域为移动计算与云计算、代码大数据、智能化软件开发.**赵文耘**,硕士,教授,中国计算机学会(CCF)会员,主要研究领域为软件工程、软件开发工具及其环境、企业应用集成.

not have to be split and assigned to multiple workers. However, in constrained spatial crowdsourcing, a single worker may not be able to complete the entire task. Constrained spatial crowdsourcing involves location and service authorization, worker available time and worker movement range. Instead, a task in constrained spatial crowdsourcing requires a group of workers to complete the task collaboratively. In response to this demand, we propose a multi-stage task allocation method on constrained spatial crowdsourcing in this paper. The core algorithm of the method will first obtain the path set corresponding to the task according to two different optimization objectives including the minimum execution time and the minimum movement range, and then sort them according to the priority. For each path in the set, the algorithm will recursively select the workers to undertake the phased tasks by means of adopting a method from which the local optimal results are obtained in both directions from the starting point and from the ending point. Once the tasks on the path can be collaboratively completed by one or a group of workers, an allocation decision for the phased task is obtained. Through the above steps, the method efficiently breaks down the task into a set of sub-tasks that can be undertaken by different workers when the constraints are met, thereby increasing the probability of completion of the task. Finally, a simulation experiment and a field experiment were carried out based on the above algorithm. In the simulation experiment, we compare our algorithm with a dynamic programming algorithm which can obtain the global optimal solution based on the same randomly generated data set. The data set includes the workers' restricted location and service authorization status, movement range and current position coordinate. The experimental results show that our algorithm has better computational efficiency than the comparison algorithm. The field experiment is performed by utilizing a self-developed campus spatial crowdsourcing task platform. The experimental results show that our algorithm can derive sub-task allocation decisions that the workers can accept in real spatial crowdsourcing environment.

Keywords　spatial crowdsourcing; task allocation; multi stage; spatial-temporal constraint; spatial topology

1　引　言

人机物融合的智能技术是发展新经济的关键技术之一[1]. 在此背景下，人类社会中个体或群体的智慧与劳动力能够参与人机物系统的运行，而众包(Crowdsourcing)[2]则是将这些不同个体或群体的能力进行组合的一种软件服务[3]. 在概念方面，众包指以公开召集的方式将传统上由指定人员执行的任务外包给未预先指定的且通常具有庞大人员基数的群组的行为. 利用众包机制，任务的执行能够充分依赖群体智慧[4]，获得更为广泛的知识来源，从而提高任务完成的效率与质量，并为人机物系统提供了融合人类社会群体协作的途径. 此外，互联网的发展催生出了一系列基于 Web 的在线众包平台，例如 Amazon Mechanical Turk、oDesk 等. 任务请求者(requester)可在众包平台上发布众包任务，已注册于平台的参与者(worker)能够选择并承担选定的任务. 从 2009 年开始，众包得到了各个领域的关注[5]，在诸如自然语言理解[6]、图片标注[7]、软件测试[8]等领域得到广泛应用.

除此以外，还有许多众包任务与物理世界的地理位置与时间要素相关，例如获取道路的实时路况信息、拍摄建筑物的各角度照片等. 针对这些类型任务的众包被称为空间众包(Spatial Crowdsourcing)[9]，它要求参与者真实地移动至指定的位置执行相应的操作，承担数据收集与感知等任务，并通过移动终端反馈操作结果. 空间众包可被视为人机物融合场景下的一种新型软件服务. 与传统众包相同，空间众包也面临着任务分配、质量管理、用户激励、隐私保护等一系列挑战[10]. 其中，任务分配的目的是在满足特定限制条件的前提下从候选参与者中选择合适的一个或一组参与者来承担任务的执行[11]. 在此背景下，许多研究者提出了各种方法来计算空间众包任

务的最优的参与者分配方案[12].这些方法基于不同的策略解决分配的最优化问题.典型的策略包括最大化完成任务的质量[13]以及最小化系统成本与开销[14-15].从任务分配的结果来看,已有的空间众包研究工作大多针对包括单个操作目标的简单任务,例如道路实时路况的获取.这些简单任务一般都能够由一位参与者单独承担.即使在多任务的场景下,每一项任务也基本独立,一个参与者可能承担其中的一项或者多项任务[16],但对于单项任务而言,其执行过程无须拆分并分配给多个参与者.

现实场景中存在另一类任务.相比于简单任务,这类任务包括一组与位置和线下服务相关的有序操作.其中,(1)位置是能够被人访问的现实世界中的地点,例如图书馆、体育馆等.各个位置间的通道形成了一张网络.在空间众包中,对位置的访问要求参与者真实移动到指定地点执行指定的操作,例如将物品递送到目标地点或者前往目标地点获取某些服务;(2)线下服务位于特定的位置,参与者需到达该位置才能使用服务,且服务的执行需要花费一定的时间.例如,图书馆的借阅服务或设备的登记服务.

另外,这类空间众包任务在分配过程中需要考虑两个方面的约束.(1)位置的访问权限与服务的使用权限.空间中某些位置具有访问权限,即只有经过授权的人员才能进入该位置所代表的空间.某些服务具有使用权限,即具备使用授权或能力的人员才被允许使用.例如,扫描仪仅能被具有权限的人员使用;(2)参与者的可用时间与移动范围的约束.这两个约束是参与者所指定的愿意执行任务的时间区间与空间活动范围.在分配过程中,时间区间之外的,或超过移动范围的任务不应分配给该参与者.

我们将具有以上所述特点的空间众包称为基于约束的空间众包.这种类型的众包任务可能无法由单个参与者单独承担,因此有些情况下需要将其拆分为多个具有依赖关联的阶段子任务并由一组参与者协同完成.

传统的空间众包分配方法无法有效应对这种新类型的众包任务.针对该挑战,本文对基于约束的空间众包的任务分配展开研究,提出了一种多阶段任务分配方法.该方法可高效地将任务分解为一组由不同参与者在符合约束条件时能够承担的阶段性子任务,以此提高任务完成的概率.在分配过程中,本方法的核心算法根据任务执行的总用时最少与移动距离最短这两个优化目标推导分配方案.我们使用该算法进行了实验,实验结果表明本方法具有较高的计算效率,并且在真实任务环境下能推导得到参与者可接受的子任务分配决策.

本文第2节通过两个动机案例展现本文所提方法的使用场景及效果;第3节定义基于约束的空间众包任务分配的概念、流程与组成要素;第4节具体描述考虑时空约束的空间众包多阶段任务分配算法;第5节介绍实验设计并对实验结果与方法进行讨论;第6节列举相关工作;最后是本文的总结与对未来工作的展望.

2 动机案例

本节通过两个动机案例对基于约束的空间众包多阶段任务分配进行解释.动机案例1面向真实应用场景,探讨多阶段任务分配的应用价值与效果;动机案例2针对在模拟空间内的任务执行需求,解释多阶段任务分配的工作方式与执行结果.

2.1 动机案例1

同城配送(如达达、蜂鸟众包、人人快递、飞毛腿等)提高了同一城市内物品递送的效率.它采用众包的方式,主要流程包括:任务发布者在平台发布需求,通常为物品派送类任务;任务参与者在平台上接受任务;任务参与者到达指定起点获取物品,送至指定终点.同城配送是一种典型的众包模式,其任务参与者不一定是专业的快递员,普通民众也可参与寄送任务的执行.

在众包任务的执行过程中,众包参与者由于特定的约束条件,无法按照平台既定的路线完成端到端的派送任务.影响参与者执行任务的主要限制条件包括:(1)参与者未拥有某些特定地点的准入权限,如学校、企业、政府机构等大部分单位的大门需要相应的权限才能进入.若参与者不能进入某个单位,则必须花费额外工作量与时间等待客户提取.如果客户不在,则可能当天无法寄送;(2)参与者需要前往远离自己平时活动范围的地方执行任务,如参与者平时在东城区活动,若承接的众包任务需要前往西城区执行,会出现路线不熟悉等情况,影响众包任务的完成效率;(3)若众包任务分配不得当,参与者在执行任务的时间段可能有其他重要事项,则会出现参与者工作积极性与工作效率降低的情况,影响众包任务的完成质量.

针对这种场景,若采用多阶段任务分配,允许多名参与者协作完成任务,则可降低上述约束带来的影响.在物品速递业务中,众包参与者可浏览当前任务,

报名参与任务，同时提交自己在该城市中的活动意愿与可用时间范围. 通过计算，参与者可被指派承担不同路线阶段的任务，相比传统的方式，多阶段任务分配可以在以下几个方面提高众包任务的完成效率与质量.

(1) 针对参与者由于权限问题无法进入某单位完成众包任务的情况，平台可以邀请该单位内的相关人员承接该任务的最后一段，通过在单位门口进行交接的方式完成整体任务.

(2) 平台按照参与者在任务报名时提交的活动意愿，给参与者分配相应任务阶段，而并不要求参与者完成整个任务. 如对于平时在东城区活动的参与者，只需要执行任务在东城区的部分，将物品传递至两区交界处，再由另一个负责西城区的参与者接力完成剩余的部分.

(3) 平台按照参与者在任务报名时提交的可用时间范围，给参与者分配最合适的任务. 如某参与者白天正常上班，晚上 8 点到 11 点是他的可用时间范围，那么平台只会给他分配涉及时间段在晚上 8 点到 11 点之间的任务.

2.2　动机案例 2

假设众包任务发生在某单位中. 图 1(a)表示了该单位的涵盖位置与服务信息的拓扑结构. 其中矩形节点表示空间位置，代表空间内的房间等建筑单元，其解释见图 1(b). 结点之间的连线表示位置之间的通道，连线上的标记表示从一个位置到另一个位置的距离以及估算的移动时间. 圆角矩形表示单位提供的线下服务，其解释见图 1(b). 这些服务位于特定的位置(位于矩形框内)，括号内的分钟数表示该服务的预估执行时间. 另一方面，某些空间位置与线下服务具有访问权限与使用权限，我们使用矩形与圆角矩形左上方的锁图标表示这类权限.

Bob、Charlie 与 David 是愿意接受众包任务的三位参与者. 他们在众包平台上指定了当天的可用时间(图中人形图标下方的时间区间，例如 Bob 在 13:00 至 14:30 内愿意承担任务)，以及当天愿意移动的范围(图 1(b)表格中灰色底色标记的位置范围，例如 Bob 可接受前往 A、B、C、D 的任务). 另外，图 1(b)表格中的钥匙图标表示参与者能够访问相应的位置或者使用相应的服务. 例如，Charlie 具有进入位置 C(会议室)、D(打印室)以及使用 SD(文件打印)等权限或能力.

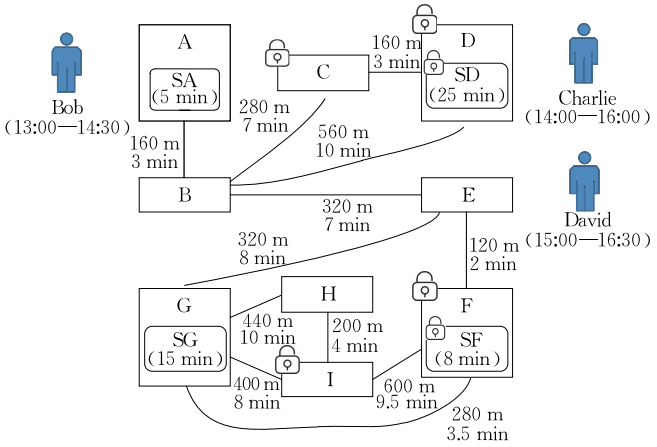
Alice 是该单位某项目负责人，由于出差原因无法亲自执行某工作流程，因此她在众包平台上发布任务：“我需要找人代替我从收发室领取一个装有合同文档的 U 盘，并前往文件打印室进行打印，然后在财务处进行文件盖章后送至我的办公室”.

当任务发布时，Bob、Charlie 与 David 分别处于位置 A、D、E 附近. 由于位置与服务存在着各类权限要求，三位参与者针对这些权限的授权情况各不相同，同时参与者自身具有可用时间与移动范围的约束，因此无法找到一个单独的参与者完成所有的任务操作.

本文所提出的任务分配方法面对上述问题，基于位置服务拓扑，在考虑参与者的授权情况以及个人约束的前提下按照特定的优化目标将任务分解为多个阶段的子任务，并将其分派给指定的参与者.

在该过程中，申请者发布的任务描述首先被映射成一组操作序列：(1) 前往 A 使用 SA；(2) 前往 D 使用 SD；(3) 前往 G 使用 SG；(4) 前往 I.

随后众包平台的任务分配算法可按照两类分配优化目标中的一个推导子任务分配方案. 第一类优化目标是“用时最少”，即尽量减少参与者整体完成



(a) 空间服务拓扑

			Bob	Charlie	David
位置	A	收发室			
	B	休息室			
	C	会议室	🔑	🔑	
	D	打印室	🔑	🔑	
	E	公共办公区			
	F	归档处		🔑	🔑
	G	财务处	🔑		🔑
	H	茶水室			
	I	办公室		🔑	🔑
服务	SA	物品领取			
	SD	文件打印		🔑	
	SF	数据归档		🔑	🔑
	SG	文件盖章			🔑

(b) 参与者移动范围与权限

图 1　示例场景

任务的总体耗时.该类优化目标适用于时间敏感类型任务的分配,即众包任务对总体执行时间有要求.典型场景包括:(1)发布者在任务描述中规定了任务的完成时间;(2)任务涉及的空间位置或线下服务在任务发布后可能变为不可达或关闭状态(此类场景要求在尽量短的时间内进行服务调用).第二类优化目标是"移动距离最短",即尽量缩短参与者协作完成任务的移动总距离.该类优化目标适用于非时间敏感,但希望降低总体空间移动距离的任务.典型场景包括:任务涉及的物品具有特殊性,例如易碎、易融化、具有较大体积或重量等.图 2 展示了基于两类优化目标得到的子任务分配方案.

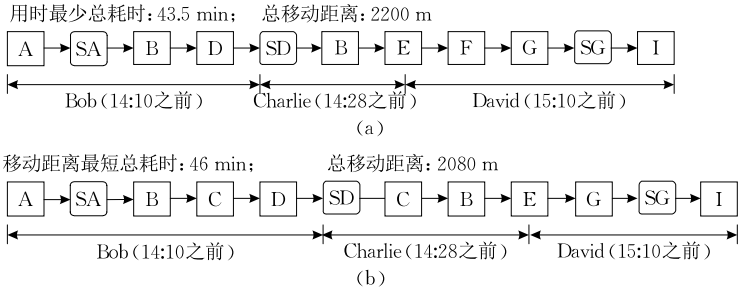


图 2 示例任务的多阶段分配方案

基于用时最少的优化目标,众包平台尝试将 Alice 提交的任务进行分解,得到图 2(a)所示方案.第一阶段由处于 A 处附近的 Bob 承担,建议其于 14:10 之前达到 A(收发室),在领取物品后通过 B(休息室)送往 D(打印室).由于 Bob 不具备文件打印相关的权限,因此平台将第二阶段的子任务委托给处于 D 处附近的 Charlie,并要求 Charlie 在 14:28 之前到达打印室交接物品并进行 SD 服务(文件打印)的操作.由于 Charlie 不愿意前往 F(归档处),因此平台将安排 Charlie 将打印好的文件及 U 盘通过 B 送至 E(公共办公区)并将物品移交给处于 E 处附近的 David.David 承担第三阶段的子任务,他被指示在 15:10 前到达 E 进行交接,随后通过 F(归档处)前往 G(财务处),并使用 SG 服务(文件盖章),最后将物品送至最终的目的地 I(办公室).在该方案中,算法选取了耗时最短的路径,同时考虑了三位参与者的权限、可用时间与移动范围.根据计算得到的子任务向参与者指派了移动路径,以及开始承担任务的建议时间.

类似地,以移动距离最短为优化目标,得到图 2(b)的子任务分配方案.与(a)方案相比,在进行任务分配时选取距离最短的路径,随后根据权限与参与者约束得到三阶段的子任务.其中,对于 Bob 而言,要求他通过 C(会议室)前往 D(打印室);对于 Charlie 而言,不要求他通过 F(归档处).

3 基于约束空间众包的定义

本节首先给出了基于约束空间众包的概念及其基本流程,随后对流程中涉及的要素进行定义.

3.1 基于约束空间众包的概念

文献[14]对空间众包的关键技术,尤其是任务分配进行了综述,其中提出了一个空间众包的概念模型,通过任务、参与者与服务器这三类要素来描绘众包的体系.基于该模型,图 3 展示了本文所提出的空间众包的概念及其定位.

任务	参与者	任务分配
任务类型 紧急任务 定点任务 静态任务 时空特性 固定的空间位置 指定的时间约束 完成任务的方式 群体协作	时空特性 指定的可用时间 实时位置 技能 空间进入权限 服务使用权限 偏好 指定的移动范围	优化目标 移动距离最短 完成时间最少 单任务分配 离线分配 服务器指定分配 主动地移动

图 3 多阶段空间众包的属性

我们将基于约束空间众包的任务类型定义为紧急任务、定点任务与静态任务.紧急表示任务需要由参与者在受限的时间期限内尽快完成;定点表示任务的执行与特定的位置点相关,即使用线下服务或递送物品均需精确到一个具体的位置;静态表示与任务相关的位置信息不会随着时间发生变化.其次,任务的时空特性描述了发布任务在时间与空间维度的约束,我们将其定义为固定的空间位置与指定的时间约束.前者表示与任务相关的位置以及服务信息由空间拓扑预先决定;后者则表示任务的完成时间期限.最后,基于约束的空间众包需要由一组参与者通过群体协作的方式实现最终的目标.

参与者的时空特性是任务分配必须考虑的因

素.对于一个参与者而言,他可指定可用的时间,即在该时间范围内可被分派任务.另外,参与者的实时位置也能通过各种手段获取.其次,参与者的技能对任务分配也产生影响.在基于约束的空间众包中,我们将参与者的技能等价于访问特定位置的权限,以及使用特定线下服务的权限.最后,参与者可指定其移动范围的约束,即不能接受前往超过该范围位置点的任务.参与者的可用时间与移动范围被认为是影响任务分配的重要约束.

在空间众包中,服务器负责数据的存储、处理与任务的分配.我们重点定义其中的任务分配.本文定义了两类优化目标来推理最优的参与者任务分配方案,分别是完成时间最少和移动距离最短.本文方法主要针对单任务的分配,并未考虑多任务存在时的分配策略.在分配过程中,我们(1)采用离线分配的方式,即在任务真实执行前决定分配方案;(2)使用服务器分配任务[11]的方式,即在知晓所有参与者当前位置的情况下由服务器自动推理分配方案;(3)假设用户在接受任务之后能够主动地向目标位置移动,而不是借由其他的原因移动到目标位置时顺带着完成任务.

当前,本文所述的空间众包暂未考虑参与者激励、信任、隐私保护等方面的因素.

3.2　基于约束空间众包的流程

图 4 展示了基于约束的空间众包的基本流程.

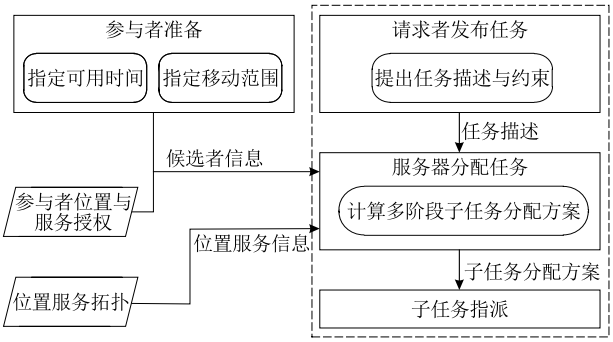


图 4　多阶段空间众包的流程

流程中的任务分配建立在两个预先定义的知识集之上,分别是参与者的位置与服务授权以及位置服务拓扑.参与者的授权信息指明了参与者具有访问指定位置和使用指定线下服务的能力;位置服务拓扑则表示了一组位置与服务在物理空间维度上的结构与关联.

参与者可在任意时刻决定是否开始接受任务的指派.在该准备时刻,参与者需要指定其能够承担任务的时间区间以及移动范围.每一次众包任务的分配都是从已准备好并空闲的参与者中选取一个最优集合,相互协作完成整体的任务目标.在此过程中,任务的请求者首先在众包平台上发布其任务,描述任务的操作序列与约束,如第 2 节中给出的例子:(1)前往 A 使用 SA;(2)前往 D 使用 SD;(3)前往 G 使用 SG;(4)前往 I.随后,服务器针对该任务需求,基于候选参与者的信息和位置服务信息,计算多阶段的子任务分配方案(具体的分配算法见第 4 节).最后,子任务的分配方案被指派给选中的参与者.

3.3　基于约束空间众包的组成要素

本文所述的空间众包有一系列要素构成,包括位置服务拓扑结构、参与者、任务、子任务分配方案等.

(1)位置服务拓扑结构

$$Topo=\langle L,S,E,LS\rangle,$$

$$L=\{l\mid l=\langle\theta_l,\varphi_l\rangle\},$$

$$\varphi_l\in\{restricted,unrestricted\},$$

$$S=\{s\mid s=\langle desp_s,\delta_s,\psi_s\rangle\},$$

$$\psi_s\in\{restricted,unrestricted\},$$

$$E=\{e\mid e=\langle(i,l_j),d_{l_i,l_j},t_{l_i,l_j}\rangle,l_i,l_j\in L\},$$

$$LS=L\times S,\text{且 } size(LS_s^{-1})=1,$$

$$Route=(n_0,\cdots,n_k),\text{且}(n_i,n_{i+1})\in E \text{ or } (n_i,n_{i+1})\in LS \text{ or } LS_{n_i}^{-1}=LS_{r_{n+1}}^{-1} \text{ or } (LS_{n_i}^{-1},r_{n+1})\in E.$$

位置服务拓扑是表达空间范围内物理位置、线下服务及相互之间关联关系的网状结构.使用 $Topo$ 表示该拓扑结构,主要由四个部分组成,分别是位置的集合 L、线下服务的集合 S、位置间的连接 E 及位置与服务的包含关系 LS.

其中,位置 l 由二元组描述.θ_l 是该位置的具体坐标点,φ_l 则描述该位置是否附加有访问权限.若位置仅供授权人员访问,则 φ_l 取值 $restricted$,反之则取值 $unrestricted$.线下服务 s 定义为一个三元组.$desp_s$ 是服务的简要描述信息.δ_s 表示该服务预估的执行时长.ψ_s 指示了该服务是否具有使用权限.当任何人都能使用该服务时(有专人提供服务或者任何人都能够使用特定的服务设备),ψ_s 取值 $unrestricted$.反之,当该服务仅允许受限人员使用或者该提供该服务的设备仅能具有使用权限的人员操作时,ψ_s 取值 $restricted$.

E 代表了实际物理空间中位置间的通道,即拓扑网络中连接两个 l(l_i 与 l_j)的边.每一个通道 e 定义为一个三元组,除了连接的两个位置之外,d_{l_i,l_j} 和 t_{l_i,l_j} 分别表示从 l_i 前往 l_j 或从 l_j 前往 l_i 的路线距离

与预估的移动时间.值得注意的是,距离与移动时间不一定成正比,即距离短可能用时会更长,在真实的上下坡现实场景中可能存在这种情况.位置与服务的映射 LS 表示了特定的线下服务所处的位置,这类服务需要参与者真实移动到服务所对应的位置才能使用.在拓扑结构中,假设每一个服务只能对应于一个位置.

拓扑结构中相连的位置或服务构成了路径 $Route$.每一条路径有序地排序了位置结点与服务结点,且相邻的结点必须存在位置的邻接关系或位置与服务的包含关系.为了便于对路径的操作,进一步定义如下的属性与操作:

$Route.startNode$ 与 $Route.endNode$ 表示路径的起点与终点;$Route.T$ 与 $Route.D$ 分别表示沿路径移动的整体用时与距离.$Route.subRoute(l_i,l_j)$ 表示从原始路上截取由 l_i 至 l_j 的子路径的操作.

(2)参与者

$w=LP_w,SP_w,l_w,MR_w,AT_w$,

$LP_w=\{l|l\in L,l.\varphi_l=restricted\}$,

$SP_w=\{s|s\in S,s.\psi_s=restricted\}$,

$MR_w=\{l|l\in L,l$ 包含于移动范围内$\}$,

$AT_w=startTime,endTime$.

每一个参与者 w 定义为一个五元组.其中,LP_w 与 SP_w 分别表示一个参与者针对受限的位置与服务的授权情况,即 LP_w 是该人员除了公共位置外能够访问的受限位置的集合,SP_w 是他能够使用的受限服务的集合.

l_w 表示参与者的实时位置.另外,根据参与者在准备接受任务时指定的移动范围与可用时间,MR_w 是移动范围内的位置集合.AT_w 是愿意承担任务的时间区间,由起始时间与结束时间的二元组描述.

(3)任务

$task=OP,t_{\text{pub}}$,

$OP=(op_1,\cdots,op_n),op_i\in\{move(l_i,l_j),utilize(l,s)\}$, $l_i,l_j,l\in L,s\in S$.

任务请求者在众包平台上发布任务.通过将任务描述映射到位置服务拓扑,将任务 $task$ 定义为一个由操作序列 OP 和发布时间 t_{pub} 构成的二元组.每一项操作 op 具有明确的目标,需要委托参与者从位置 l_i 前往指定的位置 $l_j(move)$ 或使用处于位置 l 的特定的服务 $s(utilize)$.

(4)子任务分配方案

$TA=\{stage|stage=\langle w,Instruction,t_{\text{advice}}\rangle\}$,

$Ins=(i_1,\cdots,i_n),i_k\in\{move(l_i,l_j),utilize(l,s)\},l_i,l_j,l\in L,s\in S,(l_i,l_j)\in E,(l,s)\in LS$.

众包平台根据申请者提交的任务计算得到一组子任务分配方案 TA.TA 由多个阶段组成,每一个阶段指派给一个特定参与者 w,表现为基于位置服务拓扑的有序执行指令 Ins.指令包括位置移动与服务使用,其中位置移动指令详细地指示了从一个位置移动到另一个位置应该经过的位置点.邻接阶段的连接点表示了参与者的交接位置,一般表现为由前一阶段的参与者将物品交给后一阶段的参与者从而继续后续的任务流程.另外,为了尽量缩短任务的完成时间以及减少参与者在交接时的等待时间,每一个阶段还指明了参与者应到达任务起始位置的建议时间 t_{advice}.

除此之外,定义名为 $deriveStage(r,s,d,w,st)$ 的操作,将指派给参与者 w 的在路径 r 中从 s 结点到 d 结点的操作序列转换成子任务分配方案.同时,根据路径 r 的起始执行时间 st,计算在 s 处的建议开始时间,即阶段方案中的 t_{advice}.

(5)优化目标

$Optimal_{\text{dist}}=\min(md(w_1,l_0))+\min(sum(d_{l_1,l_2}))$, l_1,l_2 是 Ins 中涉及的位置

$Optimal_{\text{time}}=\min(md(w_1,l_0))+\min(sum(t_{l_1,l_2})+sum(\delta_s))$, l_1、l_2、s 是 Ins 中涉及的位置和服务

$optimal_{\text{dist}}$ 表示移动距离最短的优化目标.该目标要求分配方案中沿路径的移动距离最短,同时要求第一阶段子任务的参与者到达任务起点的移动距离最短.$optimal_{\text{time}}$ 则表示用时最少的优化目标.它要求分配方案沿路径具有最短的移动用时,同时要求第一个参与者到达任务起点的用时最少.

4 多阶段任务分配算法

本文提出 $MultiStageAllocation$ 算法根据指定的优化目标对基于约束的空间众包任务推导得到子任务分配方案.算法的基本思路是:首先根据优化目标获取与任务对应的路径集合并根据优先级进行排序,随后依次针对每一条路径,递归地采用从起点和终点双向选取局部最优结果的方法选择承担阶段性任务的参与者.一旦路径上的任务能够被一个或一组参与者协作完成,则得到阶段任务的分配方案.

算法 1. $MultiStageAllocation$

输入:$Topo$ //位置-服务拓扑结构

$W=(w_1,\cdots,w_n)$ //候选参与者

$OptimalGoal$ //优化目标

task　　//任务
输出：*TA*　　//子任务分配方案
1. *Routes*←*GetRoutes*(*Topo*,*task*,*OptimalGoal*);
2. *TA*←{};
3. FOREACH *r* in *Routes* DO
4. *TA*←*RouteAllocation*(*r*,*W*,*task*.t_{pub},*TA*);
5. IF *TA*!={} THEN
6. RETURN *TA*;
7. END IF
8. END FOR
9. RETURN {};

MultiStageAllocation 依赖四项输入，分别是位置服务拓扑结构 *Topo*、参与者集合 *W*、任务 *task* 和优化目标 *OptimalGoal*. *MultiStageAllocation* 的执行过程分为两个阶段. 首先，根据优化目标获取与任务对应的且经过优先级排序的路径集合 *Routes*（第 1 行）. 随后，依次针对每一条路径，根据参与者的信息计算在该路径上的子任务分配方案（第 3～9 行）. 一旦得到方案即将方案返回，否则迭代计算优先级次低的路径. 若针对所有路径都无法计算出分配方案，则该次众包任务分配被视为失败.

以动机案例 2 中所描述的任务为例，*MultiStageAllocation* 将针对经过排序的路径集合 *Routes* 计算方案. 在以用时最少为优化目标的情况下，将首先对路径 A-SA-B-D-SD-B-E-F-G-SG-I 进行计算（该路径是 *Routes* 中优先级最高的，即路径总用时最短的）. 若对该路径计算失败，则继续对优先级较低的路径进行计算. *MultiStageAllocation* 依赖 *GetRoutes* 算法获取与任务对应的路径并对其进行排序. 该算法接受位置服务拓扑结构 *Topo*、优化目标 *OptimalGoal* 与任务 *task* 作为输入. 根据 *task* 操作列表 *OP* 中的每项操作 *op*，通过 *findPath* 方法在 *Topo* 中获取覆盖 *op* 起点和终点位置的局部路径集合 *op_routes*（第 1～3 行）. 随后，将各条局部路径集合按照前后顺序进行组合，生成总体任务的路径集合 *Routes*（第 4 行）. 最后，根据不同的优化目标，计算每条路径的总额外开销（总额外移动距离或总额外移动时间），由低到高将路径进行排序（第 5 行）.

算法 2.　*GetRoutes*.

输入：*Topo*　　//位置服务拓扑结构
OptimalGoal　　//优化目标
task　　//任务
输出：*Routes*　　//任务执行路径集合
1. FOREACH *op* in *task*.*OP* DO
2. *op_routes*[*op*]←*findPath*(*Topo*,*op*);
3. END FOR
4. *Routes*←*combine*(*op_routes*[]);
5. *Routes*←根据不同的优化目标，按照额外开销由低到高排序;
6. RETURN *Routes*;

以动机案例 2 中的任务为例，当前任务操作列表为（1）前往 A 使用 SA；（2）前往 D 使用 SD；（3）前往 G 使用 SG；（4）前往 I. *findPath* 方法将计算出所有上述任务操作列表的可能的局部路径，如符合操作（2）的局部路径：B-D-SD 或 B-C-D-SD. 然后对所有操作对应的局部路径进行组合以生成路径集合 *Routes*.

针对每一条路径，*MultiStageAllocation* 算法依赖 *RouteAllocation* 算法计算潜在的子任务分配集合. 算法输入是路径 *r*、参与者集合 *W* 与路径执行的起始时间 *startTime*. *RouteAllocation* 的基本过程如算法 3 所示.

算法 3.　*RouteAllocation*.

输入：*r*　//当前任务执行路径
W　//参与者集合
M　//初始分配结果
输出：*M*　//单路径任务分配结果
1. *s*←*r*.*startNode*; *d*←*r*.*endNode*;
2. *SR*←{sr_1,…,sr_n}, $sr_i \in W$ 且 sr_i 具有起点访问权限、符合 sr_i 移动范围与可用时间约束;
3. IF *SR*=∅ THEN
4. *M*←{}; RETURN *M*;
5. END IF
6. $w_1 \leftarrow w \in SR$, 根据优化目标，使 *forward*(*r*,*s*,*w*).*D-md*(*w*,*s*)或 *forward*(*r*,*s*,*w*).*T-mt*(*w*,*s*)取得最大值;
7. l_1←*forward*(*r*,*s*,w_1).*getEnd*();
8. IF *r*=完整任务路径 THEN
9. *st*←*st*+*movingTime*(w_1,*s*);
10. END IF
11. *updateCoordinate*(w_1,l_1);
12. *DR*←{dr_1,…,dr_n}, $dr_i \in W$ 且 dr_i 具有终点访问权限、符合 dr_i 移动范围与可用时间约束;
13. IF *DR*=∅ THEN
14. *M*←{}; RETURN *M*;
15. END IF
16. $w_2 \leftarrow w \in DR$, 根据目标，使 *backward*(*r*,*d*,*w*).*D-md*(*w*,*d*.*bw*)或 *backward*(*r*,*d*,*w*).*T-mt*(*w*,*d*.*bw*)取得最大值;
17. l_2←*backward*(*r*,*d*,w_2).*getStart*();
18. IF *r*.*subRoute*(*s*,l_1)∩*r*.*subRoute*(l_2,*d*)≠∅ THEN
19. l_j←交叠位置中 w_2 额外开销最小的位置;

20. $M \leftarrow M + deriveStage(r,s,l_j,w_1,st) + deriveStage(r,l_j,d,w_2,st)$; RETURN M;
21. ELSE
22. $M \leftarrow M + deriveStage(r,s,l_1,w_1,st) + deriveStage(r,l_2,d,w_2,st)$;
23. $r' \leftarrow r.subRoute(l_1,l_2)$;
24. $RouteAllocation(r',W,st+r.subRoute(s,l_1).T,M)$;
25. END IF

首先采用从起点递推的方式选择参与者.找出具有起点位置访问权限且参与者的移动范围与可用时间覆盖该起点的位置与执行时间的参与者集合 SR(第 2 行).若没有参与者满足条件,则算法返回空集合(第 3～5 行).反之,根据不同的任务优化目标,找出最合适的参与者(第 6～7 行).在该步骤中,$forward(r,s,w)$方法用于计算 w 在 r 路径中从 s 位置处出发到他最远能够访问到的位置点或服务的子路径.计算得到的子路径需要满足以下条件:(1)w 具有对子路径中所有位置和所有服务的访问授权;(2)子路径中的所有位置点都在 w 的移动范围内;(3)子路径的执行时间在 w 的可用时间范围之内.

值得注意的是,如果此时出现两个或多个特征($w=\langle LP_w,SP_w,l_w,MR_w,AT_w\rangle$)完全一致的参与者,算法将会引入参与者的信誉度(credit)指标来协助选择.信誉度是由众包平台提供的用户数据,由用户的任务完成率、任务转包率、历史承接任务数量等数据综合推算得出,当出现两个或多个特征完全一致的参与者时,算法将会选择信誉度最高的参与者进行任务分配.

针对用时最短的优化目标,从 SR 中选择沿路径前进耗时最长(即能沿路径前进最远)且离开路径起点具有最短移动时间(通过 mt 方法计算)的参与者 w_1(第 6 行).类似地,针对距离最短的优化目标,选择能够沿路径前进距离最长且离开路径起点最近(通过 md 方法计算)的参与者 w_1(第 6 行).算法使用 l_1 表示选中的参与者 w_1 能够达到的最远位置点.

下一步采用从终点反向递推的方式选择协作的参与者.类似地,从 W 中查找所有具有终点位置权限、服务权限且可用时间与移动范围符合终点操作要求的参与者集合 DR(第 12 行).在执行这步骤之前,为了得到更贴近实际场景的结果,在第一次调用 *RouteAllocation* 时需要考虑第一阶段参与者移动至起点的耗时,更新路径执行的起始时间(第 8～10 行).另外,还需要更新第一阶段参与者在执行其阶段子任务后的位置(第 11 行).当 DR 为空时,直接返回空的分配方案.反之,通过 $backward(r,d,w)$ 方法计算 w 在 r 路径中从 d 位置倒推到他最远能够到达的位置点的子路径,并使用 $l.bw$ 表示这条子路径的起点.针对不同的优化目标,计算在子路径中移动的总体耗时/总体距离与 w 前往 $l.bw$ 的耗时或距离差值,将具有最大差值的 w 选定为承担后续阶段的参与者 w_2(第 16 行).

在推导出前后两阶段的子路径后,若两条路径具有交叠,则选择位置交集中对 w_2 而言额外开销最小的位置 l_j(针对用时最少的优化目标,从交集中选择离 w_2 的当前位置移动时间最短的位置.类似地,针对距离最短的优化目标,选择离 w_2 的当前位置移动距离最短的位置)作为交接点,分别创建并返回由 w_1 承担的从 s 到 l_j 以及由 w_2 承担的从 l_j 到 d 的阶段性子任务(第 19～20 行).若两条路径不存在交集,则生成分别由 w_1 和 w_2 承担的从 s 到 l_1 以及 l_2 到 d 的子任务,同时将路径上的剩余部分(从 l_1 到 l_2)视为新的待分析路径 r',递归调用 *RouteAllocation* 进行子任务分配(第 22～24 行).其中,r' 路径的执行起始时间为 r 的实际起始执行时间加上 l_1 之前路径的耗时.

以动机案例 2 中的任务为例,针对路径 A-SA-B-D-SD-B-E-F-G-SG-I,*RouteAllocation* 算法经历了两次调用.在第一次调用中,基于从起点递推的方式推导得到应由 Bob 承担从 A 至 D 的子路径(A-SA-B-D),基于从终点反向递推的方式得到应由 David 承担从 E 至 I 的子路径(E-F-G-SG-I).第二次调用针对剩余的从 SD 至 E 的路径(SD-B),可直接推导得到由 Charlie 单独承担.将这些结果合并后的得到三阶段的子任务分配方案.类似地,在以距离最短为优化目标的情况下,也通过调用两次 *RouteAllocation* 过程得到针对距离最短的路径的三阶段任务分配方案.

多阶段任务分配算法的最佳时间复杂度为 $O(n\times\log n\times l^2)$,最坏时间复杂度为 $O(r\times n\times\log n\times l^2)$,其中 n 为报名参加某任务的参与者的平均数量,l 为任务涉及的位置或服务的平均数量,r 为任务路径集合 *Routes* 的平均长度.

RouteAllocation 采用了从起点和终点双向选取局部最优结果的设计方式和从起点开始单向选取局部最优结果的设计方式(如算法 4 所示)相比,在两者算法耗时相近的情况下,双向算法能取得一定的性能优势,即在总额外移动时间或总额外移动距

离方面略优于单向算法；此外，双向算法在从起点递推和从终点反向递推过程中出现交叠部分的时候选择交叠部分中造成最小额外开销的位置或服务. 同时，双向算法能兼顾某些特殊情况，如某个参与者满足路径中除了起点以外大部分位置或服务的限制条件的情况. 在第 5 节实验与讨论中，将结合实验数据对双向算法和单向算法的性能优劣问题作进一步讨论.

算法 4.　*RouteAllocationFromStart*.

输入：r //当前任务执行路径
　　W //参与者集合
　　st //路径执行的起始时间
　　M //初始分配结果
输出：M //单路径任务分配结果

1. $s \leftarrow r.startNode$；$d \leftarrow r.endNode$；
2. $SR \leftarrow \{sr_1, \cdots, sr_n\}$，$sr_i \in W$ 且 sr_i 具有起点访问权限、符合 sr_i 移动范围与可用时间约束；
3. IF $SR = \varnothing$ THEN
4. 　$M \leftarrow \{\}$；RETURN M；
5. END IF
6. $w_1 \leftarrow w \in SR$，根据优化目标，使 $forward(r,s,w).D\text{-}md(w,s)$ 取得最大值；
7. $l_1 \leftarrow forward(r,s,w_1).getEnd()$；
8. IF r=完整任务路径 THEN
9. 　$st \leftarrow st + movingTime(w_1, s)$；
10. END IF
11. $updateCoordinate(w_1, l_1)$；
12. $M \leftarrow M + deriveStage(r, s, l_1, w_1, st)$；
13. IF $l_1 = d$ THEN
14. 　RETURN M；
15. ELSE
16. 　$r' \leftarrow r.subRoute(l_1, d)$；
17. 　$RouteAllocation(r', W, st + r.subRoute(s, l_1).T, M)$；
18. END IF

5　实验与讨论

本节展示实验的过程与结果. 实验分为模拟实验和真实场景实验两部分. 模拟实验的目的是验证算法的有效性与可拓展性，真实场景实验的目的是验证算法的可用性. 首先介绍实验设计方案，包括模拟数据的生成方式、真实场景实验的设置等细节. 其次列举实验结果并进行评价. 最后讨论本方法的特点与局限性.

5.1　模拟实验

5.1.1　实验设计

为了比对并验证多阶段任务分配算法的可拓展性与有效性，我们另外设计了一个基于动态规划的算法作为对比对象. 该算法与本文的多阶段任务分配算法共享 *MultiStageAllocation* 与 *GetRoutes* 部分（即算法 1、2），在计算子任务分配集合方面（即算法 3），采用一种基于动态规划的全局最优算法，如算法 5 所示.

算法 5.　*OptimumAllocation*.

输入：r //任务执行路径
　　W //参与者集合
　　st //路径执行的起始时间
输出：M //任务分配结果

1. $len \leftarrow r$ 中位置和服务的总数；
2. $allocation[] \leftarrow$ 长度为 $len+1$ 的数组，其中 $allocation[k]$ 表示 r 中前 k 个位置或服务组成的局部路径的最佳分配方案（$k>0$）；
3. FOR $i \leftarrow 1$ TO len
4. 　$extraCost \leftarrow$ 根据优化目标得到的总额外开销；
5. 　$winner \leftarrow 0$；
6. 　FOR $j \leftarrow 1$ TO i
7. 　　$currCost \leftarrow allocation.cost() + minCost(w, j, i)$；
8. 　　IF $currCost < extraCost$ THEN
9. 　　$winner \leftarrow j$；
10. 　　$extraCost \leftarrow currCost$；
11. 　　END IF
12. 　END FOR
13. 　$allocation[i] \leftarrow$ 在 $allocation[winner]$ 的基础之上，将 $winner \sim i$ 段分配给最低开销的参与者；
14. END FOR
15. RETURN $allocation[len]$

OptimumAllocation 的基本原理是：利用一个数组保存前 n 个位置或服务组成的局部路径的最佳分配方案（第 2 行）. 对于其中某个局部路径，假设其为前 i 个位置或服务组成的局部路径，其中 $0 < i < k$（第 3 行），考虑其每一个可能的子局部路径（前 j 个位置或服务组成的局部路径，$0 < j < i$）与活跃参与者中在满足权限授权情况、活动范围和可用时间等约束条件的限制下，能够单独完成路径缺失部分（第 $j+1$ 个到第 i 个位置或服务）的参与者. 类似地，若此时出现两个或多个特征（$w = LP_w, SP_w, l_w, MR_w, AT_w$）完全一致的参与者，算法将会参考参与者的信誉度指标来完成选择. 找出所有的子局部路径与缺失部分单参与者填补的总额外开销（根据不同的优化目标决定）最小的组合，作为前 i 个位置或服务组成的局部路径的新方案（第 6～13 行）. 最后，输出前 n 个位置或服务组成的局部路径的分配方案. *OptimumAllocation* 的最佳复杂度为 $O(n^2 \times l^2)$，最坏时间复杂度为 $O(r \times n^2 \times l^2)$，其中 n 为报

名参加某任务的参与者的平均数量，l 为任务涉及的位置或服务的平均数量，r 为任务路径集合 *Routes* 的平均长度.

在此基础上，模拟实验的目标设置为，在相同的位置服务拓扑与参与者约束情况下：

(1)验证多阶段任务分配算法相比基于动态规划的 *OptimumAllocation* 具有更高的计算效率.

(2)验证如下结论：从起点和终点双向选取局部最优结果的设计方式和从起点开始单向选取局部最优结果的设计方式相比，具有一定的优越性.

在给定的位置服务拓扑结构的前提下，实验所用的模拟数据是随机生成的参与者的受限位置与服务的授权情况、活动范围和当前位置坐标等数据. 这组数据的随机生成方式遵照以下规则：

(1)针对参与者在受限位置与服务上的授权，对每个受限位置和服务，随机生成所有参与者的受限位置与服务的授权情况的数据.

(2)针对参与者的当前位置，首先根据位置服务拓扑结构得出所有位置的具体坐标数据和整体空间范围的横纵坐标区间，然后在该区间内随机生成每个参与者的当前位置的横纵坐标.

(3)针对参与者的移动范围，在一定区间内随机生成每个参与者的活动半径，根据参与者的当前位置和活动半径，计算得出参与者能够访问的位置集合.

在本次实验中，生成多组参与者的模拟数据. 具体生成方式如下：

(1)根据图 1 所示的位置服务拓扑生成对应的模拟平面图. 图 1 所示拓扑参考了某公司办公地点的实际拓扑结构，因此具有其合理性. 模拟的平面拓扑为长 1200 m、宽 850 m 的空间范围，并标注出每个位置或服务在该空间内的坐标. 同时生成每个位置或服务的权限概率值，该值表示对于某个参与者，有多大的概率拥有该位置或服务的访问权限. 一般而言，公开程度越高的空间或服务，其权限概率值越高，如茶水间、休息室的权限概率值均较高. 权限概率值的生成过程参考了该公司管理平台公开的人员分布真实数据：将所有人员分为高层管理人员、基层管理人员、技术类普通员工、非技术类普通员工、保洁安保人员和外部访客等不同分类，并从企业管理平台获得每个分类的数量信息. 其中外部访客分类采取 1 个月内的日平均访客数量. 上述数据均来源于某真实公司. 根据每个位置或服务对不同分类人员的开放程度，计算其权限概率值. 例如，文件盖章服务仅对高层管理人员、基层管理人员和非技术人员开放，这三类人员占总人员数量的 55%，则文件盖章服务的权限概率值为 55%.

(2)定义活跃参与者为接受任务指派的参与者. 根据不同的活跃参与者数量分别使用两种算法记录算法的执行时间(*ExecutionTime*)与分配结果. 例如，活跃参与者数量为 20000，这代表有20000 名参与者报名参与了该众包任务，可以作为任务分配对象. 活跃参与者数量选取方式为：从 20000 开始，以 20000 为增长梯度，依次递增至 160000.

(3)针对某个具体的活跃参与者数量，依据上述的三类随机数据生成规则，生成对应数量的随机模拟数据. 现在考虑每个生成的模拟参与者，针对每个位置或服务均生成一个 0 至 1 范围内的随机数，根据该位置或服务的权限概率值，决定这个模拟参与者是否拥有对应权限. 模拟参与者的当前位置为在平面拓扑范围内随机生成的坐标点. 模拟参与者的活动半径为 100 m 到 600 m 内的随机值，根据随机活动半径计算出其意愿活动范围内的位置或服务集合.

下面定义三个任务，分别评价多阶段任务分配算法与基于动态规划算法的分配结果. 这三个任务都基于图 1 所示的位置服务拓扑.

*task*1 的操作序列描述：前往会议室领取某文件(C)；前往图书馆(F)使用数据归档服务(SF)；将归档凭证递送至计算机楼(I).

*task*2 的操作序列描述(即动机案例中的任务)：前往收发室(A)使用物品领取服务(SA)；前往打印室(D)使用文件打印服务(SD)；前往财务处(G)使用文件盖章服务(SG)；前往办公室(I).

*task*3 的操作序列描述：前往收发室(A)使用物品领取服务(SA)；前往打印室(D)使用文件打印服务(SD)；前往财务处(G)使用文件盖章服务(SG)；前往归档处(F)使用数据归档服务(SF)；前往办公室(I)；前往打印室使用文件打印服务(SD)；前往会议室(C).

任务设计遵循了任务规模递增的原则，从 *task*1 到 *task*3，任务涉及的空间位置和服务的数量增加，任务描述的复杂程度增加，平均任务路径长度增加.

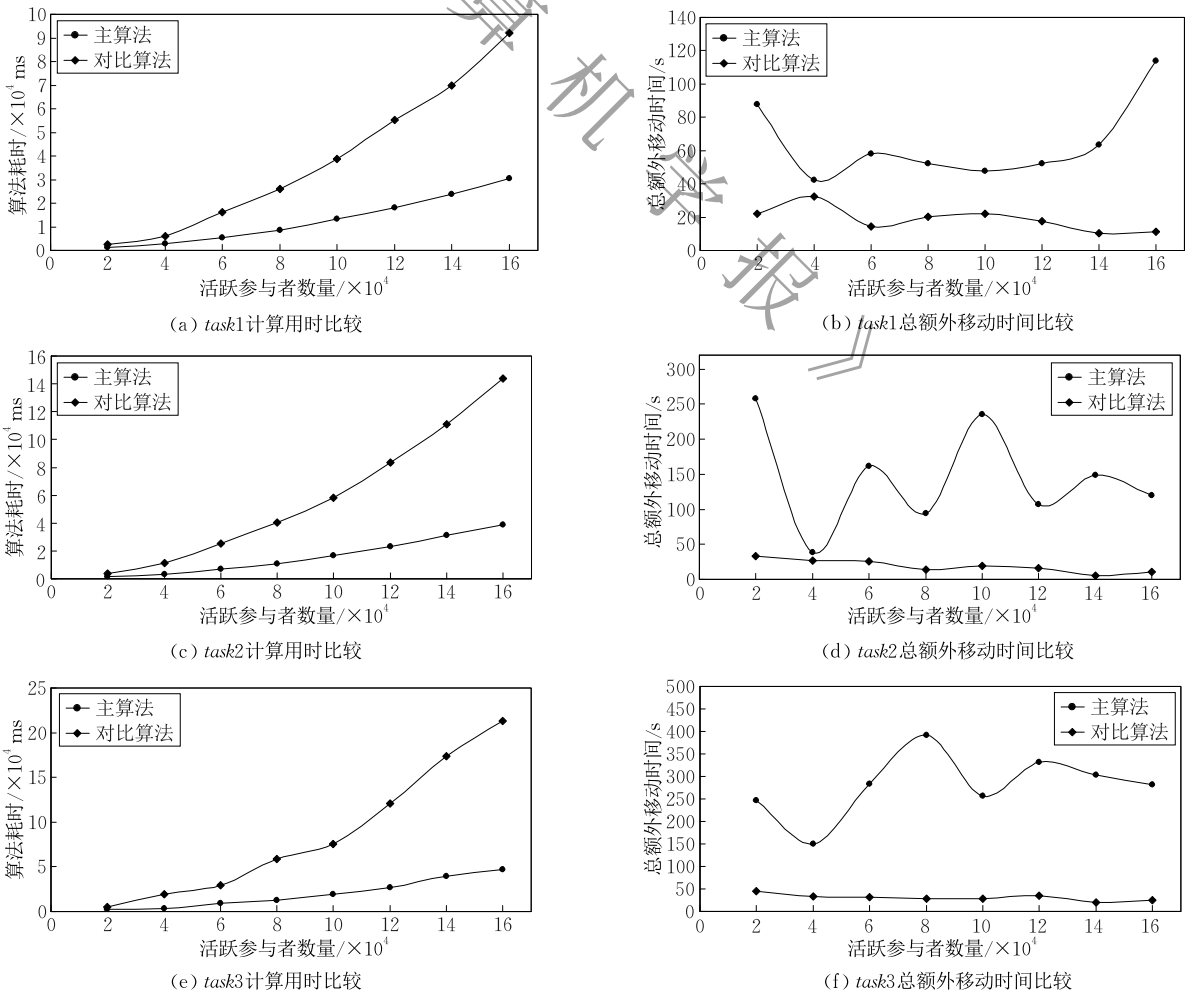
对于上述三类定义任务，根据不同的优化目标分别执行分配 *MultiStageAllocation* 算法和 *OptimumAllocation* 算法，得出任务分配结果. 其中 *MultiStageAllocation* 为本文提出的主算法，而 *OptimumAllocation* 为对比算法. 记录任务分配方

案结果的总额外移动距离(*TEDist*)和总额外移动时间(*TETime*)，分别表示所有参与者为了完成任务而从当前位置移动到阶段任务起点所花费的额外移动距离和移动时间. 在每组任务中，采用三次实验取平均值的方式得出算法的计算耗时和性能指标数据. 类似地，针对 *task*2，根据不同的优化目标分别执行双向算法和单向算法，得出任务分配结果，记录 *TEDist* 和 *TETime* 数据，以便验证双向算法和单向算法的性能优劣问题. 另外，本次实验的操作环境是 2.6 GHz Intel Core i5，8 GB DDR3.

5.1.2　实验结果与评估

图 5 与图 6 分别列举了以时间最少和以距离最短为优化目标的实验结果. 每张图中包含 6 个统计图表. (a)、(c)和(e)分别是两个算法针对 *task*1、*task*2 和 *task*3 在活跃参与者数量递增时的计算用时比较. (b)、(d)和(f)则表示两个算法在活跃参与者数量递增时针对 *task*1、*task*2 和 *task*3 的总额外移动用时或总额外移动距离方面的比较. 其中，折线是时间或距离的度量.

根据图 5(a)、(c)、(e)、图 6(a)、(c)与(e)的图表结果，首先考察同样的任务规模(即平均任务路径长度)下，两种算法计算耗时变化. 可以看出，在三组相同任务的实验结果图表中，*MultiStageAllocation* 与 *OptimumAllocation* 的计算耗时均呈现出明显的与活跃参与者数量正相关的趋势. 相对而言，*OptimumAllocation* 的增长速度较 *MultiStageAllocation* 快，几乎每张图表中 *MultiStageAllocation* 的计算耗时均低于 *OptimumAllocation*. 其次考虑在不同的任务规模下，相同活跃参与者数量的算法耗时对比. 由于三个任务的任务规模呈递增趋势，即平均任务路径长度从 *task*1 到 *task*3 依次增加，可以看出，*task*2 的每组活跃参与者数量所造成的计算耗时均高于 *task*1 的对应计算耗时，而低于 *task*3 的对应计算耗时. 第 4 节多阶段任务分配算法部分指出，*MultiStageAllocation* 的平均时间复杂度低于 *OptimumAllocation*，结合实验结果可以初步得出结论：基于

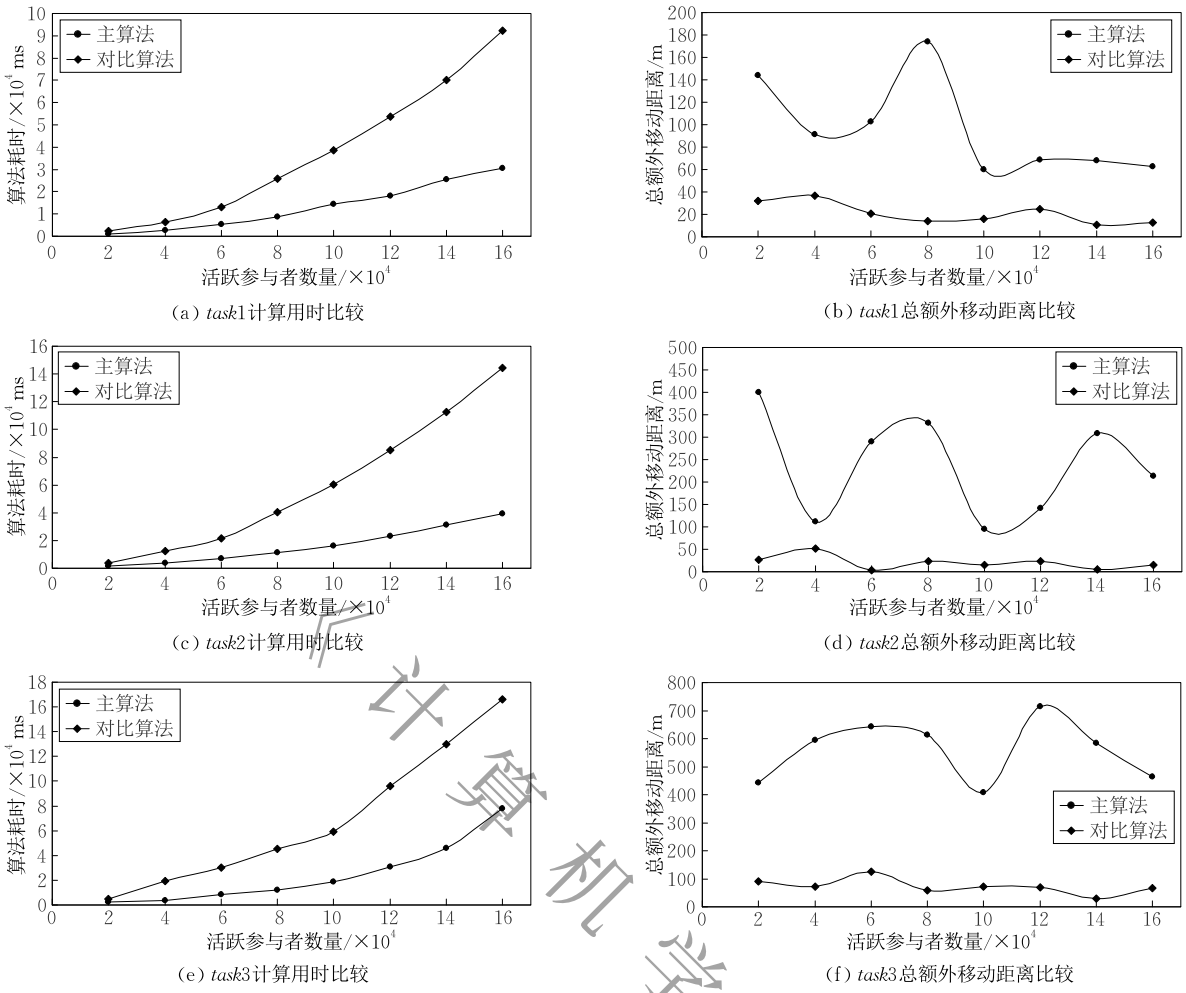


(a) *task*1计算用时比较　(b) *task*1总额外移动时间比较

(c) *task*2计算用时比较　(d) *task*2总额外移动时间比较

(e) *task*3计算用时比较　(f) *task*3总额外移动时间比较

图 5　时间最短优化目标下的实验结果



(a) *task*1计算用时比较　(b) *task*1总额外移动距离比较

(c) *task*2计算用时比较　(d) *task*2总额外移动距离比较

(e) *task*3计算用时比较　(f) *task*3总额外移动距离比较

图 6　距离最短优化目标下的实验结果

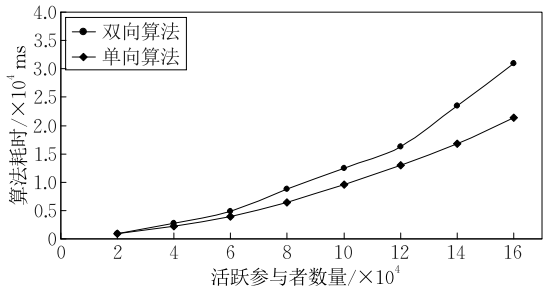
约束的空间众包多阶段任务分配算法在计算效率上要优于基于动态规划的对比算法.

在比较用时最少目标下的总额外移动时间方面(图 5(b)、(d)和(f)),*OptimumAllocation* 变化趋势比较稳定,集中在一个较小的区间内,而 *MultiStageAllocation* 的浮动范围较大. 同时,*OptimumAllocation* 的总额外移动时间优于 *MultiStageAllocation*. 考察总额外移动用时的增加率,最低约为 20%,平均保持在 200% 的水平. *OptimumAllocation* 得出的是全局最优额外移动时间,在活动参与者数量达到非常大规模的情况下将收缩至非常小的数值,而面向局部最优的贪心算法的额外开销相对较高. 相比于最优分配结果,本文所提算法推导得到的分配方案在计算效率方面具有优势. 类似地,在比较距离最短目标下的总额外移动距离方面(图 6(b)、(d)和(f)),两个算法也体现出类似的特征. *MultiStageAllocation* 得到的额外移动距离比算法二高,但计算耗时较低.

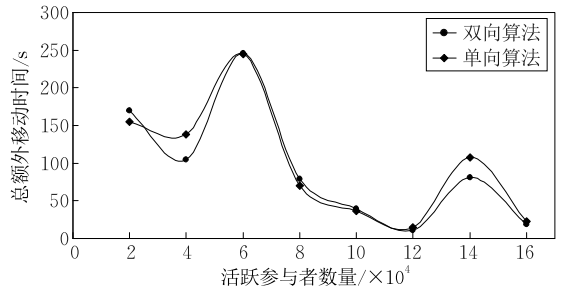
产生上述实验结果的主因是两种算法在机制上的差异. *MultiStageAllocation* 是一种基于贪心机制的多阶段任务分配算法,其基本思路是:从起点和终点开始,双向地对任务路径进行分解,在每一条局部路径的任务承担者选择方面均采用当前状态下最优的选择(造成的社会资源额外开销最低,即文中提出的两种优化目标中的一种达到最低),以得到最终的分配方案. 相对而言,*OptimumAllocation* 基于动态规划,追求全局最优解. 因此,*MultiStageAllocation* 在计算耗时方面会优于基于动态规划的 *OptimumAllocation*. 但是必然地,*MultiStageAllocation* 得出的分配方案不一定是全局最优的,很有可能得到的是近似最优的结果.

此外,图 7 的图表结果对从起点和终点双向选取局部最优结果的设计方式和从起点开始单向选取局部最优结果的设计方式进行对比. 下面我们称前者为“双向算法”,称后者为“单向算法”.

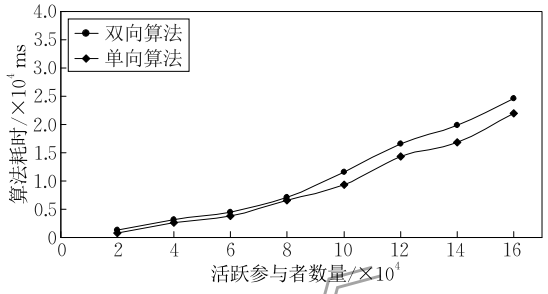
从图 7(a)、(c)的结果可以看出,双向算法与单向算法在时间最短目标和距离最短目标下都呈现出相似的增长趋势,相较而言,双向算法的算法耗时略



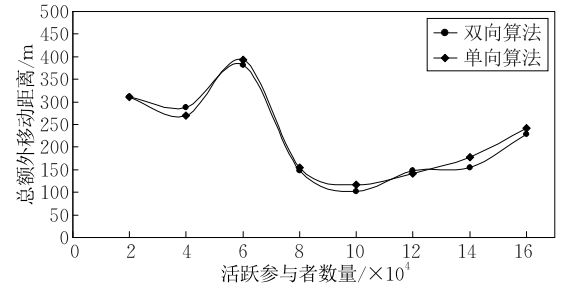
(a) 时间最短目标下计算用时比较



(b) 时间最短目标下总额外移动时间比较



(c) 距离最短目标下计算用时比较



(d) 距离最短目标下总额外移动距离比较

图 7　双向与单向算法对比实验结果

高于单向算法. 而从图 7(b)、(d)的结果可以看出，双向算法在特定优化目标下的额外开销略低于单向算法.

采用双向算法的主要优势在于：(1) 在两者算法耗时相近的情况下，双向算法能取得一定的性能优势，即在总额外移动时间或总额外移动距离方面略优于单向算法；(2) 双向算法在从起点递推和从终点反向递推出现交叠部分的时候进行选择交叠部分中造成最小额外开销的位置或服务，具有一定的灵活性；(3) 双向算法能兼顾某些特殊情况，如某个参与者满足路径中除了起点以外大部分位置或服务的限制条件，这种特殊情况会被单向算法忽略；(4) 双向算法的可拓展性强，在未来可以改写成多线程版本，进一步提高性能.

5.2　真实场景实验

5.2.1　实验设计

本节采用真实场景实验的方式验证算法的可用性. 真实场景实验基于我们自主开发的校园空间众包任务平台 CrowdFrame 进行. CrowdFrame 采用典型的 C/S 软件架构实现，其中服务端通过 HTTP 协议接受请求以及使用 MySQL 关系型数据库管理众包任务数据，客户端使用安卓原生 APP 作为用户入口，并结合百度地图 API 支持任务定位、地图导航、路线规划、附近任务查找等功能. 该平台主要功能包括：空间众包任务发布、浏览、接受、执行；已接收众包任务和已发布众包任务的管理. 目前版本的 CrowdFrame 平台支持笔者所在校区范围内的多个空间位置和服务，具体见 5.2.2 节实验结果与评估部分. 需要指出的是，CrowdFrame 的用户兼具众包任务发布者和参与者两种角色. 在本次实验中，CrowdFrame 的任务分配机制运用了本文提出的多阶段任务分配算法. 任务发布者需要在发布时遵循客户端给出的任务发布模板，描述该任务的各个步骤，给出该任务的步骤初步划分情况，以便被后台映射成 2.2 节中所描述的操作序列，进而执行分配算法. 参与者使用客户端的浏览任务功能查看并决定是否接受任务. 若接受，则通过客户端向平台发送参与者的当前位置，并指定个人的移动意愿范围与可用时间范围. 参与者针对各位置或服务的授权情况则为平台已知信息，由参与者在注册平台时填写.

客户端 APP 的界面如图 8 所示，其中各子图分别展示了查阅任务、报名、查看已报名任务以及个人相关信息的界面.

本次实验中，众包平台采用的任务优化目标选择策略如下：参考 2.2 节中对任务优化目标的描述，对于每个在平台上发布的众包任务，优先根据任务实际场景判断是否为时间敏感类型的任务：首先，众包平台从后台获取任务信息，判断该任务的描述部分是否包含发布者建议完成时间(这是一个可选字段)；其次，判断任务涉及的地点/服务是否临近最晚可用时间点. 最晚可用时间点来自于预定义的位置服务拓扑知识集，表示该位置或服务在一天之内最

(a) 任务浏览 (b) 任务报名

(c) 已接受任务列表 (d) 个人界面

图 8 CrowdFrame 安卓 APP 使用界面

晚的保持开放或可用状态的时间点.满足其中任意一项,则判定该任务为时间敏感任务.若该任务为时间敏感型,则采用“用时最少”优化目标;否则采用“移动距离最短”优化目标.

本次实验设置的范围是笔者所在高校的某校区,在持续 10 天的实验周期内邀请校区内学生下载并使用 CrowdFrame 空间众包平台进行众包任务发布和执行的测试.此外,在实验结束后,对部分任务发布者和参与者进行简要访谈,以获取他们对多阶段任务分配方式以及 CrowdFrame 使用的空间众包分配算法的评价.在实验周期内,共有 48 名在校学生参与实验,其中包括 9 名计算机科学专业学生、18 名软件工程专业学生和 21 名其他专业的学生,非计算机专业学生的比例达到 43.8%.

5.2.2 实验结果与评估

在本次实验中,平台发布任务总数为 26 个,其中 18 个任务分配并执行成功,6 个任务分配失败,2 个任务由于参与者在规定期限内未能完成导致任务过期,任务成功的比例为 69.2%.在全部实验参与人员中,有 11 人发布了 1 个任务,6 人发布了 2 个任务,1 人发布了 3 个任务,其余参与者均未发布任务;有 24 人参与了 1 个任务的执行,7 人参与了 2 个任务的执行,2 人参与了 3 个任务的执行,1 人参与了 5 个任务的执行,其余 14 人未参与执行任务.表 1 展示了本次实验中部分具有代表性的任务的分配情况和执行结果.

表 1 任务分配与执行的部分结果

任务内容	阶段	报名人数	分配结果	完成结果
去超市买食品	2	5	2	完成
实验室帮忙带午饭	2	4	1	完成
组织学术讨论会	2	4	3	完成
取快递	3	1	0	失败
操场踢球报名	2	5	4	完成
校车点排除情况查询	1	1	0	失败
826 门口帮拿咖啡	3	2	1	完成
召集人搬服务器	3	3	3	完成

以“去超市买食品”任务为例说明平台分配任务的流程,其中具体参与者的姓名用字母指代:到学校超市购买指定的食品(面包),送至计算机学院的 401 号实验室.平台计算得到的分配结果为:(1)指派学生 A,到学校超市购买指定的食品,并送至计算机学院门口;(2)指派学生 B,到计算机学院门口从 A 手中交接物品,并送至 401 号实验室内.由于学生 A 不是计算机学院的成员,并不具备计算机学院大门的准入权限,但是学生 A 距离学校超市的位置非常近,因此平台最终得出两阶段的分配方案,利用学生 A 和 B 的协作完成该任务.

另一方面,“取快递”任务的流程为:到校园快递点获取指定物品,送至任务发布者的寝室.该任务最终失败的原因在于,校园快递点和任务发布者的寝室的距离相对较远,加上报名该任务的参与者数量较少,平台无法在报名该任务的参与者中计算得出一个符合条件的分配方案.

总体而言,实验的结果能初步证明基于约束的空间众包多阶段任务分配算法在真实的众包任务分配场景中能起到较好的作用,具有一定的可用性.首先,实验参与人员包含了校区范围内不同专业的学生,一定程度上消除了计算机专业知识背景对空间众包平台使用的上手难易程度的影响,提高了被试人员的多样性.选择学生作为众包实验的主体对象,是基于现实角度的考虑,对于校区范围内如表 1 所示的和日常任务关系比较密切的众包任务,使用学生作为众包实验对象能取得较好的效果.其次,从平

台积累的任务历史数据来看，任务成功分配的比例较高，约有 7 成的任务能成功完成；实验人员的参与程度较高，37.5％的用户参与了任务发布过程，70.8％的用户参与了任务执行的过程，能够从一定程度上保证平台以及支持平台运作的任务分配算法的可行性，消除了部分随机因素的影响.

在与用户的访谈过程中，我们基于两个问题获取用户对方法的评价. 首先，我们询问了用户在众包平台使用过程中的体验. 其次，我们向用户了解了他们对众包平台采取的基于约束的多阶段分配算法的评价. 用户反馈的情况进一步证明了算法的可用性. 用户反馈显示，大部分的任务发布者认为，该平台利用空间众包多阶段任务分配算法，支持众包服务的多人次、多任务划分，将分散的资源进行聚集整合，能以较低的成本和较高的效率跨越各类局限与壁垒，来完成发布者提出的任务. 某任务发布者认为："将众包任务分解，综合考虑任务的特点、众包服务的提供者自身特点等属性，规划多阶段众包任务，这一点比较创新，提高了众包的完成的效率同时还解决了空间权限等问题. 但依旧存在一些问题，比如取快递众包任务的多阶段划分比较依赖于服务提供者的高精度的空间位置，若是任务划分出现问题，会极大影响目标人物的完成效率."

同时，任务参与者普遍认为，众包服务对于众包任务承担者来说，众包任务承担者可以有选择性的依据自己的能力、权限、技能等限制条件接受某段任务，对参与者是比较友好的. 但是由于其它阶段任务对众包任务参与者是黑盒的，整个任务的协调性和时效性可能会有所下降. 某任务参与者认为："该 APP 不需要自己挑选任务阶段，简单方便，让我有了更多接任务的欲望. 但是，有时候接收到的任务指令比较奇怪，比如把电动车钥匙带下楼然后把牛奶从电梯带到实验室."总体而言，用户访谈的结果能支持平台的可用性和多阶段任务分配算法的可行性.

5.3　讨　论

多阶段任务分配算法具有较高计算效率的原因在于，该算法采用了贪心机制，在每个阶段迭代地做出局部最优选择. 虽然在许多问题中，基于贪心机制的算法通常不会产生全局最优的解决方案，但是结合 5.1.2 节中的模拟实验数据和 5.2 节中真实场景实验的分析结果，依此法产生的解决方案的性能在可接受的范围内. 相对而言，基于动态规划的算法由于要考虑所有的组合方案，因此其复杂度与活跃参与者的数量和任务中的操作序列长度有关. 通常而言，空间众包任务越复杂，操作序列越长，算法的耗时增长速度越快. 虽然这种算法能够得到更好的子任务分配方案，但是其较长的计算耗时则会影响空间众包任务的实时分配. 综合而言，本文提出的算法在分配结果方面造成的总体社会资源额外损耗会略高，但是其计算性能方面具有一定优势.

本文是对基于约束的空间众包问题的初步研究. 现阶段，本文所提方法建立在对空间拓扑与参与者的一系列假设基础之上，因此当应用于实际场景时仍然具有以下方面的局限性.

(1) 暂未考虑真实世界的复杂空间拓扑结构. 本文定义的位置服务拓扑较为简单，即通过网络图的方式将位置进行连接，并将线下服务关联到位置结点. 任务分配的过程即是在这种网络图的基础上进行路径规划的过程. 然而，真实物理空间具有更为复杂的结构，覆盖一定区域的空间可视为其中的一阶实体，且空间与空间之间还存在嵌套关系. 另一方面，服务与位置的关系也可能需要考虑移动距离的因素. 这些特点为空间众包问题带来了更多空间维度的约束，为众包任务的分配提出了更大的难点.

(2) 暂未充分考虑参与者的主观性. 当前，本文方法主要关注任务的分配，对于参与者而言，我们假设愿意接受任务并被指派任务的参与者能够自愿地、在第一时间开始承担任务. 同时，参与者也不会中途退出任务或在等待其任务阶段前离开分配时定位的位置. 然而，参与者的物理与心理状态是实时变化的，他会时刻移动到不同的位置，另外其承担任务的意愿也会受到激励机制的影响. 在众包任务的分配过程中考虑以上这些因素能够提高分配方案的准确性与合理性.

(3) 为了简化以用时最少为优化目标的任务分配，我们简单地指定了位置服务拓扑结构中位置之间的移动用时，并简化了参与者移动到阶段任务起始位置的移动耗时. 现实场景中，参与者的移动用时依赖于交通工具、路况等因素，因此每次位置间的移动用时都会各不相同. 这个问题为空间众包的阶段性分配，尤其是以时间为优化指标的任务分配带来了难题，需要结合参与者的行为分析或实时路况计算等技术为任务分配提供更实时、有效的信息.

(4) 本文所提方法在本质上是一种离线任务分配算法，即在任务执行之前通过计算得出最佳分配策略. 各阶段任务的执行依赖于分配阶段获取的参与者位置、意愿等信息. 然而，在任务执行过程中，参

与者的位置、移动方式、用时与移动意愿会动态发生变化,这些因素会影响离线任务分配的准确性.

6 相关工作

与本文相关的研究工作主要涉及三个方面:空间众包、空间物理拓扑在空间众包中的应用以及空间众包中的任务分配.

(1)空间众包在2012年由Kazemi等人首次提出[12].在空间众包框架中,参与者向服务器发送地理位置和区域限制信息,服务器根据参与者位置为其分配附近的任务.这类空间众包的主要优化目标是在符合参与者约束的前提下使得任务分配的概率最大化.有些研究者提出并开发了在线的空间众包平台.例如,Chen等人[17]提出了一个通用的空间众包平台gMission,通过制定一套有效的涵盖参与者奖励、任务分配、结果汇总和数据质量控制的解决方案来保证众包平台的质量.gMission包含地理感应、工作人员检测和任务推荐等关键技术.

参与式感知(Participatory Sensing)是当前空间众包领域内的主流方式.参与式感知的众包要求参与者在特定活动中利用其配备传感器的移动设备来收集和共享数据.在这方面的研究工作中,Bulut等人[18]研究了基于位置查询的众包问题,该查询采用基于位置的服务(例如Foursquare)查找适当的参与者回答给定的查询问题.Cornelius等人[19]提出了一个参与式感知框架,用于开发基于移动设备实现协同机会感知的应用.Kazemi等人[20]则提出了另一种感知框架,重点关注用户在参与式感知过程中的隐私安全.参与式感知众包通常应用于小规模任务,难以扩展到基于大规模空间拓扑与约束的任务上.相比于这种类型的空间众包,本文方法所针对的众包专注于物品传递、线下服务使用等形式,通常无法仅靠配备传感器的移动设备完成.

(2)空间物理拓扑是描述空间众包特征的重要手段.Pasquale等人[21]使用网络空间的拓扑结构捕捉网络和数字设备的配置方式,包括包含和连接关系.为了支持网络物理空间的访问控制,他们开发了一种工具,支持对建筑物拓扑特征的可视化和编辑,并验证访问控制策略是否满足对可达性关系施加的安全要求.Tsigkanos等人[22]提出的技术框架对不断发展的网络物理空间进行建模并推导其时空属性.该工作利用离散的、基于Bigraph图的形式来建模网络物理空间以及变化原型,通过基于逻辑的属性规范和模型检查过程实现安全、隐私等方面的形式化推理.Tsigkanos等人[23]还针对可达性分析需求从建筑信息模型(BIM)中自动提取可分析模型,然后在设计时基于静态和动态特性对分析需求进行检查.以上这组工作大多建立在信息物理空间基础上,通过Bigraph图或者建筑信息模型等手段对物理空间进行建模,从而满足形式化验证、可达性推理等一系列目标.与这些工作相比,本文方法依赖于涵盖位置与服务信息的拓扑结构,主要描述众包任务的实施场景,强调位置之间的连接关系以及位置和线下服务的包含关系,这为任务的多阶段分配提供依据.

(3)任务分配是空间众包中的关键技术.空间众包的任务分配模式可以分为两类:参与者选择任务(WST)和服务器分配任务(SAT).在WST模式中,服务器将空间众包任务广播给所有参与者,参与者可以自行决定选取任务.采用WST模式的分配[24-25]允许参与者根据他们的个人喜好选择可用的任务.相反,SAT模式将根据任务和参与者的位置信息将任务直接指派给参与者.例如,Cheng等人[26]提出了一个参与者任务分配策略,通过引入一个有效的基于成本模型的索引,低成本、动态地维护移动的参与者和空间任务,从而使基于空间、时间多样性和任务可靠性的任务分配得分最大化.文献[27]提出的参与者招募框架,选择参与者以最大限度地提高数据收集的空间覆盖率.Cardone等人[28]在进行任务分配时考虑用户智能手机的电池电量,从而选择合适的参与者以最大化任务完成率(例如,时空覆盖率).Zhang等人[29]提出了一个参与者选择框架用于尽量减少参与者激励成本.Li等人[30]提出需要重点关注众包中的质量控制、成本控制和延迟控制三个方面,以解决任务分配数据管理的安全性问题.

在分配算法方面,Cheng等人[31]提出了多技能空间众包(MS-SC),通过找到一个最佳的参与者和任务分配策略,使得参与者和任务之间的技能相互匹配,并在预算约束下达到效益最大化.Hu等人[32]提出了一个由推理模型和任务在线分配算法组合而成的框架,用以解决众包中兴趣点(points of interest)人工标注的问题.Tong等人[33]针对全局在线子任务分配问题,提出了一种在线随机排序模型下的分配算法,以解决离线任务分配算法的缺点.Zheng等人[34]提出了一个任务分配框架,通过历史数据捕捉双方偏好,基于此优化任务分配的结果.另外,Qian等人[35]提出了一种O2O服务组合方法,将离线路径

规划和社会协作相结合从而优化服务选择的策略.

以上这些工作大多建立在简单的空间拓扑基础上计算任务与参与者的分配策略，重点关注任务的分配质量.这些任务一般都可由参与者单独完成.相比而言，本文所提出的方法在任务分配方面考虑更为细节的空间拓扑结构与特点，并结合参与者自身的约束来推导出由多人协作完成的任务分配方案.

7　结　论

相比于传统的空间众包，本文针对的空间众包场景具有更为广泛的约束，包括空间维度上的位置访问权限、线下服务使用权限以及参与者自身的可用时间与移动范围意愿.传统的空间众包技术并未针对这种类型的众包任务提出有效的、可能涉及一组参与者协同合作的任务分配解决方案.

本文提出的基于约束的多阶段任务分配方法旨在解决这类众包问题.首先，从任务、参与者与分配机制这三个维度定义此类空间众包的概念，总结众包的流程并对众包中的要素进行规范化定义.其次，提出了基于约束的多阶段任务分配算法，该算法基于贪心算法的机制，根据优化目标计算任务的分配阶段.最后，设计并实施了模拟实验和真实场景实验，验证了该算法相比于基于动态规划的算法能够在保证分配方案可接受的情况下提高计算效率.

下一步的工作将围绕 5.3 节所列举的方法局限性展开.首先，考虑更符合物理世界现实场景的空间拓扑结构，进一步增强位置服务拓扑的描述能力.其次，考虑参与者的动态特性，研究参与者的行为预测方法和在线任务分配算法以支持任务执行过程中的任务动态调整.最后，开发并完善支持众包的后台服务与移动端应用，在真实的空间场景中执行任务的发布、分配与指派过程，验证方法的实际应用效果.

参　考　文　献

[1] Li Guo-Jie, Xu Zhi-Wei. Judging new economy from perspective of Information technology trend. Bulletin of Chinese Academy of Sciences, 2017, 32(3): 233-238(in Chinese)
(李国杰，徐志伟. 从信息技术的发展态势看新经济. 中国科学院院刊，2017，32(3)：233-238)

[2] Howe J. The rise of crowdsourcing. Wired Magazine, 2006, 14(6): 1-4

[3] Peng X, Gu J, Tan T H, et al. CrowdService: Optimizing mobile crowdsourcing and service composition. ACM Transactions on Internet Technology, 2018, 18(2): 19

[4] Fleenor J W. The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economics, societies and nations. Personnel Psychology, 2006, 59(4): 982-985

[5] Feng Jian-Hong, Li Guo-Liang, Feng Jian-Hua. A survey on crowdsourcing. Chinese Journal of Computers, 2015, 38(9): 1713-1726(in Chinese)
(冯剑红，李国良，冯建华. 众包技术研究综述. 计算机学报，2015，38(9)：1713-1726)

[6] Franklin M J, Kossmann D, Kraska T, et al. CrowdDB: Answering queries with crowdsourcing//Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data. Athens, Greece, 2011: 61-72

[7] Liu Y, Lehdonvirta V, Kleppe M, et al. A crowdsourcing based mobile image translation and knowledge sharing service//Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia. Limassol, Cyprus, 2010: 6

[8] Zogaj S, Bretschneider U, Leimeister J M. Managing crowdsourced software testing: A case study based insight on the challenges of a crowdsourcing intermediary. Journal of Business Economics, 2014, 84(3): 375-405

[9] Kazemi L, Shahabi C. GeoCrowd: Enabling query answering with spatial crowdsourcing//Proceedings of the International Conference on Advances in Geographic Information Systems. Redondo Beach, USA, 2012: 189-198

[10] Chen L, Shahabi C. Spatial Crowdsourcing: Challenges and opportunities. IEEE Data Engineering Bulletin, 2016, 39(4): 14-25

[11] Karaliopoulos M, Telelis O, Koutsopoulos I. User recruitment for mobile crowdsensing over opportunistic networks//Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM). Hong Kong, China, 2015: 2254-2262

[12] Guo B, Liu Y, Wang L, et al. Task allocation in spatial crowdsourcing: Current state and future directions. IEEE Internet of Things Journal, 2018, 5(3): 1749-1764

[13] Xiong H, Zhang D, Chen G, et al. CrowdTasker: Maximizing coverage quality in piggyback crowdsensing under budget constraint//Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications (PerCom). St. Louis, USA, 2015: 55-62

[14] Lane N D, Chon Y, Zhou L, et al. Piggyback CrowdSensing (PCS): Energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities//Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems. Roma, Italy, 2013: 7

[15] Xu L, Hao X, Lane N D, et al. More with less: Lowering user burden in mobile crowdsourcing through compressive sensing//Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing. Osaka, Japan, 2015: 659-670

[16] Deng D, Shahabi C, Zhu L. Task matching and scheduling for multiple workers in spatial crowdsourcing//Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems. Seattle, USA, 2015: 21

[17] Chen Z, Fu R, Zhao Z, et al. gMission: A general spatial crowdsourcing platform. Proceedings of the VLDB Endowment, 2014, 7(13): 1629-1632

[18] Bulut M F, Yilmaz Y S, Demirbas M. Crowdsourcing location-based queries//Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops). Seattle, USA, 2011: 513-518

[19] Cornelius C, Kapadia A, Kotz D, et al. Anonysense: Privacy-aware people-centric sensing//Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services. Breckenridge, USA, 2008: 211-224

[20] Kazemi L, Shahabi C. A privacy-aware framework for participatory sensing. ACM Sigkdd Explorations Newsletter, 2011, 13(1): 43-51

[21] Pasquale L, Ghezzi C, Pasi E, et al. Topology-aware access control of smart spaces. Computer, 2017, 50(7): 54-63

[22] Tsigkanos C, Kehrer T, Ghezzi C. Modeling and verification of evolving cyber-physical spaces//Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. Paderborn, Germany, 2017: 38-48

[23] Tsigkanos C, Kehrer T, Ghezzi C. Architecting dynamic cyber-physical spaces. Computing, 2016, 98(10): 1011-1040

[24] Alt F, Shirazi A S, Schmidt A, et al. Location-based crowdsourcing: Extending crowdsourcing to the real world//Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries. Reykjavik, Iceland, 2010: 13-22

[25] Deng D, Shahabi C, Demiryurek U. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing//Proceedings of the 21st ACM Sigspatial International Conference on Advances in Geographic Information Systems. Orlando, USA, 2013: 324-333

[26] Cheng P, Lian X, Chen Z, et al. Reliable diversity-based spatial crowdsourcing by moving workers. Proceedings of the VLDB Endowment, 2015, 8(10): 1022-1033

[27] Reddy S, Estrin D, Srivastava M. Recruitment framework for participatory sensing data collections//Proceedings of the International Conference on Pervasive Computing. Helsinki, Finland, 2010: 138-155

[28] Cardone G, Foschini L, Bellavista P, et al. Fostering participaction in smart cities: A geo-social crowdsensing platform. IEEE Communications Magazine, 2013, 51(6): 112-119

[29] Zhang D, Xiong H, Wang L, et al. CrowdRecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint//Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing. Seattle, USA, 2014: 703-714

[30] Li G, Wang J, Zheng Y, et al. Crowdsourced data management: A survey. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(9): 2296-2319

[31] Cheng P, Lian X, Chen L, et al. Task assignment on multi-skill oriented spatial crowdsourcing. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(8): 2201-2215

[32] Hu H, Zheng Y, Bao Z, et al. Crowdsourced POI labelling: Location-aware result inference and task assignment//Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE). Helsinki, Finland, 2016: 61-72

[33] Tong Y, She J, Ding B, et al. Online mobile micro-task allocation in spatial crowdsourcing//Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE). Helsinki, Finland, 2016: 49-60

[34] Zheng L, Chen L. Mutual benefit aware task assignment in a bipartite labor market//Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE). Helsinki, Finland, 2016: 73-84

[35] Qian W, Peng X, Sun J, et al. O2O service composition with social collaboration//Proceedings of the 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). Urbana, USA, 2017: 451-461

FAN Ze-Jun, M. S. candidate. His research interests include the fusion of the ternary human-machine-thing, mobile cloud computing software system and edge computing.

SHEN Li-Wei, Ph. D., associate professor. His research interests include the fusion of the ternary human-machine-thing, mobile application development and analysis, etc.

PENG Xin, Ph. D., professor. His research interests include mobile computing and cloud computing, code big data and intelligent software development.

ZHAO Wen-Yun, M. S., professor. His research interests include software engineering, software development tools and environment, enterprise application integration, etc.

Background

This paper studies the problem of task allocation in the field of spatial crowdsourcing. Crowdsourcing is a kind of software service that combines the abilities of different individuals or groups. In terms of concepts, crowdsourcing refers to the act of outsourcing a task traditionally performed by a designated person to a group that is not pre-specified and usually has a large base of personnel in a publicly convened manner. In the field of crowdsourcing, spatial crowdsourcing refers to crowdsourcing for task types related to geographic and temporal elements of the physical world. It requires workers to move to the specified location to perform the corresponding operations, undertake tasks such as data collection and perception, and feedback the operation results through the mobile terminal. The purpose of task allocation is to select one or a group of suitable workers from the candidate workers to undertake the execution of the task, while satisfying certain constraints.

In the field of spatial crowdsourcing task allocation, the problem can usually be solved by finding an optimal worker and task allocation strategy through an algorithm or framework, so that the skills between workers and tasks are mutually matched and benefits under budget constraints are maximized. Most of the existing work is based on a simple spatial topology to calculate the allocation strategy of tasks and workers, focusing on the quality of task allocation. These tasks are generally done by one worker alone. Even in a multitasking scenario, each task is essentially independent, and one worker may undertake one or more of these tasks, but the execution of a single task need to be split and assigned to multiple workers. In contrast, the proposed method in this paper considers more detailed spatial topologies and features in task allocation, and combines the constraints of the workers to derive a task allocation scheme that is coordinated by multiple people. In this paper, in the spatial crowdsourcing scenario with spatial location access, offline service usage permissions and considering the available time and movement range constraints of the workers, the problem that a single worker may not be able to complete the execution of the entire task is studied. The constrained spatial crowdsourcing multi-stage task allocation method efficiently decomposes a task into a set of sub-tasks that can be undertaken by different workers when the constraints are met. Besides, we prove that the method has high computational efficiency and an allocation decision accepted by the workers in the real scene can be obtained.

This work is supported by the National Key Research and Development Program "Cloud Computing Architecture and Platform for the Fusion of the Ternary Human-Machine-Thing" (2018YFB1004800).