基于动态效用的时空众包在线任务分配

余敦辉　　张灵莉*　　付　聪
(湖北大学计算机与信息工程学院　武汉　430062)

摘　要：为提升众包任务在线分配的总体效用，该文提出一种适用于时空众包环境的在线任务分配方法。该方法针对时空众包环境下的在线任务分配问题，首先提出一种以众包任务为中心的 K 最近邻算法来进行候选众包工人的选择，进而设计一种基于动态效用的阈值选择算法，实现众包工人与任务的最优分配。实验结果显示，文中所提出算法具有较好的有效性和可行性，并能在一定程度上保证众包工人的可靠性，优化平台总效益。

关键词：任务分配；时空众包；K 最近邻算法；阈值选择算法
中图分类号：TP393　　**文献标识码**：A　　**文章编号**：1009-5896(2018)07-1699-08
DOI: 10.11999/JEIT170930

Online Task Allocation of Spatial Crowdsourcing Based on Dynamic Utility

YU Dunhui　　ZHANG Lingli　　FU Cong
(*School of Computer Science and Information Engineering*, *Hubei University*, *Wuhan* 430062, *China*)

Abstract: In order to improve the overall effectiveness of the online assignment of crowdsourcing tasks, an online task assignment method is proposed for the space-time crowdsourcing environment. To deal with the problem of online task assignment in spatiotemporal crowdsourcing environment, a K-NearestNeighbor (KNN) algorithm is firstly proposed based on crowdsourcing task to select the candidate crowdsourcing workers. Then a threshold selection algorithm based on dynamic utility is designed to realize the optimal allocation of crowdsourcing workers and tasks. Experimental results show that the proposed algorithm is effective and feasible, and can guarantee the reliability of crowdsourcing workers and optimize the overall efficiency of the platform.

Key words: Task assignment; Spatial crowdsourcing; K-NearestNeighbor (KNN) algorithm; Threshold selection algorithm

1　引言

随着移动互联网时代的到来，众包(crowdsourcing)[1−5]这种通过群体智慧求解问题的新兴商业生产模式应运而生。尤其是近几年，互联网+相关技术与共享经济模式的迅猛发展，以及移动设备的大量普及与广泛应用，更是为众包技术带来了更多的外延需求。传统的在线众包平台逐步演变出了一种新型的服务模式-时空众包(spatiotemporal cmwdsourcing)(也称为空间众包或移动众包)[6]。

在学术界，很多学者对时空众包展开了研究，文献[7]主要对时空众包研究中的任务分配、质量控制与隐私保护问题进行了深入讨论，文献[8]综合考虑第三方工作地点对任务分配的影响和任务分配的动态性，提出了空间众包环境下的 3 类对象在线任务分配问题，文献[9]提出了面向多技能的空间众包任务分配问题，在尽量减少对于工人的支出和保证最大数量的任务分配的同时，实现最佳的任务-工人分配，文献[10]提出了一种将双目标优化与多臂赌博机相结合的空间众包任务动态分配方法，旨在尽量减少行程成本的前提下最大限度地提高任务可靠性。

在现有的时空众包研究中，任务分配问题是其核心问题之一，但现有的任务分配算法存在以下两点不足：(1)目前大部分研究是基于静态场景进行的，但实际应用中，众包任务和工人往往是动态出现的。(2)现有在线任务分配均只考虑到了众包工人和任务，并未结合时空众包应用中时空属性来进行分析和预测。

本文将综合考虑众包工人、众包任务以及其所携带的时空信息等因素，在文献[11]的基础上改进时空众包环境下的在线任务分配问题(TSC-OTA 问题)的算法，使其更好地适用于当下的时空众包平台，

收稿日期：2017-10-09；改回日期：2018-04-08；网络出版：2018-05-09
*通信作者：张灵莉　zll02100210@163.com
基金项目：国家重点基础研究发展计划(2014CB340404)，国家自然科学基金(61373037, 61672387)
Foundation Items: The National Key Basic Research and Department Program of China (2014CB340404), The National Natural Science Foundation of China (61373037, 61672387)

有效提升在线任务分配的效率，更大程度地优化平台总效用。例如，近年来流行的各类实时专车和外卖服务平台，如滴滴出行，Uber，百度外卖等，均采用时空众包方式提供服务，其中，专车用户和点单用户所发出的订单为众包任务，接到订单的专车司机和快送员为众包工人，本文的算法将从平台的角度出发，减小平台开销的同时优化任务分配的总效用，从而为平台提供最优的任务-工人分配结果。

本文的主要贡献如下：(1)本文提出一种基于CT-KNN(Crowdsourcing Task KNN, CT- KNN)查询算法的众包工人选择算法，根据最近一段时间内搜索区域的平均密度对比，选出距离当前众包任务最近的 K 个候选众包工人；(2)本文提出一种基于动态效用的阈值选择算法，分别计算任务成功率期望、任务实际完成时长和工人的单位时间效用，并通过动态效用的对比来完成任务和候选工人的最优分配。

2 基本概念

定义 1 时空众包任务[11]：时空众包任务通常被定义为如下 5 元组的形式，$t=<l_t,r_t,p_t,d_t,f_t>$，其中 l_t 为任务 t 在空间中的位置，r_t 为任务 t 的范围半径，p_t 为任务发布时间，d_t 为任务截止时间，f_t 为完成任务 t 应获得的报酬费用。

定义 2 时空众包工人[11]：时空众包工人是指众包任务的参与者，被定义为如下 6 元组的形式，记为 $w=<l_w,a_w,d_w,r_w,c_w,s_w>$，其中 l_w 为工人 w 的当前的空间位置，a_w 为抵达平台的时间，d_w 为离开平台的时间，r_w 为工人 w 所能接受任务的范围半径，c_w 为 w 的所能完成的最大任务数量，s_w 为工人 w 的历史任务成功率。

定义 3 时空众包环境下的在线任务分配问题(TSC-OTA 问题)[6]：在时空众包平台上，给定一组空间众包任务 T，一组众包工人 W，和效用函数 $U(t,w)$, TSC-OTA 问题的目的是找到一个分配 A 使得任务分配的总效用最大化，即为 $\mathrm{MaxSum}(A)=\sum_{t\in T,w\in W}U(t,w)$，其满足以下约束：(1)期限约束：任务分配结果应在众包任务截止时间 d_t 之前完成；(2)容量约束：每个任务 t 在 A 中最多出现 1 次，每个工人 w 在 A 中最多出现 c_w 次；(3)不变性约束：任务分配一旦给出，则不能改变；(4)空间约束：任务分配 A 需要满足任务 t 在以 l_w 为中心、r_w 为半径的范围内。

定义 4 众包任务完成期：众包任务完成期代表众包工人接到平台所分配的任务到任务结束这段时间间隔，定义为 $a=e_t-b_t$，其中，b_t 为平台预估众包工人接到平台所分配的任务的时间，即为预估任务开始时间，e_t 为预估任务结束时间。

定义 5 众包任务实际完成时长：众包任务实际完成时长代表众包工人完成平台所分配的任务的实际时长，定义为 $z=a\times e$，其中 e 为惩罚因子，表示众包工人的任务成功率期望越低，其惩罚因子越大，完成该任务的实际时间越长。

定义 6 众包效用：众包效用是指众包工人在单位时间内完成众包任务所获得的报酬，定义为 $U(t,w)=\dfrac{f_t\times\bar{s}_w}{z}$，其中 $\bar{s}_w$ 表示带时间权重的任务成功率期望。

3 TSC-OTA 问题的整体实现方案

为解决 TSC-OTA 问题，本文在第 1 步中引入了一种基于 Subsyn 算法的任务预测方法对最终任务的出现进行预测[12,13]，该方法作为未来的研究方向和重点还需不断地优化和改进，故在后文中不作过多赘述。第 2 步构建了一种以众包任务为中心的 CT-KNN 算法，先按照最近一段时间内该位置的历史任务分布情况和历史平均密度确定一个初始区域，然后根据当前的实际密度来调整该区域的范围，最后筛选出距离所选众包任务最近的 K 个候选众包工人。第 3 步提出一种基于动态效用的阈值选择算法，通过时间效应因子权重函数和惩罚因子分别计算任务成功率期望和任务实际完成时长，然后确定众包工人的单位时间效用，最终得到一个最优分配 A 使得众包任务-工人分配的总效用最大化。

4 基于 CT-KNN 查询算法的众包工人选择

KNN 查询(K Nearest-Neighbor query)的基本思想是给定查询对象及正整数 K，从数据库中找出距离查询对象最近的 K 个对象[14–16]。本文提出一种基于 CT-KNN(Crowdsourcing Task KNN, CT-KNN)算法的众包工人选择。该算法采用查询圆方式，首先根据最近一段时间内的历史任务的分布情况确定 K_0，计算该位置最近一段时间内所选择区域的平均密度 $\bar{\rho}$，从而确定初始的搜索区域 R_o，然后计算该区域 R_o 内众包工人的实际密度分布 ρ，最后进一步扩大或缩小查询圆来确定最终包含 K 个最近邻对象的搜索区域 R_f，并这 K 个众包工人按照距离进行升序排序。

4.1 CT-KNN 查询算法

假设 $P=\{p_1,p_2,\cdots,p_N\}$ 为众包工人的集合，$p_i\in P$ (i 为正整数)表示众包工人的唯一标识，$p(x,y)$ 表示某时刻众包工人 p 所在的位置。$Q=\{q_1,q_2,\cdots,q_M\}$ (M 为正整数)为众包任务的集合，假设众包任务在一段时间内是静止的，$q_j\in Q$ (j 为正

整数)表示众包任务的唯一标识，$q(x,y)$ 表示某时刻众包任务 q 所在的位置。算法执行过程如表 1 所示。

表 1　CT-KNN 查询算法

算法 1　CT-KNN 算法

输入：众包任务 q，众包工人 p

输出：q 的 K 近邻

(1)在地图上任选一个众包任务 $q(x,y)$ 作为圆心，根据该位置最近一段时间内的历史任务分布情况确定 K_0，计算该位置最近一段时间内所选择区域的平均密度 $\overline{\rho}$；

(2)以找到 K_0 个众包工人 p 为目标来确定搜索区域 R_o，将其半径记为 r；

(3)根据搜索区域 R_o 计算区域密度 $\rho = K/S_{R_o}$；

(4)若 $\rho = \overline{\rho}$，则 $K = K_0$，初始区域 R_o 即为最终区域 R_f，输出 $\{p_1, p_2, \cdots, p_K\}$ 算法结束；

(5)若 $\rho < \overline{\rho}$, $K_1 = (1-\alpha)K_0 (0<\alpha<1)$，以找到 K_1 个众包工人 p 为目标来确定最终区域 R_f，计算众包任务 q 与 $p_1, p_2, \cdots, p_{k1}$ 的距离并按升序排列，返回前 K_1 个即为众包任务 q 的 K 近邻，算法结束；

(6)若 $\rho > \overline{\rho}$, $K_1 = (1+\beta)K_0 (0<\beta<1)$，以找到 K_1 个众包工人 p 为目标来确定最终区域 R_f，计算众包任务 q 与 $p_1, p_2, \cdots, p_{k1}$ 的距离并按升序排列，返回前 K_1 个即为众包任务 q 的 K 近邻，算法结束。

4.2 算法举例

例：如图 1 所示，当前分区中有查询对象 q(75, 75)、移动对象 $p1$(50,85), $p2$(85,60), $p3$(105,115), $p4$(20,60)。设众包工人以 q 为圆心，根据该位置最近一段时间内的历史任务分布情况可确定 $K_0 = 3$，$\overline{\rho} = 2000\pi/4$，此时初始的搜索区域 R_o 的半径 r_o=50, $\rho = 2500\pi/3$。由于 $\rho > \overline{\rho}$，$K_1 = (1+\beta)K_0 = 4$，算法需要扩大查询圆，直到找到 K_1 个众包工人，此时最终搜索区域确定为 R_f，半径 r_f 为 57，则众包任务 q 最终得到的 K 个最近邻对象分别为 p_1, p_2, p_3, p_4，且 $D_{p2} < D_{p1} < D_{p3} < D_{p4}$。

4.3 算法分析

在 CT-KNN 查询算法中，找到距离所选众包任务最近的 K 个众包工人的时间复杂度和空间复杂度分别为 $O(n)$ 和 $O(n)$，将这 K 个众包工人进行排序的时间复杂度和空间复杂度分别为 $O(\log n)$和 $O(1)$。因此，CT-KNN 查询算法的时间复杂度和空间复杂度分别为 $O(n\log n)$和 $O(n)$。

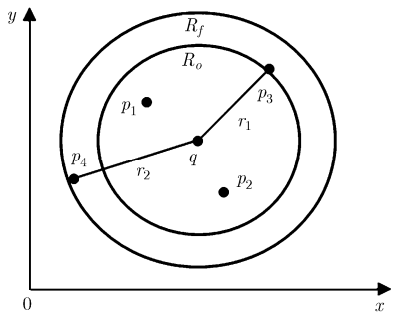


图 1　CT-KNN 算法

5　基于动态效用的阈值选择算法的众包任务分配

5.1 算法整体设计思想

本文扩展 Greedy-RT 算法[11]，设计了一种基于动态效用的阈值选择(Dynamic Utility Threshold Selection, DUTS)算法。针对 Greedy-RT 算法适用于两类对象匹配时仅允许一类对象动态出现的问题，本文设计的 DUTS 算法可在两类对象随时动态出现时进行分配，进而更大程度地优化平台效益。算法总体被分为如下 4 步：

(1)计算任务成功率期望：　根据 CT-KNN 查询算法找出距离众包任务最近的 K 个众包工人，通过其在平台中完成任务的历史记录计算出任务成功率期望，并引入了时间效应因子权重函数[17]来体现历史记录对于任务成功率的影响趋势，距离当前时刻越近，对于成功率的影响越大，否则，影响较小并最终趋向于 0。其表示为

$$w_i(t') = \frac{\cos(2\pi t'/T') + 1}{2} \tag{1}$$

其中，$t' \in [0, T'/2]$，$w_i(t') \in [0,1]$，适用于反映时间对于成功率的重要程度。

通过众包工人 w 在平台中完成任务的历史记录计算出该工人的加权任务成功率 $s_w^{'}$、带时间权重的任务成功率 s_w^* 和任务成功率期望 $\overline{s}_w$，公式为

$$s_w^* = s_w^{'} \cdot w_i(t') \tag{2}$$

$$\overline{s}_w = \sum_{i=1}^{n} s_w^* \Big/ n \tag{3}$$

(2)计算任务实际完成时长：　在任务分配过程中，假设众包工人的历史任务成功率可以代表其可靠性，为了激励工人提高其任务成功率，本文通过设置惩罚因子来延长任务成功率期望较低的工人完成任务的实际时长，以此对其进行一定程度的惩罚和警告，从而尽可能地保证平台为众包任务所分配的工人都是可靠的。根据层次分析法[18]，可将任务成功率期望 $\overline{s}_w$ 分成 3 段，并将其与惩罚因子 e 之间建立如表 2 所示对应关系，并依据定义 5，可计算出任务实际完成时长 z。众包工人的历史任务成功率越低，与之对应的惩罚因子的值越大(实验表明，当 $e \in [1,2]$ 时，算法效果最好)，众包工人接到平台所分配任务的等待时间就会越长。

(3)计算工人的单位时间效用：　根据定义 6 计

表 2 任务成功率期望 $\bar{s}_w$ 与惩罚因子 e 对应表

任务成功率期望 $\bar{s}_w$	惩罚因子 e	该工人接到任务所需等待的时间
$0 \le \bar{s}_w < 0.6$	2.0	正常分配时间的 2.0 倍
$0.6 \le \bar{s}_w < 0.8$	1.2	正常分配时间的 1.2 倍
$0.8 \le \bar{s}_w \le 1.0$	1.0	正常分配时间

算当前可做出分配的效用 $U(t,w)$，该效用函数在 DUTS 算法中，对于每个出现的众包工人 w，将其看做 c_w 个同时出现且容量为 1 的众包工人[5]。

(4)进行众包任务分配： 动态地选择一个阈值，将第 3 步计算出的效用和所选阈值做比较，效用高于该阈值的工人可优先进行分配，当仅余下低于阈值的分配时，按照效用从高到底做出分配，直至所有任务和工人均已完成分配。

5.2 DUTS 算法执行过程

算法首先根据任务分配全过程中可能获得的最大效用 $U_{\max}$，动态地选择一个阈值 e^k，其中 $U_{\max}$ 可以通过效用函数计算得到。当一个新任务或者工人出现时，计算出当前可做出匹配的效用和阈值进行比较。若所有工人和任务均已完成分配，将所有分配结果加入结果集 A 中。算法执行过程如表 3 所示。

5.3 算法举例

例：任务 T，工人 W 在当前空间中的位置关系如图 2 所示。

设集合 T,W 中元素信息见表 4。由于 $U_{\max}=100$，$\theta=\left\lfloor \ln\left(U_{\max}+1\right)\right\rfloor=5$，$k$ 从 1, 2, 3, 4, 5 中随机取值。假设取 $k=4$，则阈值 $e^k \approx 20.1$。

设对象出现顺序 $w_1,t_1,t_2,w_2,t_3,t_4,w_3$，动态效用阈值算法运行过程如图 3 所示。如图 3(a)所示，首先对最先出现的 2 个对象 w_1,t_1 进行任务分配。该分配效用 $U\left(t_1,w_1\right)=90.0>e^k$，算法做出该分配。随后出现对象是 t_2，如图 3(b)所示。由于算法已做出

表 3 基于动态效用的阈值选择算法

算法 2 DUTS 算法

输入：任务 t，工人 w，容量 c_w，效用 $U(.,.)$

输出：一个分配结果集 A

(1)计算最大效用 $U_{\max}$ 和 θ, $\theta=\left\lfloor \ln\left(U_{\max}+1\right)\right\rfloor$，并从整数集合 $\{1,2,\cdots,\theta\}$ 中随机地选择一个 k 计算出阈值 e^k；

(2)当一个新的任务 t 或者工人 w 出现时，执行步骤 3；否则，执行步骤 6；

(3)计算当前可做出分配工人的 $\bar{s}_w$ 和 z，输出 $U(t,w)$；

(4)若 $U(t,w)<e^k$，返回步骤 2；

(5)若 $U(t,w)\ge e^k$ 且 c_w 未达到上限，将此分配加入结果集 A 中，返回步骤 2；

(6)若仍存在未分配的任务 t，工人 w，计算当前可做出分配的效用 $U(t,w)$，并按效用 U 降序进行分配，将分配结果加入 A，算法结束。

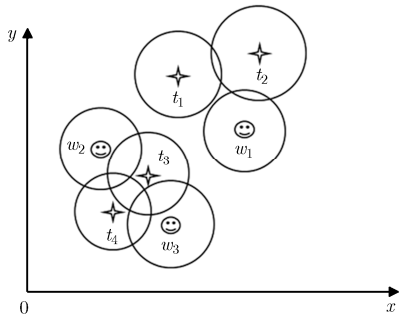


图 2 任务、工人位置示意图

表 4 任务、工人信息表

对象	坐标	半径	报酬	任务成功率期望	容量	任务完成期
t_1	(50,70)	15.0	20	–	–	0.2
t_2	(70,75)	16.5	100	–	–	0.9
t_3	(40,40)	13.5	60	–	–	0.6
t_4	(30,30)	12.5	90	–	–	1.0
w_1	(65,55)	13.5	–	0.9	1	
w_2	(25,50)	13.5	–	0.2	2	
w_3	(45,25)	15.0	–	0.8	2	

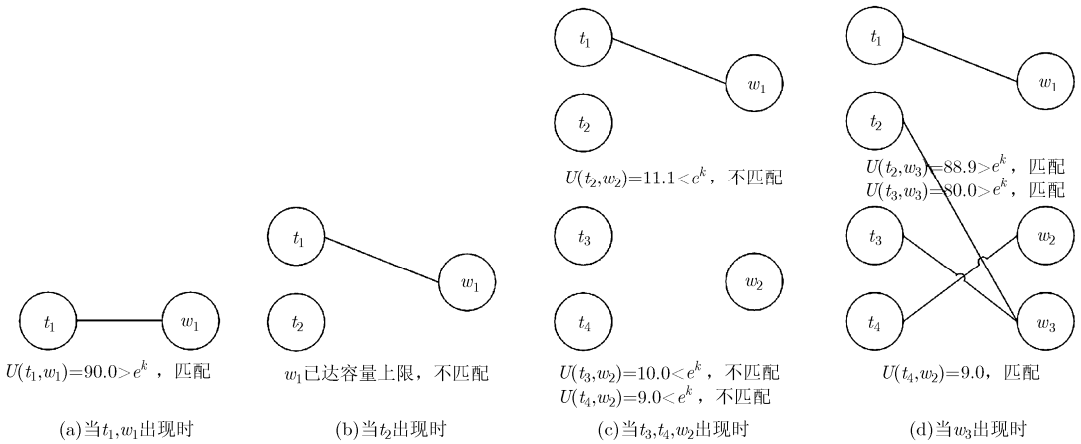


图 3 基于动态效用的阈值选择算法运行过程

$\langle w_1,t_1\rangle$ 的分配，且 w_1 的容量为 1，所以此时不做任务分配。随后出现的 3 个对象是 w_2,t_3,t_4，如图 3(c)所示，可做出分配 $\langle t_2,w_2\rangle$，$\langle t_3,w_2\rangle$ 和 $\langle t_4,w_2\rangle$。由于 $U(t_2,w_2)=11.1<e^k$，$U(t_3,w_2)=10.0<e^k$,$U(t_4,w_2)=9.0<e^k$，所以算法不做任务分配。最后出现的对象是 w_3，如图 3(d)所示。通过 w_3 可做出分配 $\langle t_2,w_3\rangle$，$\langle t_3,w_3\rangle$ 和 $\langle t_4,w_3\rangle$。其效用分别为 $U(t_2,w_3)=88.9>e^k$，$U(t_3,w_3)=80.0>e^k$，$U(t_4,w_3)=72.0>e^k$，由于 w_3 的容量为 2，所以算法做出分配 $\langle t_2,w_3\rangle$ 和 $\langle t_3,w_3\rangle$。此时，仅剩下 t_4 和 w_2 还未分配，由于在当前时空中所有任务均需在任务结束时间之前作出分配，因此算法做出分配 $\langle t_4,w_2\rangle$，并将其加入结果集 A 中。因此，最终分配为 $A=\{\langle t_1,w_1\rangle,\langle t_2,w_3\rangle,\langle t_3,w_3\rangle,\langle t_4,w_2\rangle\}$，总效用为 $90.0+88.9+80.0+9.0=267.9$。

6　实验分析

本实验在具有 2.4 GHz Inter(R)Core(TM)i5 处理器和 4 GB 内存的机器上运行，操作系统为 Windows 10，编程语言为 Java，实验数据如表 5 所示。

6.1 基于模拟数据的结果分析

本实验从总效用、运行时间和内存开销 3 个方面比较不同参数对 Greedy-RT 算法和 DUTS 算法的影响。模拟数据依据表 5 生成。设最大效用 $U_{\max}$ 为 100，在实际应用中，最大效用可以通过当前区域内的分配情况预估得到。

(1)在总效用方面，具体实验结果如图 4 所示。

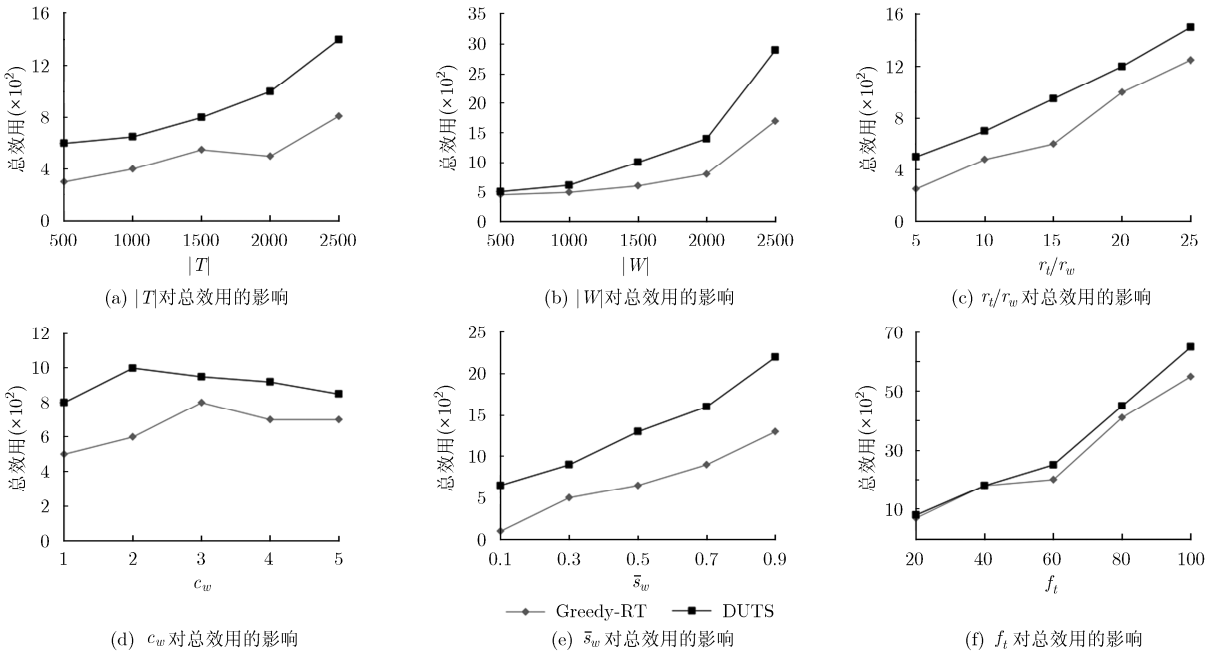
表 5 实验数据参数

参数	取值
$\|T\|$	500, 1000, 1500, 2000, 2500
$\|W\|$	500, 1000, 1500, 2000, 2500
r_t/r_w	5, 10, 15, 20, 25
c_w	1, 2, 3, 4, 5
$\bar{s}_w$	0.1, 0.3, 0.5, 0.7, 0.9
f_t	20, 40, 60, 80, 100
$U_{\max}$	100

随着众包任务、众包工人、范围半径、众包工人容量和任务成功率的增长，两种算法的效用都在增大，其中 DUTS 算法的增长速度更快一些；随着报酬的不断增加，两种算法的效用均呈现出线性增长，DUTS 算法略高于 Greedy-RT 算法。

(2)在运行时间方面，具体实验结果如图 5 所示。随着众包任务的数量增长，当$|T|$>1000 时，DUTS 算法的运行时间高于 Greedy-RT 算法；当众包工人的数量增长时，Greedy-RT 算法耗时更长一些；改变范围半径和众包工人容量时，DUTS 算法所需要的时间短于 Greedy-RT 算法；改变任务成功率和报酬时，DUTS 算法的耗时长于 Greedy-RT 算法，且可发现任务成功率和报酬的增长对算法运行时间的影响不大。

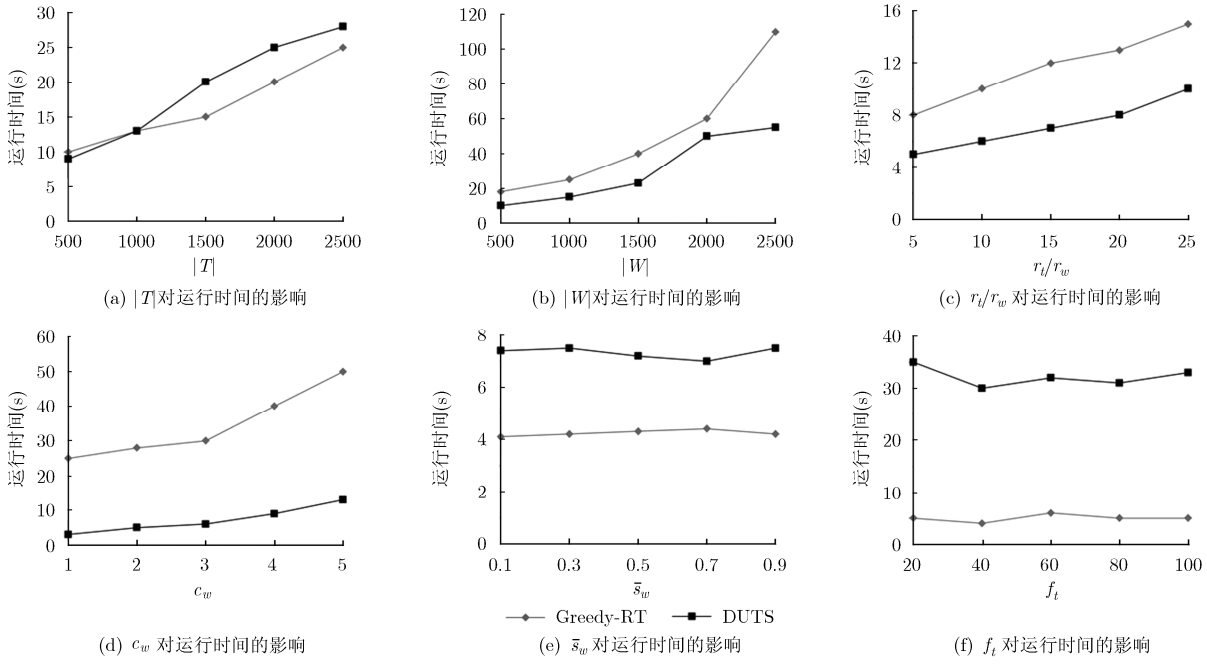
(3)在内存开销方面，具体实验结果如图 6 所示。改变众包任务，由于 Greedy-RT 算法需要占用一定的空间来存储假设的匹配结果，所以总体来说 Greedy-RT 算法内存开销略大；改变众包工人，当



(a) $|T|$对总效用的影响　(b) $|W|$对总效用的影响　(c) r_t/r_w 对总效用的影响

(d) c_w 对总效用的影响　(e) $\bar{s}_w$ 对总效用的影响　(f) f_t 对总效用的影响

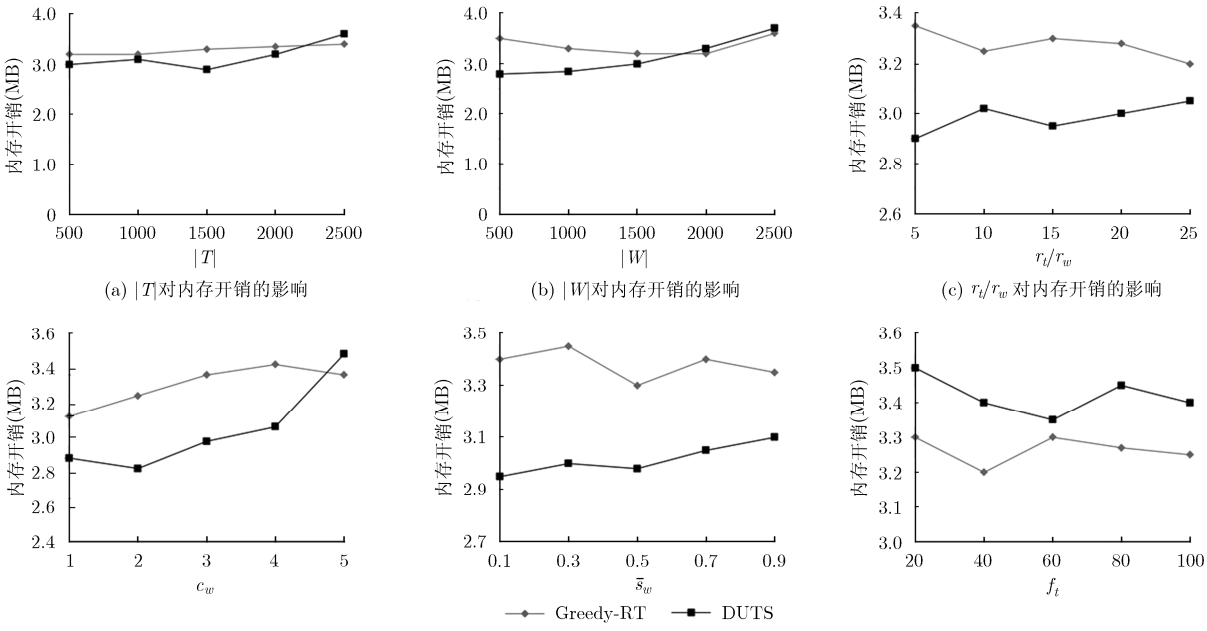
图 4　Greedy-RT 算法和 DUTS 算法在总效用方面的实验结果



(a) $|T|$对运行时间的影响 (b) $|W|$对运行时间的影响 (c) r_t/r_w 对运行时间的影响

(d) c_w 对运行时间的影响 (e) $\bar{s}_w$ 对运行时间的影响 (f) f_t 对运行时间的影响

图 5 Greedy-RT 算法和 DUTS 算法在运行时间方面的实验结果



(a) $|T|$对内存开销的影响 (b) $|W|$对内存开销的影响 (c) r_t/r_w 对内存开销的影响

(d) c_w 对内存开销的影响 (e) $\bar{s}_w$ 对内存开销的影响 (f) f_t 对内存开销的影响

图 6 Greedy-RT 算法和 DUTS 算法在内存开销方面的实验结果

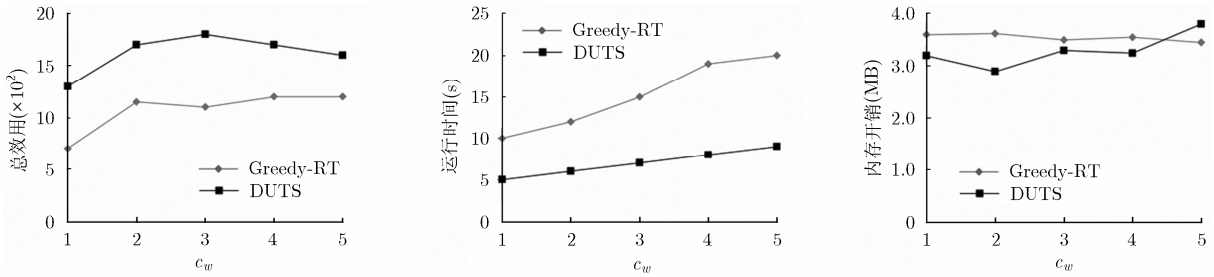
$|W|$<2000 时，Greedy-RT 算法匹配消耗的空间更大，但由于 DUTS 算法需要空间来进行候选工人的选择，所以当$|W|$>2000 时 DUTS 算法内存消耗高于 Greedy-RT 算法；改变范围半径和任务成功率，Greedy-RT 算法的消耗更多；改变工人容量，当 c_w<4 时，Greedy-RT 算法消耗更多；改变报酬，DUTS 算法消耗更大一些。

6.2 基于真实数据的结果分析

本实验的真实数据由聚数力网站(http://dataju.cn)[19]上下载并整理得到。众包工人的容量在 1~5 的范围内。实验结果如图 7 所示，基本与模拟数据集所得实验结果相似：(1)在总效用方面，DUTS 算法始终优于 Greedy-RT 算法。(2)在运行时间方面，Greedy-RT 算法更长一些，且 DUTS 算法较为稳定，基本保持在 5~10 s 之间。(3)在内存开销方面，当 c_w<4 时，Greedy-RT 算法的消耗多一些。

6.3 实验结论

通过对不同的模拟数据集和真实数据集进行实



(a) c_w 对总效用的影响　(b) c_w 对运行时间的影响　(c) c_w 对内存开销的影响

图 7 真实数据集结果

验，根据上述实验结果可以发现：(1)在总效用方面，DUTS 算法在大多数情况下优于 Greedy-RT 算法。(2)在运行时间方面，随着众包工人任务的数量、范围半径和容量的增长，Greedy-RT 算法的时间消耗略高于 DUTS 算法，当众包任务的数量、任务成功率和报酬发生改变时，Greedy-RT 算法所用时间更短。(3)在内存开销方面，当众包任务的数量、众包工人的数量、范围半径、容量和任务成功率增长时，Greedy-RT 算法的消耗高于 DUTS 算法，当报酬增长时，DUTS 算法的开销更大一些。

综上所述的结果可说明，本文的算法较高效，且具有较好的实际应用价值。

7　结束语

本文研究了时空众包环境下的在线任务分配问题，利用 CT-KNN 算法选出合适的众包工人，并动态选择不同的阈值进行分配，有效提高平台的分配效率。本文通过实验证明了所提算法具有较好的近似效果，且能够满足时空众包对于实时性的要求。在未来的时空众包研究中，将在本文的基础上进一步探讨如何优化在线任务分配算法，并将任务预测和任务规划问题加入到下一步的研究内容中。

参 考 文 献

[1] HOWE J. The rise of crowdsourcing[J]. *Wired Magazine*, 2016, 14(6): 1–4.

[2] BRABHAM D C. Crowdsourcing the public participation process for planning projects[J]. *Planning Theory*, 2009, 8(3): 242–262.

[3] 芮兰兰, 张攀, 黄豪球, 等. 一种面向众包的基于信誉值的激励机制[J]. 电子与信息学报, 2016, 38(7): 1808–1815. doi: 10.11999/JEIT151095.
RUI Lanlan, ZHANG Pan, HUANG Haoqiu, *et al*. Reputation-based incentive mechanisms in crowdsourcing [J]. *Journal of Electronics & Information Technology*, 2016, 38(7): 1808–1815. doi: 10.11999/JEIT151095.

[4] 施战, 辛煜, 孙玉娥, 等. 基于用户可靠性的众包系统任务分配机制研究[J]. 计算机应用, 2017, 37(9): 2449–2453.
SHI Zhan, XIN Yu, SUN Yue, *et al*. An allocation mechanism based on the reliability of users for crowdsourcing systems[J]. *Journal of Computer Applications*, 2017, 37(9): 2449–2453.

[5] 冯剑红. 基于众包的数据查询处理关键技术研究[D]. [博士论文], 清华大学, 2015.
FENG Jianhong. Key techniques of crowdsourced query processing[D]. [Ph.D. dissertation], Tinghua University, 2015.

[6] LI Yu, YIU Manlung, and XU Wenjian. Oriented online route recommendation for spatial crowdsourcing task workers[C]. 14th International Symposium on Advances in Spatial and Temporal Database, SSTD 2015, HongKong, China, 2015: 137–156. doi: 10.1007/978-3-319-22363-6_8.

[7] 童咏昕, 袁野, 成雨蓉, 等. 时空众包数据管理技术研究综述[J]. 软件学报, 2017, 28(1): 35–58. doi: 10.13328/j.cnki.jos.005140.
TONG Yongxin, YUAN Ye, CHENG Yurong, *et al*. Survey on spatiotemporal crowdsourced data management tecllniques[J]. *Journal of Software*, 2017, 28(1): 35–58. doi: 10.13328/j.cnki.jos.005140.

[8] 宋天舒, 童咏昕, 王立斌, 等. 空间众包环境下的 3 类对象在线任务分配[J]. 软件学报, 2017, 28(3): 611–630. doi: 10.13328/j.cnki.jos.005166.
SONG Tianshu, TONG Yongxin, WANG Libin, *et al*. Online task assignment for three types of objects under spatial crowdsourcing environment[J]. *Journal of Software*, 2017, 28(3): 611–630. doi: 10.13328/j.cnki.jos.005166.

[9] CHENG Peng, LIAN Xiang, CHEN Lei, *et al*. Task assignment on multi-skill oriented spatial crowdsourcing[J]. *IEEE Transactions on Knowledge & Data Engineering*, 2016, 28(8): 2201–2215. doi: 10.1109/TKDE.2016.2550041.

[10] HASSAN U U and CURRY E. Efficient task assignment for spatial crowdsourcing: A combinatorial fractional optimization approach with semi-bandit learning[J]. *Expert Systems with Applications*, 2016, 58: 36–56.

[11] TONG Yongxin, SHE Jieying, DING Bolin, *et al*. Online mobile micro-task allocation in spatial crowdsourcing[C]. 2016 IEEE 32nd International Conference on Data Engineering, 2016: 49–60. doi: 10.1109/ICDE.2016.7498228.

[12] XUE Andyyuan, ZHANG Rui, ZHENG Yu, *et al*. Destination prediction by sub-trajectory synthesis and privacy protection against such prediction[C]. 2013 IEEE 29th International Conference on Data Engineering, 2013: 254–265. doi: 10.1109/ICDE.2013.6544830.

[13] 杨航. 基于历史信息的移动对象轨迹预测研究[D]. [硕士论文], 广西师范大学, 2016.
YANG Hang. Research on prediction of trajectories of moving objects based on historical information[D]. [Master dissertation], Guangxi Normal University, 2016.

[14] 宋晓宇, 孙业挺, 孙焕良. CYPK-KNN: 一种改进的移动对象 KNN 查询算[J]. 沈阳建筑大学学报(自然科学版), 2006, 22(6): 1004–1007.
SONG Xiaoyu, SUN Yeting, and SUN Huanliang. CYPK-KNN: A modified monitoring KNN queries over moving objects algorithm[J]. *Journal of Shenyang Jianzhu University* (*Natural Science*), 2006, 22(6): 1004–1007.

[15] 邓斌. 带权不确定图的 K 最近邻查询算法[D]. [硕士论文], 上海海洋大学, 2015.
DENG Bin. K-nearnest neighbors query algorithm in weighted uncertain graph[D]. [Master dissertation], Shanghai Ocean University, 2015.

[16] 牛剑光, 陈荦, 赵亮, 等. 高度动态环境下移动对象连续 K 近邻查询算法[J]. 计算机科学, 2011, 38(3): 182–186.
NIU Jianguang, CHEN Luo, ZHAO Liang, *et al*. Processing continuous K nearest neighbor queries on highly dynamic moving objects[J]. *Computer Science*, 2011, 38(3): 182–186.

[17] 梁俊杰, 尹利. 一种基于半余弦函数的个性化推荐方法[P]. 中国专利, ZL 201510103073.8, 2016-05-11.

[18] 百度百科. 层次分析法(运筹学理论)[OL]. https://baike.baidu.com/item/层次分析法/1672, 2017.6.

[19] 聚数力. Uber 纽约市乘车数据[OL]. http://dataju.cn/Dataju/web/datasetInstanceDetail/210, 2017, 7.

余敦辉： 男，1974 年生，副教授，研究方向为服务计算、大数据.
张灵莉： 女，1993 年生，硕士生，研究方向为大数据.
付 聪： 男，1991 年生，硕士生，研究方向为大数据.