软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(11):5249−5266 [doi: 10.13328/j.cnki.jos.006731]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于事件的社交网络上的 CCP 事件规划*

吴定明, 林俊杰, 陆克中, 徐宇明

(深圳大学 计算机与软件学院, 广东 深圳 518060)

通信作者: 陆克中, E-mail: kzlu@szu.edu.cn

摘　要: 在基于事件的社交网络 (EBSNs) 上, 事件规划一直是一个热点研究问题. 事件规划问题的核心是基于事件和用户的约束条件, 对于一组事件, 为每个事件选择一组用户, 以最大化预先定义的目标函数. 在实际应用中, 事件冲突、事件容量、用户容量、社交偏好、事件偏好, 简称为 CCP, 即冲突 conflict、容量 capacity、偏好 preference, 是规划方案需要考虑的重要因素. 然而, 现有的所有工作均未在研究事件规划问题时考虑 CCP. 为了获得更加合理有效的规划方案, 首次提出一种 CCP 事件规划问题. 相比只考虑部分因素的规划, CCP 事件规划面临着问题更复杂、约束条件更多的困难. 为了有效求解该问题, 提出事件导向的贪心用户选择算法、事件导向的动态规划算法及基于收益预测的快速版本和事件导向的近似最优用户选择算法. 大量的实验结果验证所提算法的有效性和高效性.

关键词: 基于事件的社交网络; 事件规划; 多约束

中图法分类号: TP311

中文引用格式: 吴定明, 林俊杰, 陆克中, 徐宇明. 基于事件的社交网络上的CCP事件规划. 软件学报, 2023, 34(11): 5249–5266. http://www.jos.org.cn/1000-9825/6731.htm

英文引用格式: Wu DM, Lin JJ, Lu KZ, Xu YM. CCP-aware Event Planning on Event-based Social Networks. Ruan Jian Xue Bao/Journal of Software, 2023, 34(11): 5249–5266 (in Chinese). http://www.jos.org.cn/1000-9825/6731.htm

CCP-aware Event Planning on Event-based Social Networks

WU Ding-Ming, LIN Jun-Jie, LU Ke-Zhong, XU Yu-Ming

(College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China)

Abstract: Event planning on event-based social networks (EBSNs) has been attracting research efforts for decades. The key insight of the event planning problem is assigning a group of users to a set of events, such that a pre-defined objective function is maximized, subject to a set of constraints. In real applications, important factors, such as event conflicts, event capacities, user capacities, social preferences between users, event preferences, abbreviated as conflict, capacity, and preference (CCP), are necessary to be considered in the event planning. This work summarizes these important factors as conflict, capacity, and preference, denoted by CCP for short. Existing works do not consider CCP when computing event plans. Hence, this study proposes CCP-based event planning problem on event-based social networks, so that more reasonable event plans can be obtained. Since this is the first time to propose CCP-based event planning problem, none of the existing methods can be directly applied to solve it. Compared with previous event planning problem that only considers part of CCP factors, the challenges of solving the CCP-based event planning problem include the complexity of the new problem, more constraints involved. Hence, this paper proposes these algorithms to solve the CCP-based event planning problem in efficient and effective ways. Extensive experiments are conducted and the results prove the effectiveness and the efficiency of the proposed algorithms.

Key words: event-based social networks (EBSNs); event planning; multi-constraints

移动互联网时代的到来, 为线上和线下互动 (O2O) 的新型商业模式提供了绝佳的发展机遇. 基于事件的社交网络 (event-based social networks, EBSNs) 逐渐成为人们线上获取信息分享信息、线下进行面对面交流互动的平

* 基金项目: 广东省自然科学基金 (2019A1515011721, 2019A1515011064); 深圳市基础研究项目 (20200806102941001)
收稿时间: 2021-08-10; 修改时间: 2021-10-06, 2021-11-25, 2022-03-14; 采用时间: 2022-04-04; jos 在线出版时间: 2023-05-18
CNKI 网络首发时间: 2023-05-19

台, 例如国内的豆瓣、国外的 Meetup、Plancast 和 EventBrite 等. 在此类平台上, 活动组织者发布社交活动的时空信息 (活动的召集地点和活动的举办时间) 和活动的文本属性 (描述活动的主题和内容) 等, 用户可以在线上注册感兴趣的活动, 然后在线下参加相应的活动. 截至 2017 年, Meetup 约有 3 500 万用户; 截止到 2018 年初, 豆瓣的注册用户达到了 1.6 亿, 月活跃用户 3 亿人次.

在基于事件的社交网络上, 事件规划一直是一个热点研究问题. 事件规划问题的核心是基于事件和用户的约束条件, 对于一组事件, 为每个事件选择一组用户, 以最大化预先定义的目标函数. 由于约束条件和目标函数形式各异, 因此, 事件规划问题的具体形式多种多样. 在实际应用中, 事件冲突、事件容量、用户容量、社交偏好、事件偏好是规划方案需要考虑的 5 个重要因素. 事件冲突这一因素避免给同一个用户安排时间上冲突的事件, 事件容量这一因素避免参与人数不足或者参与人数过多影响事件举行, 用户容量这一因素确保用户能有足够的精力参与每个安排的事件, 社交偏好这一因素使得参与同一事件的用户有较好的社交体验, 事件偏好这一因素确保为每一个用户安排其感兴趣的事件. 同时考虑这 5 个因素的规划问题旨在得出一个能同时满足活动组织者和用户的需求的规划方案, 这 5 个因素缺少任意一个都会影响规划方案的可行性和效果.

表 1 列出了已有的 6 种代表性事件规划问题在这 5 个因素上的表现, 其中, 效用感知的社交事件规划 (USEP) 问题[1]忽略了社交偏好; 社交事件组织 (SEO) 问题[2]没有考虑事件之间的冲突, 也无法计算为每个用户安排多个事件的规划方案; 基于冲突与容量的社交事件规划 (GEACC) 问题[3]、基于约束的全局社交事件规划 (GEPC) 问题[4]和影响力感知的社交事件规划 (IGEPA) 问题[5]都忽略了用户的社交偏好; 反馈感知的社交事件规划 (FASEA) 问题[6]没有考虑用户对事件的偏好和用户的社交偏好. 本文将上述的 5 个因素归纳为冲突 conflict、容量 capacity、偏好 preference, 简称 CCP.

表 1 CCP 规划问题与现有事件规划问题对比

约束条件	USEP[1]	SEO[2]	GEACC[3]	GEPC[4]	FASEA[6]	IGEPA[5]	CCP
事件冲突	√	×	√	√	√	√	√
事件容量	√	√	√	√	√	√	√
用户容量	√	×	√	√	√	√	√
社交偏好	×	√	×	×	×	×	√
事件偏好	√	√	√	√	×	√	√

根据上文所述, 在基于事件的社交网络规划中, 考虑 CCP 的规划问题比只考虑部分因素的规划问题在实际应用中能获得更合理的规划方案. 因此, 如何定义一种新的基于 CCP 的事件规划问题, 是本文面临的挑战之一. 另外, 由于现有工作从未考虑 CCP 全部因素的规划问题, 因此现有的事件规划的算法能够提供的参考性有限. 而考虑 CCP 的规划问题显然比只考虑部分因素的规划问题更复杂、约束条件更多, 从而更加难以解决. 为此, 本文提出了基于事件的社交网络上的 CCP 事件规划问题, 同时兼顾冲突和容量的约束以及偏好的需求, 以弥补现有工作的不足. 针对该问题, 本文提出了 3 个求解算法, 分别是事件导向的贪心用户选择算法, 事件导向的动态规划算法及基于收益预测的快速版本和事件导向的近似最优用户选择算法. 最后, 通过一些真实数据集上的实验, 验证了本文提出算法的有效性和高效性.

本文第 1 节对基于事件的社交网络上的 CCP 事件规划问题进行定义. 第 2 节对相关工作加以总结. 第 3 节详细介绍 3 个算法, 并分析其复杂度. 第 4 节通过真实数据集上的实验来验证本文提出算法的有效性和高效性. 第 5 节对全文加以总结, 并提出对未来工作的设想.

1 相关工作

本文研究的 CCP 问题是基于事件的社交网络上的一种规划问题, 同时也和二分图匹配问题相关: 它考虑了用户之间的社交关系, 是一种更复杂的二分图匹配问题. 本节从以下两个方面总结与本文相关的现有工作: (1) 基于事件的社交网络上的规划问题; (2) 最大加权二分匹配问题.

1.1 基于事件的社交网络上的规划问题

基于事件的社交网络上的规划问题的核心是如何匹配事件和用户, 现有的工作基于不同的目标收益研究了在不同约束条件下的求解算法.

早期, Li 等人在文献 [2] 中研究了社交事件组织 (SEO) 问题, 该工作限定每个事件有参与人数的下界和上界, 以用户对事件的兴趣度和用户之间的社交关系为目标收益, 基于贪心策略求解为每个用户安排一个事件的规划方案. 但是, 该研究工作没有考虑事件之间的冲突, 也无法计算为每个用户安排多个事件的规划方案.

She 等人在文献 [1] 中研究了效用感知的社交事件规划 (USEP) 问题, 该工作考虑事件在时间和空间上的冲突, 限定每个事件有参与人数的上界、每个用户有行程开销的上界, 以用户对事件的兴趣度为目标收益, 基于贪心策略求解为每个用户安排多个事件的规划方案. 同年, She 等人在文献 [3] 中研究了基于冲突与容量的社交事件规划 (GEACC) 问题, 该工作考虑事件之间的冲突, 限定每个事件有参与人数的上界、每个用户有参加事件的上界, 以用户对事件的兴趣度为目标收益, 设计了基于流网络的算法, 求解为每个用户安排多个事件的规划方案. 随后, She 等人在文献 [7] 中进一步研究了在线版本的 GEACC 问题, 该工作根据社交平台上的用户动态, 实时计算最新的事件规划方案. Cheng 等人在文献 [4,8] 中研究了基于约束的全局社交事件规划 (GEPC) 问题及其增量规划版本 (IEP), 该工作考虑事件在时间上的冲突, 限定每个事件有参与人数的下界和上界、每个用户有行程开销的上界, 以用户对事件的兴趣度为目标收益, 基于贪心策略求解为每个用户安排多个事件的规划方案. 随后, Xin 等人在文献 [9] 中进一步研究了 GEPC 问题, 提出了启发式的动态规划算法, 更高效地解决了 GEPC 问题. She 等人在文献 [6] 中研究了反馈感知的社交事件规划 (FASEA) 问题, 该工作基于用户的反馈信息, 自适应地实时计算最新的规划方案. Kou 等人在文献 [5] 中研究了影响力感知的社交事件规划 (IGEPA) 问题, 该工作考虑事件在时间和空间上的冲突, 限定每个事件有参与人数的上界、每个用户有参加事件的上界, 以用户对事件的兴趣度和用户在社交网络中的影响力为目标收益, 设计了基于线性规划的算法求解为每个用户安排多个事件的规划方案. Cheng 等人在文献 [10] 中研究了双边偏好稳态规划问题, 该工作同时考虑用户对事件的偏好和事件举办方对用户的偏好, 限定每个事件有参与人数的上界、每个用户有行程开销的上界, 基于贪心策略求解为每个用户安排多个事件的规划方案. Sun 等人在文献 [11] 中研究了个性化事件规划 (PEPA) 问题, 该工作考虑事件在时间上的冲突, 限定每个事件有参与人数的上界、每个用户有参加事件的上界, 以用户对事件的个性化兴趣度为目标收益, 基于贪心策略求解为每个用户安排多个事件的规划方案. Ding 等人在文献 [12] 中研究了基于效用-时间的社交事件规划 (UTSEP) 问题, 该工作考虑事件在时间上的冲突和用户参加事件的行程开销, 限定每个事件有参与人数的上界, 以用户对事件的兴趣度为目标收益, 设计了基于贪心策略和动态规划的算法, 求解为每个用户安排多个事件的规划方案. 但是, 这些研究工作都没有考虑用户之间的社交关系.

Tong 等人在文献 [13] 中研究了瓶颈感知的社交事件规划 (BSEA) 问题, 该工作限定每个事件有参与人数的上界, 以用户对事件的兴趣度和用户与事件的空间距离为目标收益, 基于贪心策略求解为每个用户安排多个事件的规划方案. 但是在事件的规划方案中, 该研究工作对用户的社交关系约束较弱, 简化为每个用户有一个朋友与其参加同一个事件, 没有考虑用户之间社交关系的强弱, 也忽略了参加同一个事件的多个用户之间的社交关系.

此外, 还有一些与事件规划相关但具体问题定义有差异的研究工作. Bikakis 等人在文献 [14] 中研究了社交事件调度 (SES) 问题, 该工作以最大化事件参与度为目标, 求解事件的时间规划方案, 而本文的研究是为事件选择合适的用户. Gao 等人在文献 [15] 中研究了多角色的事件组织 (MRSEO) 问题, 该工作旨在为一个事件选择能覆盖 m 个角色的 k 个用户, 使得和谐值最大化. 其中, 和谐值由用户对事件的兴趣度和用户之间的熟悉度决定. Yin 等人在文献 [16] 中研究了用户与事件的双向选择问题, 基于用户和事件举办方的画像, 设计了双向选择的算法框架, 有效地预测了事件的邀请与接收情况.

1.2 最大加权二分匹配问题

最大加权二分匹配问题是指在一个二分图上求解权重之和最大的节点匹配方案, 早期经典的求解算法没有考虑节点之间的冲突和节点的容量[17−19]. 随后, 文献 [20−23] 将节点容量引入到匹配问题中. Wong 等人在文献 [20]

中研究了空间匹配问题, 该工作将客户与其距离较近的服务供应商进行匹配, 每个服务供应商有接待客户数量的上界. Leong 等人在文献 [21,22] 中进一步研究了如何获得使得匹配的空间距离之和最小的方案, 并提出了基于边剪枝的启发式算法和近似算法. Sun 等人在文献 [23] 中研究了基于效用最大化的匹配求解方案. 上述这些研究工作都没有考虑节点之间的冲突. 另外, 在线版的二分匹配问题[24−28]主要针对待匹配节点预先未知、增量获得的情况, 其求解算法需要在收到节点数据时动态地计算匹配方案.

2 问题定义

本节给出基于事件的社交网络上的 CCP 规划问题的正式定义. 设基于事件的社交网络上的事件集为 $E=\{e_1,\ldots,e_m\}$, 共包含 m 个事件; 每个事件 $e_j\in E$ 以四元组 $\langle\gamma_j,\delta_j,t_j^s,t_j^t\rangle$ 表示, 其中 γ_j 为事件 e_j 参与人数的下界, 即事件 e_j 最少需要被分配多少人, δ_j 为事件 e_j 参与人数的上界, 即事件 e_j 最多可以被分配多少人, t_j^s 和 t_j^t 分别表示事件 e_j 的开始时间和结束时间. 设基于事件的社交网络上的用户集为 $U=\{u_1,\ldots,u_n\}$, 共包含 n 个用户; 每个用户 $u_i\in U$ 有一个参加事件数的上界 c_i, 即用户 u_i 最多能被安排多少事件.

一个用户 $u_i\in U$ 对一个事件 e_j 感兴趣的程度, 以事件偏好 $\phi(u_i,e_j):U\times E\to[0,1]$ 来表示, $\phi(u_i,e_j)$ 的值越大表示用户 u_i 对事件 e_j 越感兴趣. 用户 u_i 与用户 u_j 之间的关系以社交偏好 $\psi(u_i,u_j):U\times U\to[0,1]$ 来表示, 两个用户之间的社交偏好是对称的, 即 $\forall u_i,u_j\in U,\psi(u_i,u_j)=\psi(u_j,u_i)$. $\psi(u_i,u_j)$ 的值表示用户 u_i 和 u_j 愿意参加同一事件的可能性, 如果 $\psi(u_i,u_j)=0$, 则表示这两个用户不愿意参加同一事件. 事件 e_i 与事件 e_j 在时间上的冲突关系以 $\phi(e_i,e_j):E\times E\to\{0,1\}$ 来表示. 如果 $[t_i^s,t_i^t]\cap[t_j^s,t_j^t]=\varnothing$, 则 $\phi(e_i,e_j)=0$, 即事件 e_i 与事件 e_j 在时间上不冲突, 可以被分配给同一个用户; 否则 $\phi(e_i,e_j)=1$. 两个事件之间的冲突关系是对称的, 即 $\forall e_i,e_j\in E,\phi(e_i,e_j)=\phi(e_j,e_i)$.

事件集 E 和用户集 U 上的一个规划 M 是 E 与 U 之间的一个多重匹配. 在一个规划 M 中, 一个用户 u_i 被安排参加一组事件, 以 $M(u_i)$ 表示, 且 $M(u_i)\subseteq E$; 一个事件 e_j 被分配一组用户, 以 $M^{-1}(e_j)$ 表示, 且 $M^{-1}(e_j)\subseteq U$. M 是一个可行规划当且仅当满足下列 3 个条件.

(1) 事件的参与人数要不少于其参与人数下界且不大于其参与人数上界, 即 $\forall e_j\in E\left(M^{-1}(e_j)=\varnothing|\gamma_j\leqslant\left|M^{-1}(e_j)\right|\leqslant\delta_j\right)$.

(2) 用户的参加事件数要不多于其参加事件数上界, 即 $\forall u_i\in U(|M(u_i)|\leqslant c_i)$.

(3) 用户的规划中没有彼此冲突的事件, 即 $\forall u\in U\left(\forall e_i,e_j\in M(u)\left(\phi(e_i,e_j)=0\right)\right)$.

在一个可行规划 M 中, 一个事件 e 的收益定义为:

$$\theta(e)=(1-\alpha)\sum_{u\in M^{-1}(e)}\phi(u,e)+\alpha\sum_{u_i,u_j\in M^{-1}(e),i>j}\psi(u_i,u_j),$$

其中, 参数 $\alpha\in[0,1]$ 用于调整事件偏好和社交偏好在收益中的重要性. 一个可行规划 M 的整体收益定义为 M 中所有事件的收益之和:

$$\Theta(M)=\sum_{e\in E}\theta\left(M^{-1}(e)\right).$$

定义 1. 基于事件的社交网络上的 CCP 规划问题. 是指寻找一个使得整体收益 $\Theta(M)$ 最大的可行规划 M^*, 即 $M^*=\operatorname{argmax}\{\Theta(M)|M\text{ 是一个可行规划}\}$.

CCP 规划问题的计算复杂度分析. 在 CCP 问题中, 用户的数量为 $|U|$, 事件的数量为 $|E|$, 计算最大整体收益的可行规划的计算复杂度为 $O(2^{|E||U|})$.

定理 1. CCP 规划问题的判定问题版本是 NP 完全的.

证明: 文献 [2] 中的社交事件组织 (SEO) 问题是一个 NP-hard 问题. 同时, SEO 问题是 CCP 问题的一个特例 (用户参与事件数的上界 $c=1$, 所有事件不互相冲突 $\forall e_i,e_j\in E,\phi(e_i,e_j)=0$), 因此基于事件的社交网络上的 CCP 规划问题是一个 NP-hard 问题. 给定一个规划 M, 可以在多项式时间内验证 M 的可行性以及 M 的整体收益是否超过给定阈值. 因此, CCP 规划问题的判定问题版本是 NP 完全的.

3 CCP 规划问题求解算法

本节介绍 CCP 规划问题的 3 个求解算法, 分别是事件导向的贪心用户选择算法、事件导向的动态规划算法及基于收益预测的快速版本和事件导向的近似最优用户选择算法.

3.1 事件导向的贪心用户选择算法

事件导向的贪心用户选择算法的主要思路是: 基于事件 e 的参与人数上界 δ 降序, 依次为每个事件贪心地选择 δ 个用户, 与其现有事件冲突或已达到参与事件数上界的用户不在事件 e 的可选择用户范围内. 具体来说, 对于事件 e, $M^*(e)$ 表示已选用户集合, $U^*(e)$ 表示未来可选择的用户集合, 计算集合 $U^*(e)$ 中所有用户的函数值 $g(u,e|M^*(e))$, 该函数值最大的用户被优先选择. 函数 $g(u,e|M^*(e))$ 的形式, 即贪心策略有以下两种:

$$g(u,e|M^*(e))=(1-\alpha)\phi(u,e)+\alpha\sum_{u_i\in M^*(e)}\psi(u_i,u) \tag{1}$$

$$g(u,e|M^*(e))=(1-\alpha)\phi(u,e)+\alpha\sum_{u_i\in M^*(e)}\psi(u_i,u)+\frac{\alpha(\delta_e-|M^*(e)|-1)\sum\limits_{u_k\in U^*(e)-\{u\}}\psi(u_k,u)}{|U^*(e)|-1} \tag{2}$$

公式 (1) 考虑了两个因素: 待选择的用户对事件 e 的偏好 $\phi(u,e)$ 和待选择的用户与已选择的用户的社交偏好 $\sum\limits_{u_i\in M^*(e)}\psi(u_i,u)$. 公式 (2) 考虑了 3 个因素: 待选择的用户对事件 e 的偏好 $\phi(u,e)$、待选择的用户与已选择的用户的社交偏好 $\sum\limits_{u_i\in M^*(e)}\psi(u_i,u)$ 以及待选择的用户与未来可能被选择的用户的社交偏好 $\dfrac{\alpha(\delta_e-|M^*(e)|-1)\sum\limits_{u_k\in U^*(e)-\{u\}}\psi(u_k,u)}{|U^*(e)|-1}$. 该算法的伪代码见算法 1.

算法 1. 事件导向的贪心用户选择算法.

输入: 事件集 E, 用户集 U;
输出: 一个可行规划 M.

1. $PE\leftarrow$ 对所有事件根据其参与人数上界进行降序排序;
2. **For** (PE 中的每个事件 e)
3. $userList(e)\leftarrow getValidUserList(e)$;
4. **If** ($|userList(e)|<\gamma$)
5. **continue**;
6. **End If**
7. **If** ($|userList(e)|\geqslant\gamma$ and $|userList(e)\leqslant\delta|$)
8. $M^{-1}(e)\leftarrow userList(e)$;
9. **End If**
10. **If** ($|userList(e)|>\delta$)
11. $M^{-1}(e)\leftarrow$ 根据公式 (1) 或者公式 (2) 为事件 e 贪心地选择 δ 个用户;
12. **End If**
13. **For** ($M^{-1}(e)$ 中的每个用户 u)
14. $M(u)\leftarrow M(u)\cup\{e\}$;
15. **End For**
16. **End For**
17. **Return** $M\leftarrow\cup_u\{M(u)\}$;

算法 1 首先将所有事件按照参与人数上界非增排序, 获得事件偏序 PE, 即优先为参与人数上界较大的事件选择用户 (第 1 行). 对于 PE 的每个事件 e, 计算可以参与事件 e 的用户集合 $userList(e)$ (第 2–3 行). 如果 $userList(e)$ 包含的用户数量小于事件 e 的参与人数下界, 则不为该事件选择用户 (第 4–6 行); 如果 $userList(e)$ 包含的用户数量不小于事件 e 的参与人数下界同时不大于事件 e 的参与人数上界, 则选择 $userList(e)$ 中的所有用户 (第 7–9 行); 如果 $userList(e)$ 包含的用户数量大于事件 e 的参与人数上界, 则根据公式 (1) 或者公式 (2) 为事件 e 贪心地选择 δ 个用户 (第 10–12 行). 最后, 在为每个事件都选择完用户后, 就得到了一个可行规划 M. 下面用例 1 对算法 1 的过程进行说明.

子过程. $getValidUserList(e)$. /*可以参加事件 e 的用户集合*/

输入: 事件 e, 用户集 U;
输出: 可以参加事件 e 的用户集合 $userList(e)$.

```
1.  userList(e) ← U;
2.  For (U 中的每个用户 u)
3.      If (|M(u)| ⩾ c_u)
4.          userList(e) ← userList(e) − {u};
5.          continue;
6.      End If
7.      For (M(u) 中的每个事件 e_i)
8.          If (φ(e_i, e) = 1)
9.              userList(e) ← userList(e) − {u};
10.             break;
11.         End If
12.     End For
13. End For
14. Return userList(e);
```

例 1. 表 2 给出了 4 个用户的事件偏好值和 3 个事件的举办时间, 用户的社交偏好值见表 3, 表中用户后括号内的数值表示该用户的参加事件数上界, 事件后括号内的数值表示该事件的参与人数下界和上界. 算法 1 先将所有事件按照参与人数上界非增排序, 获得事件偏序 e_3, e_2, e_1, 图 1 演示了依次为 3 个事件选择的用户. 首先, 对于事件 e_3, 可以选择的用户集合 $userList(e_3)$ 为 $\{u_1,u_2,u_3,u_4\}$, 由于 $userList(e_3)$ 包含的用户数量位于事件 e_3 的参与人数下界和上界之间, 因此选择 $userList(e_3)$ 中的所有用户, 见图 1 中的步骤 (1). 接下来, 对于事件 e_2, 可以选择的用户集合 $userList(e_2)$ 为 $\{u_1,u_2,u_3,u_4\}$, 由于 $userList(e_2)$ 包含的用户数量大于事件 e_2 的参与人数上界, 因此需要从 4 个用户中选择 2 个用户. 具体的用户选择过程为: ① 计算 $userList(e_2)$ 中所有用户的函数值 $g(u,e|M^*(e))$, 以公式 (1) 为例, 即 $g(u_1,e_2|M^*(e_2))=0.5\times0.2=0.1$、$g(u_2,e_2|M^*(e_2))=0.5\times0.2=0.1$、$g(u_3,e_2|M^*(e_2))=0.5\times0.2=0.1$、$g(u_4,e_2|M^*(e_2))=0.5\times0.9=0.45$, 由于 $g(u_4,e_2|M^*(e_2))$ 最大, 因此优先选择用户 u_4, 即 $M^*(e_2)=\{u_4\}$; ② 接下来再计算用户 u_1,u_2,u_3 的函数值 $g(u,e|M^*(e))$, 即 $g(u_1,e_2|M^*(e_2))=0.5\times0.2+0.5\times0.6=0.4$、$g(u_2,e_2|M^*(e_2))=0.5\times0.2+0.5\times0.9=0.55$、$g(u_3,e_2|M^*(e_2))=0.5\times0.2+0.5\times0.1=0.15$, 由于 $g(u_2,e_2|M^*(e_2))$ 最大, 因此优先选择用户 u_2. 至此, $M^*(e_2)=\{u_2,u_4\}$ 包含的用户数量达到了事件 e_2 的参与人数上界, 那么为事件 e_2 选择的用户是 $\{u_2,u_4\}$, 见图 1 中的步骤 (2). 同理, 对于事件 e_1, 选择用户 u_3, 见图 1 中的步骤 (3). 此时, 所有事件都计算完成, 算法终止.

算法复杂度分析: 算法先对所有事件排序, 时间复杂度为 $O(|E|\log|E|)$. 接着为每个事件选择用户, 用公式 (1) 作为贪心策略, 计算所有用户的函数值 $g(u,e|M^*(e))$ 的复杂度为 $O(|U|)$, 那么基于公式 (1) 的事件导向的贪心用户选择算法的复杂度为 $O(|E||U|+|E|\log|E|)$. 用公式 (2) 作为贪心策略, 计算所有用户的函数值 $g(u,e|M^*(e))$ 的复杂度为 $O\left(|U|^2\right)$, 那么基于公式 (2) 的事件导向的贪心用户选择算法的复杂度为 $O(|E||U|^2+|E|\log|E|)$.

表 2 事件偏好和事件时间

事件	$u_1(2)$	$u_2(2)$	$u_3(3)$	$u_4(3)$	举办时间
$e_1(1,1)$	0.1	0.6	0.6	0.4	11:00–13:00
$e_2(1,2)$	0.2	0.2	0.2	0.9	11:00–15:00
$e_3(2,4)$	0.7	0.2	0.5	0.3	7:00–11:00

表 3 社交偏好

用户	$u_1(2)$	$u_2(2)$	$u_3(3)$	$u_4(3)$
$u_1(2)$	1	0.9	0.9	0.6
$u_2(2)$	0.9	1	0.6	0.9
$u_3(3)$	0.9	0.6	1	0.1
$u_4(3)$	0.6	0.9	0.1	1

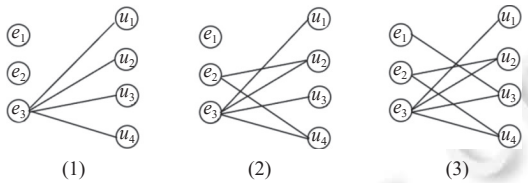

图 1 算法 1 依次为 3 个事件选择的用户

3.2 事件导向的动态规划算法

事件导向的动态规划算法的主要思路是: 基于事件 e 的参与人数上界 δ 降序, 依次为每个事件选择至多 δ 个用户, 使得该事件的收益 $\theta\left(M^{-1}(e)\right)$ 最大, 与其现有事件冲突或已达到参与事件数上界的用户不在事件 e 的可选择用户范围内. 求解最大化给定事件收益的用户组合问题的主要思路是: 给定事件 e, 其可选择的用户 u 按社交偏好总和 $\Psi(u)=\sum_{u_i\in U}\psi(u,u_i)$ 降序排序, 基于动态规划的思想, 定义子问题 $f(i,j)$ 为在前 i 个用户中选择 j 个用户获得的最大收益, $S(i,j)$ 为子问题 $f(i,j)$ 对应的用户组合, $C(i,j)$ 表示前 i 个用户中包含第 i 个用户的 j 个用户组合的集合. 当 $i=0$ (表示没有用户可选择), 或 $j=0$ (表示不需要选择用户), 或 $i<j$ (表示需要选择的用户数量大于可选择的用户数量) 时, 子问题 $f(i,j)$ 无意义. 因此, 在子问题 $f(i,j)$ 中, i 和 j 满足 $i\geqslant j$. 当 $i=j$ 时, 在前 i 个用户中选择 j 个用户的子问题的解为在前 $i-1$ 个用户中选择 $j-1$ 个用户的子问题的基础上加上第 i 个用户后获得的收益, $f(i,j)$ 用公式 (3) 计算.

$$f(i,j)=f(i-1,j-1)+(1-\alpha)\phi(u_i,e)+\alpha\sum_{u_k\in S(i-1,j-1)}\psi(u_i,u_k) \tag{3}$$

当 $i>j$ 时, $f(i,j)$ 取下面两种情况的最大值:

(1) 不选择第 i 个用户: 在前 i 个用户中选择 j 个用户的子问题转化为在前 $i-1$ 个用户中选择 j 个用户的子问题, 那么 $f(i,j)=f(i-1,j)$.

(2) 选择第 i 个用户: 在前 i 个用户中选择 j 个用户的子问题转化为在前 $i-1$ 个用户中选择 $j-1$ 个用户的所有组合中, 哪个组合加上第 i 个用户后获得的收益最大, 那么 $f(i,j)$ 用公式 (4) 计算.

$$f(i,j)=\max_{j-1\leqslant k<i}\{\theta(U_x)+(1-\alpha)\phi(u_i,e)+\alpha\sum_{u\in U_x}\psi(u_i,u)|U_x\in C(k,j-1)\} \tag{4}$$

给定事件 e, 基于其可选择用户数量 n, 则 $S(n,\delta)$ 为使得该事件收益最大的用户组合, $f(n,\delta)$ 为事件 e 获得的最大收益值. 该算法的伪代码见算法 2.

算法 2. 事件导向的动态规划算法.

输入: 事件集 E, 用户集 U;
输出: 一个可行规划 M.

1. $PE\leftarrow$ 对所有事件根据其参与人数上界进行降序排序;
2. **For** (每一个用户 u)

3. $\Psi(u) \leftarrow \sum_{u_i \in U} \psi(u, u_i)$;
4. **End For**
5. **For** (PE 中的每个事件 e)
6. $userList(e) \leftarrow getValidUserList(e)$;
7. **If** ($|userList(e)| < \gamma$)
8. **continue**;
9. **End If**
10. **If** ($|userList(e)| \geqslant \gamma$ and $|userList(e) \leqslant \delta|$)
11. $M^{-1}(e) \leftarrow userList(e)$;
12. **End If**
13. **If** ($|userList(e)| > \delta$)
14. 将 $userList(e)$ 中所有用户按照社交偏好总和 $\Psi(u)$ 非增排序;
15. **For** ($j = 1$ to δ)
16. **For** ($i = j$ to $|userList(e)|$)
17. **If** ($i == j$)
18. $f(i,j) \leftarrow f(i-1,j-1) + (1-\alpha)\phi(u_i, e) + \alpha \sum_{u_k \in S(i-1,j-1)} \psi(u_i, u_k)$;
19. $S(i,j) \leftarrow S(i-1,j-1) \cup \{u_i\}$;
20. $C(i,j) \leftarrow \{S(i,j)\}$;
21. **Else**
22. **For** ($k = j-1$ to $i-1$)
23. **For** ($U_x \in C(k, j-1)$)
24. $C(i,j) \leftarrow C(i,j) \cup \{U_x \cup \{u_i\}\}$;
25. **End For**
26. **End For**
27. $f(i,j) \leftarrow \max_{j-1 \leqslant k < i} \left\{ \theta(U_x) + (1-\alpha)\phi(u_i, e) + \alpha \sum_{u \in U_x} \psi(u_i, u) | U_x \in C(k, j-1) \right\}$;
28. $f(i,j) \leftarrow \max\{f(i,j), f(i-1,j)\}$;
29. 相应更新 $S(i,j)$
30. **End If**
31. **End For**
32. **End For**
33. $M^{-1}(e) \leftarrow S(|userList(e)|, \delta)$;
34. **End If**
35. **For** ($M^{-1}(e)$ 中的每个用户 u)
36. $M(u) \leftarrow M(u) \cup \{e\}$;
37. **End For**
38. **End For**
39. **Return** $M \leftarrow \cup_u \{M(u)\}$;

算法 2 首先将所有事件按照参与人数上界非增排序, 获得事件偏序 PE, 即优先为参与人数上界较大的事件选择用户 (第 1 行). 对于每一个用户 u, 计算该用户与其他所有用户的社交偏好总和 $\Psi(u) = \sum_{u_i \in U} \psi(u, u_i)$ (第 2–4 行).

对于 PE 中的每一个事件 e, 计算可以参与事件 e 的用户集合 $userList(e)$, 如果 $userList(e)$ 包含的用户数量小于事件 e 的参与人数下界, 那么不为该事件选择用户 (第 5–9 行); 如果 $userList(e)$ 包含的用户数量不小于事件 e 的参与人数下界同时不大于事件 e 的参与人数上界, 则选择 $userList(e)$ 中的所有用户 (第 10–12 行); 如果 $userList(e)$ 包含的用户数量大于事件 e 的参与人数上界 (第 13 行), 将 $userList(e)$ 中的所有用户按照社交偏好总和 $\Psi(u)$ 非增排序 (第 14 行), 然后基于动态规划的思想, 为事件 e 选择 δ 个用户, 最大化该事件的收益 (第 15–38 行). 最后, 当为所有事件选择完用户后, 就得到了一个可行规划 M (第 39 行). 下面用例 2 对算法 2 的过程进行说明.

例 2. 表 2 给出了 4 个用户的事件偏好值和 3 个事件的举办时间, 用户的社交偏好值见表 3, 表中用户后括号内的数值表示该用户的参加事件数上界, 事件后括号内的数值表示该事件的参与人数下界和上界. 根据算法 3, 4 个用户的社交偏好总和分别为 $\Psi(u_1)=0.9+0.9+0.6=2.4$, $\Psi(u_2)=0.9+0.6+0.9=2.4$, $\Psi(u_3)=0.9+0.6+0.1=1.6$, $\Psi(u_4)=0.6+0.9+0.1=1.6$. 将所有的事件按照参与人数上界非增排序, 获得事件偏序 e_3, e_2, e_1, 图 2 演示了算法 2 依次为 3 个事件选择的用户. 首先, 对于事件 e_3, 可以选择的用户集合 $userList(e_3)$ 为 $\{u_1,u_2,u_3,u_4\}$, 由于 $userList(e_3)$ 包含的用户数量处于事件 e_3 的参与人数下界和上界之间, 因此选择 $userList(e_3)$ 中的所有用户, 见图 2 中的步骤 (1). 接下来, 对于事件 e_2, 可以选择的用户集合 $userList(e_2)$ 为 $\{u_1,u_2,u_3,u_4\}$, 由于 $userList(e_2)$ 包含的用户数量大于事件 e_2 的参与人数上界, 因此需要从 4 个用户中选择 2 个用户. 具体的用户选择过程为: 将 $userList(e_2)$ 中的所有用户按照社交偏好总和非增排序, 获得偏序 u_1、u_2、u_3、u_4, 根据算法 3, $f(i,j)$、$S(i,j)$、$C(i,j)$ 的计算过程见表 4、表 5 和表 6. 根据计算结果, 为事件 e_2 选择用户 $S(4,2)=\{u_2,u_4\}$, 获得最大收益, 见图 2 中的步骤 (2). 同理, 对于事件 e_1, 选择用户为 u_3, 见图 2 中的步骤 (3). 此时, 所有事件都计算完成, 算法终止.

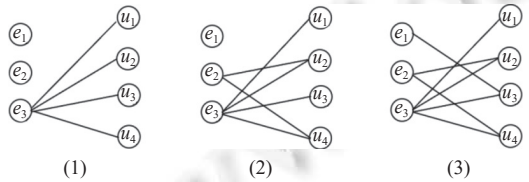

图 2 算法 2 依次为 3 个事件选择的用户

表 4 算法 2 中的计算过程 $f(i,j)$

i	j		
	0	1	2
0	—	—	—
1	—	0.1	—
2	—	0.1	0.65
3	—	0.1	0.65
4	—	0.45	1.0

表 5 算法 2 中的计算过程 $S(i,j)$

i	j		
	0	1	2
0	—	—	—
1	—	$\{u_1\}$	—
2	—	$\{u_1\}$	$\{u_1,u_2\}$
3	—	$\{u_1\}$	$\{u_1,u_2\}$
4	—	$\{u_4\}$	$\{u_2,u_4\}$

表 6 算法 2 中的计算过程 $C(i,j)$

i	j		
	0	1	2
0	—	—	—
1	—	$\{u_1\}$	—
2	—	$\{u_2\}$	$\{u_1,u_2\}$
3	—	$\{u_3\}$	$\{u_1,u_3\}\{u_2,u_3\}$
4	—	$\{u_4\}$	$\{u_1,u_4\}\{u_2,u_4\}\{u_3,u_4\}$

算法复杂度分析: 算法先对所有事件排序, 时间复杂度为 $O(|E|\log|E|)$. 接着为每个事件选择用户, 计算所有子问题的时间复杂度为 $O(|U|!)$. 因此, 算法的时间复杂度为 $O(|E||U|!+|E|\log|E|)$.

基于收益预测的快速算法. 当 $i>j$, 在选择第 i 个用户时, 需要计算 $C(i,j)$ 中所有用户组合的收益, 计算 $f(i,j)$ 的复杂度较高, 为 $O\left((j-1)C_{i-1}^{j-1}\right)$. 为了提高计算效率, 提出一种基于收益预测的快速算法, 可以减少 $C(i,j)$ 中用户组合收益的计算. 收益预测的思路是: 对于事件 e, 新加入的用户 u 的事件偏好值为 $\phi(u_i,e)$, 假设其社交偏好值为 $\max_{u_i\in U}\psi(u,u_i)$, 依此来计算用户组合收益的预测值, 易知该预测值为真实收益值的上界. 然后, 将 $C(i,j)$ 中的所有用户组合按照预测收益值非增排序. 如果 $f(i-1,j)$ 的值大于所有预测收益的最大值, 则 $f(i,j)=f(i-1,j)$, 即不必选择第 i 个用户; 否则, 按照预测收益值非增的顺序, 依次计算用户组合的真实收益值. 在计算过程中, 如果当前已

获得的最大真实收益值大于下一个用户组合的预测收益值, 那么无需再计算, 因为剩下所有用户组合都必定不会获得更大收益.

以例 2 中为事件 e_2 选择用户的计算过程为例 (α 的值设为 0.5), 在计算 $f(3,2)$ 时, 对于新加入的用户 u_3, 事件偏好为 $\phi(u_3,e_2)=0.2$, 假设社交偏好为 $\max_{u_i\in U}\psi(u_3,u_i)=0.9$, 计算 $C(3,2)$ 中的所有用户组合的预测收益值. 将 $C(3,2)$ 中的所有用户组合按照预测收益值非增排序, 获得用户组合偏序 $(\{u_1,u_3\},0.65)$、$(\{u_2,u_3\},0.65)$. 已知 $f(2,2)=0.65$, 由于偏序中的第 1 个用户组合 $\{u_1,u_3\}$ 的预测收益值为 $0.65=f(2,2)$, 因此无需计算 $C(3,2)$ 中的所有用户组合的真实收益值, 那么 $f(3,2)=f(2,2)=0.65$, $S(3,2)=S(2,2)=\{u_1,u_2\}$; 在计算 $f(4,2)$ 时, 对于新加入的用户 u_4, 事件偏好为 $\phi(u_4,e_2)=0.9$, 假设社交偏好为 $\max_{u_i\in U}\psi(u_4,u_i)=0.9$, 计算 $C(4,2)$ 中的所有用户组合的预测收益值. 将 $C(4,2)$ 中的所有用户组合按照预测收益值非增排序, 获得用户组合偏序 $(\{u_1,u_4\},1.0)$、$(\{u_2,u_4\},1.0)$、$(\{u_3,u_4\},1.0)$. 已知 $f(2,2)=0.65<1.0$, 需要计算第 1 个用户组合 $\{u_1,u_4\}$ 的真实收益值, 即 $0.5\times0.2+0.5\times0.9+0.5\times0.6=0.85>0.65$, 因此当前最大真实收益值为 0.85, 由于偏序中的第 2 个用户组合 $\{u_2,u_3\}$ 的预测收益值为 $1.0>0.85$, 因此需要继续计算第 2 个用户组合 $\{u_2,u_4\}$ 的真实收益值, 即 $0.5\times0.2+0.5\times0.9+0.5\times0.9=1.0>0.85$, 当前最大真实收益值更新为 1.0. 由于偏序中的第 3 个用户组合 $\{u_3,u_4\}$ 的预测收益值为 1.0, 不大于当前最大真实收益, 因此无需再计算该用户组合的真实收益值. 在这个过程中, 基于收益预测的快速算法计算了 7 个用户组合 $\{u_1\}$、$\{u_2\}$、$\{u_3\}$、$\{u_4\}$、$\{u_1,u_2\}$、$\{u_1,u_4\}$、$\{u_2,u_4\}$ 的收益值, 对比表 6, 算法 2 中的 $C(i,j)$ 的计算过程, 少计算了 3 个用户组合 $\{u_1,u_3\}$、$\{u_2,u_3\}$、$\{u_3,u_4\}$ 的收益值.

3.3 事件导向的近似最优用户选择算法

事件导向的动态规划算法 (算法 2) 依次为每个事件 e_j 从可以参加该事件的用户集合中选择 δ_j 个用户, 使得该事件的收益 $\theta(e_j)$ 最大. 算法 2 在递归求解过程中, 当 $i>j$ 时, 将在前 i 个用户中选择 j 个用户的子问题转化为在前 $i-1$ 个用户中选择 $j-1$ 个用户的所有组合中, 哪个组合加上第 i 个用户后获得的收益最大, 复杂度为 $\mathrm{O}\left((j-1)C_{i-1}^{j-1}\right)$. 本节提出事件导向的近似最优用户选择算法, 该算法通过一种启发式规则为每个事件选择用户, 虽然不能确保选择的用户能最大化事件的收益, 但该算法能获得接近最大收益的解, 同时, 其计算复杂度低于算法 2. 具体来说, 当 $i>j$ 时, 在选择第 i 个用户的情况下, 把在前 i 个用户中选择 j 个用户的子问题转化为在前 k 个用户中选择 $j-1$ 个用户 $f(k,j-1)$, $j-1\leqslant k<i$ 这 $i-j+1$ 个子问题的基础上, 哪个子问题的解加上第 i 个用户后获得的收益最大, 那么 $f(i,j)$ 用公式 (5) 计算.

$$f(i,j)=\max_{j-1\leqslant k<i}\left\{f(k,j-1)+(1-\alpha)\phi(u_i,e)+\alpha\sum_{u_x\in S(k,j-1)}\psi(u_i,u_x)\right\} \tag{5}$$

该近似最优用户选择算法基于 $f(k,j-1)$, $j-1\leqslant k<i$ 这 $i-j+1$ 个子问题的收益来计算 $f(i,j)$ 的解, 考虑的用户组合数量远远小于算法 2 中 $C(i,j)$ 包含的用户组合数量, 虽然有可能遗漏 $C(i,j)$ 中具有较大收益的用户组合, 但是计算复杂度大幅降低. 该算法的伪代码见算法 3.

算法 3. 事件导向的近似最优用户选择算法.

输入: 事件集 E, 用户集 U;
输出: 一个可行规划 M.

1. $PE\leftarrow$ 对所有事件根据其参与人数上界进行降序排序;
2. **For** (每一个用户 u)
3. $\Psi(u)\leftarrow\sum_{u_i\in U}\psi(u,u_i)$;
4. **End For**

5. **For** (PE 中的每个事件 e)
6. $userList(e) \leftarrow getValidUserList(e)$;
7. **If** ($|userList(e)| < \gamma$)
8. **continue;**
9. **End If**
10. **If** ($|userList(e)| \geqslant \gamma$ and $|userList(e) \leqslant \delta|$)
11. $M^{-1}(e) \leftarrow userList(e)$;
12. **End If**
13. **If** ($|userList(e)| > \delta$)
14. 将 $userList(e)$ 中所有用户按照社交偏好总和 $\Psi(u)$ 非增排序;
15. **For** ($j = 1$ to δ)
16. **For** ($i = j$ to $|userList(e)|$)
17. **If** ($i == j$)
18. $f(i,j) \leftarrow f(i-1,j-1) + (1-\alpha)\phi(u_i,e) + \alpha \sum_{u_k \in S(i-1,j-1)} \psi(u_i,u_k)$;
19. $S(i,j) \leftarrow S(i-1,j-1) \cup \{u_i\}$;
20. **Else**
21. $f(i,j) = \max_{j-1 \leqslant k < i} \left\{ f(k,j-1) + (1-\alpha)\phi(u_i,e) + \alpha \sum_{u_x \in S(k,j-1)} \psi(u_i,u_x) \right\}$;
22. $f(i,j) \leftarrow \max\{f(i,j), f(i-1,j)\}$;
23. 相应更新 $S(i,j)$
24. **End If**
25. **End For**
26. **End For**
27. $M^{-1}(e) \leftarrow S(|userList(e)|, \delta)$;
28. **End If**
29. **For** ($M^{-1}(e)$ 中的每个用户 u)
30. $M(u) \leftarrow M(u) \cup \{e\}$;
31. **End For**
32. **End For**
33. **Return** $M \leftarrow \cup_u \{M(u)\}$;

例 3. 表 2 给出了 4 个用户的事件偏好值和 3 个事件的举办时间, 用户的社交偏好值见表 3, 表中用户后括号内的数值表示该用户的参加事件数上界, 事件后括号内的数值表示该事件的参与人数下界和上界. $f(i,j)$ 和 $S(i,j)$ 的计算过程见表 7、表 8. 以为事件 e_2 选择用户的计算过程为例 (α 的值设为 0.5), 算法 3 在计算 $f(4,2)$ 时, 新加入的用户为 u_4, 其事件偏好为 $\phi(u_4,e_2) = 0.9$. 用户组合 $S(4,2)$ 的计算过程为: 将用户 u_4 加入 3 个用户组合 $S(1,1)$、$S(2,1)$、$S(3,1)$ 中, 由于 $S(1,1)$、$S(2,1)$、$S(3,1)$ 均为 $\{u_1\}$, 那么 $S(4,2) = \{u_1,u_4\}$. 用户组合 $S(4,2)$ 的收益值为 $0.5\times0.2 + 0.5\times0.9 + 0.5\times0.6 = 0.85 > f(3,2)$, 因此 $f(4,2) = 0.85$. 然而, 算法 2 在计算子问题 $f(4,2)$ 时, 获得的收益值为 1.0, 比算法 3 获得的收益值大, 但是, 计算的用户组合数比算法 3 多, 包括了 $\{u_1,u_4\}$、$\{u_2,u_4\}$、$\{u_3,u_4\}$.

算法复杂度分析: 该算法先对所有事件排序, 时间复杂度为 $O(|E|\log|E|)$. 接着为每个事件选择用户, 子问题的数量为 $O(|U|)$, 计算每个子问题的时间复杂度为 $O(|U|)$. 因此算法的时间复杂度为 $O(|E||U|^2 + |E|\log|E|)$.

表 7 算法 3 中的计算过程 $f(i,j)$

i	j		
	0	1	2
0	—	—	—
1	—	0.1	—
2	—	0.1	0.65
3	—	0.1	0.65
4	—	0.45	0.85

表 8 算法 3 中的计算过程 $S(i,j)$

i	j		
	0	1	2
0	—	—	—
1	—	$\{u_1\}$	—
2	—	$\{u_1\}$	$\{u_1,u_2\}$
3	—	$\{u_1\}$	$\{u_1,u_2\}$
4	—	$\{u_4\}$	$\{u_1,u_4\}$

4 实验及分析

本节评估本文提出的算法在不同数据集上、不同参数设定下的计算效率和求得规划的整体收益. 具体包括数据集和实验环境介绍、基准算法介绍和实验对比结果.

4.1 数据集及实验环境

本文算法用 Java 及 JDK1.8.0_201 实现. 实验在一台服务器上运行, 该服务器的操作系统为 Ubuntu 16.04.3 LTS, 处理器为 Intel(R) Xeon(R) CPU E7-4830 v4@2.00 GHz, 内存容量为 2 TB, 硬盘容量为 3 TB.

本实验使用的数据由从 Meetup 平台 (https://secure.meetup.com/meetup_api) 爬取的数据和人工构造的数据组成. 爬取的数据包含 2019 年发布在 Meetup 平台上的 4 959 个事件和参加这些事件的 42 428 个用户, 涵盖了 57 个国家. 其中, 每个用户有描述其兴趣的文本信息, 每个事件有描述其内容的文本信息. 用户的事件偏好值为用户的兴趣文本与事件的内容文本之间的相似度, 两个用户的社交偏好值为这两个用户曾经参加过的事件集合的 Jaccard 系数. 人工构造的数据包括每个事件的参与人数下界、事件的参与人数上界以及用户的参加事件数上界.

为了测试算法在不同数据集上的性能, 本实验使用 5 个真实数据集. 其中 3 个数据集是从爬取的数据中提取了澳大利亚、英国、美国 3 个国家的事件和用户, 形成了 3 个不同的数据集. 数据集 CA 和 NYC 来自文献 [29], 是加州和纽约地区 2012 年的 Meetup 数据. 表 9 中总结了数据集中的各参数, 其中, 事件数量为 m, 用户数量为 n, 用户的参加事件数上界为 c, 事件的参与人数上界为 δ, 事件的参与人数下界设置为参与人数上界的 1/2, α 的值设为 0.5, 冲突率表示冲突事件占事件总数的百分比.

为了测试算法在不同参数设定下的性能, 本实验从爬取的数据中提取了不同规模、不同参数的数据集, 如表 10 所示. 在评估一个参数的不同取值对算法性能的影响时, 其他参数均取默认值 (表 10 中的粗体参数值).

表 9 数据集

数据集	m	n	δ 的范围	c 的均值	冲突率
澳大利亚	240	200	[1, 5]	50	0.30
英国	365	200	[1, 5]	50	0.25
美国	1 000	200	[1, 5]	50	0.25
CA	200	100	[1, 10]	40	0.15
NYC	200	150	[1, 5]	40	0.25

表 10 参数设定

参数名称	参数值
冲突率	**0.25**
c 的均值	10, 25, **50**, 75, 100
δ 的范围	**[1, 5]**, [1, 10], [1, 50], [1, 100], [1, 150]
m	300, 500, **1 000**, 2 000, 4 000
n	100, **200**, 1 000, 10 000, 40 000

本文研究的 CCP 规划问题与已有文献中的规划问题不完全相同, 因此现有算法并不能直接用来求解 CCP 规划问题. 本实验将文献 [2] 中的 PCADG 算法进行了修改和扩展, 作为实验中的两个基准算法 Baseline_1 和 Baseline_2. 其他相关文献中的算法不能通过简单的修改和扩展来求解 CCP 规划问题. 下面详述算法 PCADG 的修改部分. 算法 PCADG 中, 每个用户只能参与 1 个事件, 没有解决事件冲突的部分. 修改的部分包括: 当通过贪心策略选出一对用户和事件时, 首先检查该事件是否和该用户已经安排的事件有冲突, 如有冲突, 则通过贪心策略继续选择下一对用户和事件, 如没有冲突, 则将该事件安排给该用户. 如果该用户被安排的事件数量达到上界, 在接下

来的贪心策略中将不再考虑该用户.

本实验评估的 10 个算法具体描述如下.

• Random-E: 在满足 CCP 问题的约束条件下, 为每个用户随机安排事件.

• Random-U: 在满足 CCP 问题的约束条件下, 为每个事件随机选择用户.

• MinCostFlow-GEACC[3]: 该算法没有考虑用户之间的社交偏好.

• Greedy-based Alg[4]: 将文献 [4] 中的问题的旅行预算设为无穷大, 算法没有考虑用户之间的社交偏好.

• Baseline$_1$: 基于用户-事件对的贪心算法+公式 (1).

• Baseline$_2$: 基于用户-事件对的贪心算法+公式 (2).

• Event-G$_1$: 事件导向的贪心用户选择算法+公式 (1).

• Event-G$_2$: 事件导向的贪心用户选择算法+公式 (2).

• Event-DP: 事件导向的动态规划算法.

• Event-DP*: 事件导向的近似最佳用户选择算法.

4.2 不同数据集上的实验结果

表 11 和表 12 显示以上 10 个算法在 5 个真实数据集上求解 CCP 规划问题的时间开销、获得规划的整体收益、事件偏好、社交偏好. 本节对比的算法从算法思想上可分为: 基于贪心思想的算法 Baseline$_1$、Baseline$_2$、Event-G$_1$、Event-G$_2$、MinCostFlow-GEACC、Greedy-based、基于动态规划思想的算法 Event-DP 和 Event-DP*和随机选择算法 Random-U、Random-E.

表 11 求解 CCP 规划问题的时间开销和规划方案的整体收益

算法	澳大利亚				英国				美国			
	计算时间 (s)	整体收益	事件偏好	社交偏好	计算时间 (s)	整体收益	事件偏好	社交偏好	计算时间 (s)	整体收益	事件偏好	社交偏好
Baseline$_1$	0.36	296.11	**181.87**	114.24	0.46	575.08	**350.26**	224.82	1.40	1 213.97	**690.80**	522.17
Baseline$_2$	41.59	483.98	163.90	320.08	67.90	638.59	343.42	295.17	281.33	1 513.27	663.38	849.89
Event-G$_1$	**0.07**	293.35	180.49	112.86	**0.09**	566.39	349.41	216.98	**0.18**	1 207.52	685.83	521.69
Event-G$_2$	0.42	479.31	167.21	312.10	0.65	632.14	343.46	288.66	1.41	1 492.80	664.03	828.77
Event-DP	21 323	**537.87**	143.07	**394.80**	28 416	**824.56**	244.94	**579.62**	63 084	**1 879.47**	508.87	**1 370.60**
Event-DP*	0.89	531.58	143.13	388.45	1.34	751.32	280.42	470.90	2.64	1 667.76	568.35	1 099.41
MinCostFlow-GEACC	1.56	248.30	174.97	73.33	3.58	494.02	342.42	151.60	26.50	1 026.00	659.17	366.83
Greedy-based	0.049	189.52	162.71	26.81	0.077	350.27	292.12	58.15	0.19	790.55	624.68	165.87
Random-U	0.037	94.19	72.53	21.66	0.054	113.39	99.46	13.93	0.09	234.92	215.90	19.02
Random-E	0.01	117.53	92.51	25.02	0.013	151.61	131.72	19.88	0.05	312.20	281.86	30.34

在整体收益方面, 基于动态规划思想的算法明显优于基于贪心思想的算法和随机选择算法, 这是因为基于动态规划思想的算法在为事件选择用户时, 在多种可能用户组合的情况下选择最优, 而基于贪心思想的算法只基于已经选定的用户来计算下一个待选择的用户产生的收益, 考虑的用户组合情况不如基于动态规划思想的算法全面, 容易陷入局部最优, 随机选择算法在随机选择过程中没有考虑社交偏好和事件偏好, 因此整体收益最低. 同时, 算法 Event-DP 所获得的整体收益高于算法 Event-DP*. 这是因为算法 Event-DP 为每个事件选择了收益值最大的用户组合, 而算法 Event-DP*是基于部分用户组合情况进行近似最优选择. 在基于贪心思想的算法中, 算法 Baseline$_2$ 和 Event-G$_2$ 获得的整体收益高于算法 Baseline$_1$ 和 Event-G$_1$. 这是因为公式 (2) 在选择用户时不仅考虑了待选择用户与已选择用户之间的社交偏好, 还考虑了待选择用户与未来可能被选择用户之间的社交偏好, 而公式 (1) 只考虑了前一个因素. 在时间开销方面, 算法 Baseline$_1$、Event-G$_1$、Event-G$_2$、Event-DP*、MinCostFlow-GEACC、Greedy-based、Random-U、Random-E 显著优于算法 Baseline$_2$ 和 Event-DP. 这是因为算法 Event-DP 为了获得每个事件的最大收益值, 计算了大量的用户组合, 算法 Baseline$_2$ 在每次选择用户时需要根据公式 (2) 计算

所有“用户-事件”带来的收益. 在事件偏好方面, 基于贪心思想的算法和基于动态规划思想的算法明显优于随机选择算法, 这是因为随机选择算法在求解过程中没有考虑事件偏好. 基于贪心思想的算法获得的事件偏好略高于基于动态规划思想的算法, 这是因为基于贪心思想的算法容易选择事件偏好较高的规划陷入局部最优, 导致整体收益和社交偏好低于基于动态规划思想的算法. 在社交偏好方面, 算法 Baseline$_1$、Baseline$_2$、Event-G$_1$、Event-G$_2$、Event-DP、Event-DP*明显优于算法 MinCostFlow-GEACC、Greedy-based、Random-U、Random-E, 这是因为算法 MinCostFlow-GEACC、Greedy-based、Random-U、Random-E 在事件选择用户的过程中没有考虑社交偏好的因素. 同时, 算法 Event-DP 获得的社交偏好是最大的, 这是因为算法 Event-DP 为了获得每个事件的最大收益值, 计算了大量的用户组合. 综合整体收益和时间开销, 算法 Event-G$_2$ 和 Event-DP*能在较短时间获得较大整体收益值. 虽然算法 Event-DP 在所有算法中获得的整体收益最大, 但是其时间开销较大, 而算法 Event-DP*能获得与 Event-DP 相近的整体收益, 并且其时间开销小. 因此, 在第 4.3 节的大规模数据集实验上, 将把算法 Event-DP*作为基于动态规划思想算法的代表.

表 12　求解 CCP 规划问题的时间开销和规划方案的整体收益

算法	CA				NYC			
	计算时间 (s)	整体收益	事件偏好	社交偏好	计算时间 (s)	整体收益	事件偏好	社交偏好
Baseline$_1$	0.27	485.84	297.5	188.34	0.35	206.57	166.0	40.57
Baseline$_2$	3.42	685.68	304.0	381.68	7.22	212.36	175.0	37.36
Event-G$_1$	**0.09**	479.24	275.0	204.24	**0.17**	208.20	171.0	37.20
Event-G$_2$	0.26	631.80	282.5	349.30	0.39	213.76	175.5	38.26
Event-DP	—	—	—	—	4979.39	**269.17**	128.0	**141.17**
Event-DP*	1.01	**733.36**	220.5	**512.86**	0.51	213.81	175.5	38.31
MinCostFlow-GEACC	0.76	395.85	**306.0**	89.85	0.52	202.72	177.0	25.72
Greedy-based	0.044	380.80	285.5	95.30	0.025	201.07	**176.5**	24.57
Random-U	0.023	55.55	25.0	30.55	0.025	8.98	6.0	2.98
Random-E	0.011	70.68	30.0	40.68	0.007	10.87	7.0	3.87

4.3　不同参数设定下的实验结果

本节分析并比较本文提出的算法和 2 个基准算法在不同参数设定下的时间开销和整体收益, 实验涉及的参数见前文表 10. 图 3 所示为算法的整体收益和时间开销随用户数量 n 的变化情况.

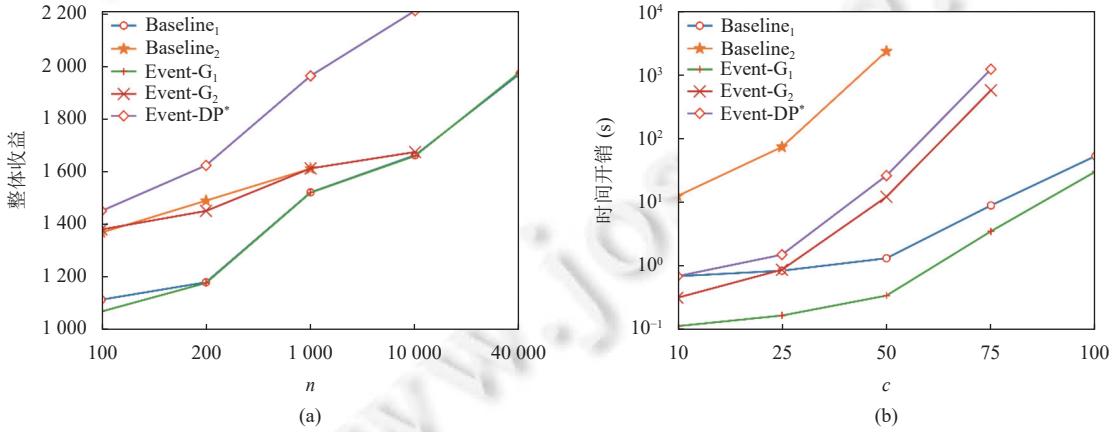

图 3　整体收益和时间开销随用户数量的变化

从图 3 可以看出, 所有算法的整体收益和时间开销随着用户数量的增加而增加. 该现象显然是合理的, 因为随着用户数量的增加, 对于每个事件而言, 可挑选的用户数量就会相应增多, 整体收益和时间开销必然增加. 算法 Event-DP*所获得的整体收益是最高的; 算法 Baseline$_1$、算法 Baseline$_2$ 所获得的整体收益分别和算法 Event-G$_1$、

算法 Event-G_2 相差不大. 对于时间开销, 算法 Event-G_1、算法 Event-G_2 的时间开销分别小于算法 Baseline$_1$、算法 Baseline$_2$, 算法 Event-G_2 的时间开销小于算法 Event-DP*, 同时算法 Event-DP*的时间开销小于算法 Baseline$_2$. 由此可见算法 Event-DP*在整体收益和时间开销上均优于算法 Baseline$_2$, 而算法 Event-G_1、算法 Event-G_2 在时间开销上分别优于算法 Baseline$_1$、算法 Baseline$_2$, 因此佐证了本文提出的算法 Event-DP*、算法 Event-G_1、算法 Event-G_2 的合理性.

图 4 所示为整体收益和时间开销随事件数量 m 的变化情况, 可以看出, 这几种算法的整体收益和时间开销随着事件数量的增加而增加. 该现象显然是合理的, 因为随着事件数量的增加, 需要进行规划的事件数量就会增多, 整体收益和时间开销必然增加. 算法 Event-DP*所获得的整体收益是最高的; 算法 Baseline$_1$、算法 Baseline$_2$ 所获得的整体收益分别和算法 Event-G_1、算法 Event-G_2 相差不大. 对于时间开销, 算法 Event-G_1、算法 Event-G_2 的时间开销分别小于算法 Baseline$_1$、算法 Baseline$_2$, 算法 Event-G_2 的时间开销小于算法 Event-DP*. 当事件数量增加到一定程度时, 算法 Baseline$_1$ 的时间开销会接近甚至超过算法 Event-DP*的时间开销. 由此可见算法 Event-DP*在整体收益和时间开销上均优于算法 Baseline$_2$, 而算法 Event-G_1、算法 Event-G_2 在时间开销上分别优于算法 Baseline$_1$、算法 Baseline$_2$, 因此佐证了本文提出的算法 Event-DP*、算法 Event-G_1、算法 Event-G_2 的合理性.

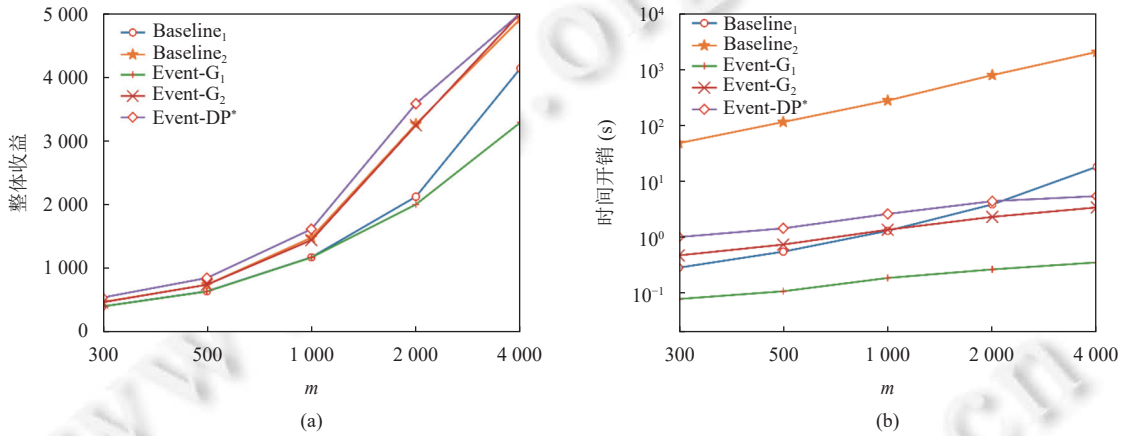

图 4 整体收益和时间开销随事件数量的变化

图 5 所示为整体收益和时间开销随用户的参加事件数上界 c 的变化情况, 可以看出, 这几种算法的整体收益随着用户参加事件数上界的增加而增加. 该现象显然是合理的, 因为随着用户参加事件数上界的增加, 对于每一个用户来说, 可以参加的事件数量变多, 整体收益必然增加; 算法 Baseline$_2$ 在用户参加事件数上界较小时时间开销是最大的, 这是因为算法 Baseline$_2$ 使用公式 (2) 计算用户-事件对的收益, 然后将收益最大的用户-事件加入到规划中. 公式 (2) 考虑了目标用户与未来可能被选择的用户的社交偏好, 当有用户达到参加事件数上界时, 该用户在未来就不可能被选择, 因此, 之前计算的用户-事件对的收益需要更新, 排除该用户的影响. 在用户参加事件数上界较小时, 用户达到参加事件数上界的情况较多, 因此算法 Baseline$_2$ 需要重新计算多对用户-事件的收益, 从而时间开销较大; 其他 4 种算法的时间开销随着用户参加事件数上界的增加而增加, 该现象显然是合理的, 因为对于每个事件来说, 可挑选的用户数量增加, 规划的时间开销也随之增加. 以上 5 种算法的时间开销最后都随着用户参加事件数上界的增加而趋于稳定, 这是因为事件的参与人数上界总和是有限的. 算法 Event-DP*所获得的整体收益是最高的; 算法 Baseline$_1$、算法 Baseline$_2$ 所获得的整体收益分别和算法 Event-G_1、算法 Event-G_2 相差不大. 对于时间开销, 算法 Event-G_1、算法 Event-G_2 的时间开销分别小于算法 Baseline$_1$、算法 Baseline$_2$, 算法 Event-G_2 的时间开销小于算法 Event-DP*. 由此可见算法 Event-DP*在整体收益和时间开销上均优于算法 Baseline$_2$, 而算法 Event-G_1、算法 Event-G_2 在时间开销上分别优于算法 Baseline$_1$ 和算法 Baseline$_2$, 因此佐证了本文提出的算法 Event-DP*、算法 Event-G_1、算法 Event-G_2 的合理性.

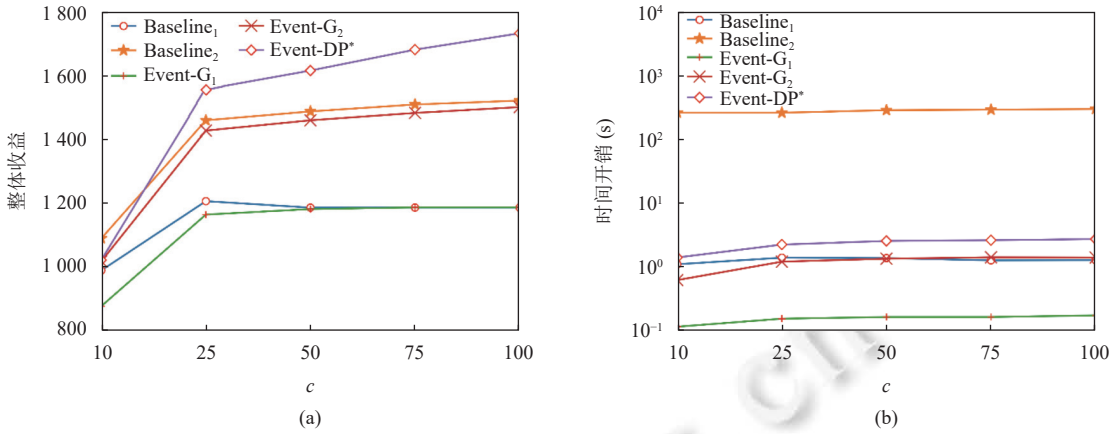

图 5　整体收益和时间开销随用户的参加事件数上界的变化

图 6 所示为整体收益和时间开销随事件的参与人数上界 δ 的变化情况, 可以看出, 这几种算法的整体收益和时间开销随着用户参加事件数上界先增加最后趋于稳定. 该现象显然是合理的, 因为随着事件的参与人数上界的增加, 对于每一个事件来说, 可以参加该事件的用户数量变多, 整体收益和时间开销必然增加, 而当事件的参与人数上界增加到一定程度时, 整体收益收到用户的参加事件数的限制而趋于稳定, 从而时间开销也趋于稳定. 算法 Event-DP*所获得的整体收益和算法 Baseline$_2$ 的整体收益相差不大, 但是时间开销比算法 Baseline$_2$ 小; 算法 Baseline$_1$、算法 Baseline$_2$ 所获得的整体收益分别和算法 Event-G$_1$、算法 Event-G$_2$ 相差不大. 对于时间开销, 算法 Event-G$_1$、算法 Event-G$_2$ 的时间开销分别小于算法 Baseline$_1$、算法 Baseline$_2$. 由此可见算法 Event-DP*在时间开销上优于算法 Baseline$_2$, 而算法 Event-G$_1$、算法 Event-G$_2$ 在时间开销上分别优于算法 Baseline$_1$ 和算法 Baseline$_2$, 因此佐证了本文提出的算法 Event-DP*、算法 Event-G$_1$、算法 Event-G$_2$ 的合理性.

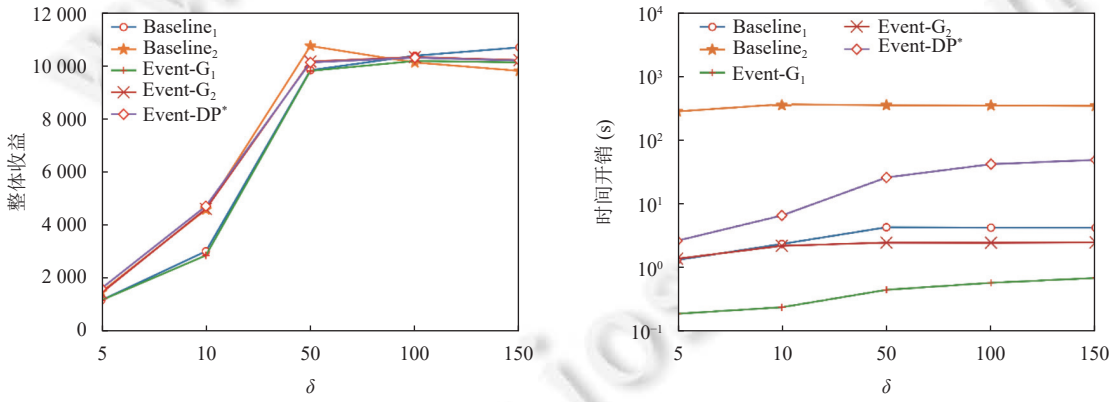

图 6　整体收益和时间开销随事件的参与人数上界的变化

表 13 为算法 2 中基于收益预测的快速计算技术在降低计算开销方面的效果. Event-DP 基础版是算法 2 去掉快速计算技术的版本, Event-DP 快速版是算法 2 加上快速计算技术的版本. 在澳大利亚数据集上提取了具有不同用户数量$|U|$ ($|U|$ = 50, 100, 150) 的数据集, 其他参数使用表 10 中的默认值. 表 13 对比了 Event-DP 基础版和 Event-DP 快速版的时间开销和计算社交偏好的数量随用户数量的变化情况, 可以看出, Event-DP 基础版的时间开销和计算社交偏好数量随用户数量增长而迅速增加, 而 Event-DP 快速版的时间开销和计算社交偏好数量相对增长较慢, 并显著低于 Event-DP 基础版.

表 13 时间开销和计算社交偏好的数量随用户数量的变化

算法	50		100		150	
	计算时间 (s)	社交偏好数量	计算时间 (s)	社交偏好数量	计算时间 (s)	社交偏好数量
Event-DP基础版	42.11	4.37×10^{8}	2252.66	1.47×10^{10}	21907.08	1.35×10^{11}
Event-DP快速版	16.84	1.64×10^{7}	727.30	5.24×10^{7}	6681.30	2.69×10^{8}

4.4 实验总结

第 4.2 节和第 4.3 节在不同参数不同场景下对 6 个算法的性能进行了比较, 包括不同的用户-事件数量比例, 不同的事件冲突率, 不同的用户数量和事件数量, 以及不同的用户参加事件数上界和事件参与人数上界. 实验结果表明: 算法 Event-DP 得出的规划的整体收益最高, 适用于中小规模数据; 算法 Event-DP*得出的规划的整体收益仅次于算法 Event-DP, 能够在较大规模数据 (10^4 数量级用户, 10^3 数量级事件) 上快速得出结果; 算法 Event-G_2 得出的规划的整体收益低于算法 Event-DP, 但其在大规模数据上的可扩展性最好.

5 结论及展望

本文重点研究社交网络中的事件规划这一重要热点问题, 同时从冲突、容量、偏好 3 个角度进行规划方案计算, 提出求解 CCP 事件规划问题的高效算法, 提高了社交网络中事件规划方案的质量. 为了验证提出算法的有效性, 通过实验对比了两个基准算法. 实验结果显示提出的算法在大规模数据上、在不同参数设定下能够快速计算出高收益的规划方案. 未来工作中, 可以考虑用流网络建模 CCP 事件规划问题, 提供能获得更大整体收益高性能算法.

References:

[1] She JY, Tong YX, Chen L. Utility-aware social event-participant planning. In: Proc. of the 2015 ACM SIGMOD Int'l Conf. on Management of Data. Melbourne: ACM Press, 2015. 1629−1643. [doi: 10.1145/2723372.2749446]

[2] Li KQ, Lu W, Bhagat S, Lakshmanan LVS, Yu C. On social event organization. In: Proc. of the 20th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2014. 1206−1215. [doi: 10.1145/2623330.2623724]

[3] She JY, Tong YX, Chen L, Cao CC. Conflict-aware event-participant arrangement. In: Proc. of the 31st IEEE Int'l Conf. on Data Engineering. Seoul: IEEE, 2015. 735–746. [doi: 10.1109/ICDE.2015.7113329]

[4] Cheng YR, Yuan Y, Chen L, Giraud-Carrier C, Wang GR. Complex event-participant planning and its incremental variant. In: Proc. of the 33rd IEEE Int'l Conf. on Data Engineering. San Diego: IEEE, 2017. 859−870. [doi: 10.1109/ICDE.2017.135]

[5] Kou FF, Zhou ZM, Cheng H, Du JP, Shi YX, Xu P. Interaction-aware arrangement for event-based social networks. In: Proc. of the 35th IEEE Int'l Conf. on Data Engineering. Macao: IEEE, 2019. 1638–1641. [doi: 10.1109/ICDE.2019.00161]

[6] She JY, Tong YX, Chen L, Song TS. Feedback-aware social event-participant arrangement. In: Proc. of the 2017 ACM Int'l Conf. on Management of Data. Chicago: ACM Press, 2017. 851–865. [doi: 10.1145/3035918.3064020]

[7] She JY, Tong YX, Chen L, Cao CC. Conflict-aware event-participant arrangement and its variant for online setting. IEEE Trans. on Knowledge and Data Engineering, 2016, 28(9): 2281–2295. [doi: 10.1109/TKDE.2016.2565468]

[8] Cheng YR, Yuan Y, Chen L, Giraud-Carrier C, Wang GR, Li BY. Event-participant and incremental planning over event-based social networks. IEEE Trans. on Knowledge and Data Engineering, 2021, 33(2): 474–488. [doi: 10.1109/TKDE.2019.2931906]

[9] Xin JC, Li M, Xu WZH, Cai YZ, Lu MH, Wang ZQ. Efficient complex social event-participant planning based on heuristic dynamic programming. In: Proc. of the 23rd Int'l Conf. on Database Systems for Advanced Applications. Gold Coast: Springer, 2018. 264–279. [doi: 10.1007/978-3-319-91458-9_16]

[10] Cheng YR, Wang GR, Li BY, Yuan Y. Bilateral preference stable planning over event based social networks. Ruan Jian Xue Bao/Journal of Software, 2019, 30(3): 573−588 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5682.htm [doi: 10.13328/j.cnki.jos.005682]

[11] Sun HL, Zhang S, Huang JB, He L, Jiang XL, Duan ZT. A personalized event-participant arrangement framework based on user interests in social network. Computer Networks, 2020, 183: 107607. [doi: 10.1016/j.comnet.2020.107607]

[12] Ding LL, Zhang HL, Chen Z, Song BY. Utility-time social event planning on EBSN. In: Proc. of the 20th IEEE Int'l Conf. on Mobile Data Management. Hong Kong: IEEE, 2019. 182–190. [doi: 10.1109/MDM.2019.00-58]

[13] Tong YX, She JY, Meng R. Bottleneck-aware arrangement over event-based social networks: The max-min approach. World Wide Web,

2016, 19(6): 1151−1177. [doi: 10.1007/s11280-015-0377-6]
[14] Bikakis N, Kalogeraki V, Gunopulos D. Social event scheduling. In: Proc. of the 34th IEEE Int'l Conf. on Data Engineering. Paris: IEEE, 2018. 1272−1275. [doi: 10.1109/ICDE.2018.00128]
[15] Gao SY, Zhang ZB, Su S, Zia MA. Multi-role event organization in social networks. Information Sciences, 2018, 447: 229−243. [doi: 10.1016/j.ins.2018.03.017]
[16] Yin ZK, Xu T, Zhu HS, Zhu C, Chen EH, Xiong H. Matching of social events and users: A two-way selection perspective. World Wide Web, 2020, 23(2): 853−871. [doi: 10.1007/s11280-019-00724-7]
[17] Burkard R, Dell'Amico M, Martello S. Assignment Problems: Revised Reprint. Philadelphia: SIAM, 2009.
[18] West DB. Introduction to Graph Theory. 2nd ed., Upper Saddle River: Prentice-Hall, 2001.
[19] Gabow HN, Sankowski P. Algorithms for weighted matching generalizations I: Bipartite graphs, *b*-matching, and unweighted *f*-factors. SIAM Journal on Computing, 2021, 50(2): 440−486. [doi: 10.1137/16M1106195]
[20] Wong RCW, Tao YF, Fu AWC, Xiao XK. On efficient spatial matching. In: Proc. of the 33rd Int'l Conf. on Very Large Data Bases. Vienna: VLDB Endowment, 2007. 579−590.
[21] Leong HU, Yiu ML, Mouratidis K, Mamoulis N. Capacity constrained assignment in spatial databases. In: Proc. of the 2008 ACM SIGMOD Int'l Conf. on Management of Data. Vancouver: ACM Press, 2008. 15−28. [doi: 10.1145/1376616.1376621]
[22] Leong HU, Mouratidis K, Yiu ML, Mamoulis N. Optimal matching between spatial datasets under capacity constraints. ACM Trans. on Database Systems, 2010, 35(2): 9. [doi: 10.1145/1735886.1735888]
[23] Sun Y, Huang J, Chen YG, Zhang R, Du XY. Location selection for utility maximization with capacity constraints. In: Proc. of the 21st ACM Int'l Conf. on Information and Knowledge Management. Maui: ACM Press, 2012. 2154−2158. [doi: 10.1145/2396761.2398592]
[24] Karp RM, Vazirani UV, Vazirani VV. An optimal algorithm for on-line bipartite matching. In: Proc. of the 22nd Annual ACM Symp. on Theory of Computing. Baltimore: ACM Press, 1990. 352−358. [doi: 10.1145/100216.100262]
[25] Mehta A. Online matching and ad allocation. Foundations and Trends® in Theoretical Computer Science, 2013, 8(4): 265−368. [doi: 10.1561/0400000057]
[26] Kesselheim T, Radke K, Tönnis A, Vöcking B. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In: Proc. of the 21st European Symp. on Algorithms. Sophia Antipolis: Springer, 2013. 589−600. [doi: 10.1007/978-3-642-40450-4_50]
[27] Ting HF, Xiang XZ. Near optimal algorithms for online maximum edge-weighted *b*-matching and two-sided vertex-weighted *b*-matching. Theoretical Computer Science, 2015, 607: 247−256. [doi: 10.1016/j.tcs.2015.05.032]
[28] Tong YX, She JY, Ding BL, Wang LB, Chen L. Online mobile micro-task allocation in spatial crowdsourcing. In: Proc. of the 32nd IEEE Int'l Conf. on Data Engineering. Helsinki: IEEE, 2016. 49−60. [doi: 10.1109/ICDE.2016.7498228]
[29] Pham TAN, Li XT, Cong C, Zhang ZJ. A general graph-based model for recommendation in event-based social networks. In: Proc. of the 31st Int'l Conf. on Data Engineering. Seoul: IEEE, 2015. 567−578. [doi: 10.1109/ICDE.2015.7113315]

附中文参考文献:

[10] 成雨蓉, 王国仁, 李博扬, 袁野. 基于事件的社交网络上的双边偏好稳态规划. 软件学报, 2019, 30(3): 573−588. http://www.jos.org.cn/1000-9825/5682.htm [doi: 10.13328/j.cnki.jos.005682]

吴定明(1982−), 女, 博士, 副教授, 主要研究领域为数据管理, 包括图数据, 时空数据, 查询处理与优化技术, 数据挖掘.

陆克中(1982−), 男, 博士, 教授, 博士生导师, 主要研究领域为大数据理论, 并行与分布式计算, 无线传感器网络.

林俊杰(1998−), 男, 硕士, 主要研究领域为数据挖掘, 信息检索.

徐宇明(1999−), 男, 学士, 主要研究领域为数据挖掘, 信息检索.