doi:10.19920/j.cnki.jmsc.2021.09.006

依托平台协作配送成本分摊的有效方法研究①

饶卫振[1]，徐　丰[1]，朱庆华[2*]，尉芳芳[3]，刘从虎[3]
（1. 山东科技大学经济管理学院，青岛 266590；2. 上海交通大学安泰经济管理学院，上海 200030；
3. 上海交通大学中美物流研究院，上海 200030）

摘要：依托平台的协作配送问题，在合理时间内有效计算公平成本分摊方案至关重要. 核仁解是公认的公平分摊方案，但需要通过复杂的优化计算. 提出了一个能通过公式近似快速计算核仁解的方法，发现任意满足总体理性分摊方案 x 的 2^n-1（n 为大联盟 N 中成员数）个子联盟 S（S 为 N 的子集）的满意度 $e(S,x)$ 之和为常数，且不同 x 对应的任意子联盟 S 与互补联盟 $N\backslash S$ 的满意度之和，即 $L_S=e(S,x)+e(N\backslash S,x)$ 为常数. 基于子联盟满意度越均衡，分配方案越合理的准则，构造了分配方案 x 对应的所有子联盟满意度均衡量化函数 $f(x)=\sum[e(S,x)-0.5L_S]^2$. 显然，$f$ 越小表示子联盟满意度越均衡. 证明了存在分配方案 x^* 使 f 取到极小值，且 x^* 满足总体理性、唯一性、可加性、策略等价相对不变性、一致性、匿名性和可比性等众多分摊方案合理属性. 最后，采用文章所提方法和核仁解求解方法，计算了诸多已有文献中的成本分摊算例，求解结果表明文章提出的方法计算速度比传统核仁解求解方法快数万倍以上，与最新求解核仁解及 Shapley 值的有效算法相比也具有明显的性能优势，且求解结果与核仁解的结果平均偏差只有5%左右. 更重要的是，提出的新方法本身具有科学内涵，可以应用于任何支付可转移的合作博弈成本分摊问题.

关键词：协作配送；成本分摊方法；满意度；核仁解
中图分类号：F570.5　　**文献标识码**：A　　**文章编号**：1007－9807(2021)09－0105－22

0　引　言

由于协作配送能够显著节约配送成本、减少配送车辆尾气排放和缓解城区交通拥堵[1]，同时具有经济、环境和社会三方面效益，被称为可持续配送模式[2,3]. 因此，也受到越来越多的学者关注. 协作配送运营模式也十分符合我国坚持以“创新、协调、绿色、开放、共享”发展理念. 在国务院颁发的《物流业发展中长期规划(2014 年～2020 年)》中明确指出，鼓励物流企业采用协作联盟的方式提升物流业的集约化水平.

在实际应用中，协作配送运作中的核心问题之一是要设计一个公平的成本分摊机制，因为利益分摊不公平会直接影响协作成员的积极性，这也是当前协作配送在实际应用中遇到的难题之一[4]. 而依托平台的协作配送问题时效性要求高，不仅要求成本分摊结果的公平合理，而且还需要在合理的时间内完成成本分摊[5]，该问题属于合作博弈中的成本分摊问题，当前应用在物流协作成本分摊问题中的方法达 40 多种[6]. 但被公认的公平成本分摊方法主要包括 Shapley 值[7]和核仁解[8]，其主要原因是 Shapley 值和核仁解均

① 收稿日期：2019－11－15；修订日期：2020－09－24.
基金项目：国家自然科学基金资助重点项目(71632007)；泰山学者工程专项经费资助项目(tsqn201909111)；教育部人文社会科学基金资助项目(21YJA630075；20YJCZH175)；山东省社会科学基金资助项目(20CGLJ32).
通讯作者：朱庆华(1970—)，女，江苏省太仓人，博士，教授，博士生导师. Email：qhzhu@sjtu.edu.cn

能满足众多的合理属性[9]. 因此, Shapley 值与核仁解自提出以来得到了广泛的关注, 如当前核仁解被应用于物流协作[10], 通讯网络构建[11], 电网电量损失分摊[12], 供应链合作[13], 银行 ATM 机设备协作建设[14], 机场建设和维护[15]等十几个不同领域的成本分摊问题.

然而, 核仁解的求解极具挑战性. 因此, 有很多学者研究求解核仁解的有效方法. 当前求解的主要方法是采用线性规划方法求解. 该方法最早由 Kohlberg 在 1972 年提出[16], 其线性规划模型包含 n 个变量(n 为合作成员的个数), 但约束条件达 $2n!$ 个. 随后有较多学者研究了采用线性规划方法求解核仁解, 该类方法主要可以分为两类, 一类是采用单个线性规划模型求解, 如 Kohlberg 提出的方法, 在 1974 年 Owen 基于 Kohlberg 的研究模型提出了包含 $2^{n+1}+n$ 个变量和 4^n+1 个约束条件的线性规划模型[17]. Pureto 在 2013 年提出了一个与 Owen 具有相同复杂度的单个线性规划模型, 但具有更高的计算稳定性和效率[18]; 另一类是通过循环迭代方式求解多个线性规划模型的方式, 最先提出该类方法的学者为 Maschler, 他在 1979 年提出的模型需要求解多个复杂度为 $O(4^n)$ 的线性规划模型[8], 随后 Behringer 和 Drăgan 在 1981 年分别在各自的研究中将该类模型的复杂度降为 $O(2^n)$[19, 20]. Potters 在 1996 年提出了至多需要求解 $n-1$ 个复杂度为 $O(2^n)$ 的线性规划模型[21], Derks 和 Kuipers 通过研究得到了比 Potter 更加高效的求解模型[22]. Hallefjord 提出了减少求解核仁解线性规划模型中约束条件的有效方法[23], 随后有较多学者基于约束条件减少策略开展了进一步的研究, 如 Fromen[24] 和 Nguyen[25] 围绕减少线性规划模型约束条件数量, 提高求解核仁解模型的效率进行了研究. 另外, 在最近两年 Perea 通过抽样子联盟的方式,研究了求解核仁解的优化模型[26]. Lu 针对共享单车服务研究了成本分摊核仁解的有效求解方法[27]. Tae 针对带时间窗的车辆路径问题中的顾客分摊配送费用问题, 将核仁解线性规划模型分为主问题和子问题两部分, 并设计了有效的求解方法[28].

综上所述, 当前求解核仁解的方法几乎仍然是以求解线性规划模型为主, 且约束条件数量通常至少包含 2^n 数量级, 其中又存在多重解, 即使专业研究人员计算也经常会得到错误的结果[29, 30]. 因此, 也有不少研究者尝试非线性规划的方法[31, 32],同时部分学者研究了成本(收益)函数模糊或属于某个区间, 信息不完全等非传统成本分摊问题的核仁解求解方法[33-35]. 但遗憾的是, 当前优化方法求解规模较大的成本分摊问题核仁解, 依然具有较大的挑战性, 难以在合理时间内得到结果.

而依托平台的协作配送问题, 是典型的较大规模成本分摊问题. 传统的谈判协作配送模式, 存在效率低成本高, 难以合理制定成本分摊方法等缺点. 当前随着互联网技术、平台经济的发展, 在很多行业已证明依托平台的运作模式与传统经营模式相比会更加快捷高效. 因此我国也十分重视平台经济的发展,如国务院总理李克强在 2019 年 3 月 5 日第十三届全国人民代表大会第二次会议上作的《政府工作报告》中提到:“支持新业态新模式发展,促进平台经济、共享经济健康成长。加快在各行业各领域推进互联网 +”. 另外,从 2020 年 1 月开始,全国乃至世界范围爆发了“新型冠状病毒肺炎”疫情,紧急医疗物资的科学协调、调度和配送,都迫切需要一个能远程协调各方、集中优化调度的智能数据平台,这也必将进一步引起相关学者、企业机构和政府部门对发展平台物流模式的关注和重视.

依托平台的协作配送模式是指物流配送企业, 通过借助现代通讯和互联网等技术, 依托第三方数字化平台, 在线实现组建联盟[36]、协作调度[37]和成本分摊等协作过程. 由于第三方数字化平台需面对城区中所有物流配送企业, 在线组建协作配送联盟, 因此可能会形成规模达几十甚至更大的联盟, 并且在线协作配送模式时效性要求高. 如果采用传统计算核仁解的方法, 难以在合理时间内完成成本分摊核仁解的计算. 因此, 求解类似依托平台的协作配送的较大规模成本分摊问题, 研究更加有效的求解核仁解的方法依然是非常值得研究和探索的领域[38, 39].

文章拟研究一种能够高效求解或近似求解核仁解的方法, 应用于依托平台的协作配送成本分摊问题. 核心贡献包括: 1) 提出一种新的成本分摊方法, 不仅能实现不采用传统线性规划优化方法, 只采用公式就能快速近似计算核仁解; 2) 而

且提出的成本分摊方法具有自身的科学内涵，从理论上证明了该方法输出的成本分摊方案，同时能满足总体理性、唯一性、可加性、策略等价相对不变性和一致性等众多分摊方案合理属性；3）更重要的是，该方法可以应用于任何支付可转移的合作博弈成本分摊问题.

1　协作配送成本分摊核仁解

1.1　依托平台的协作配送问题

借助现代通讯技术、互联网和移动支付等技术，成立第三方平台，撮合各配送企业完成协作配送全过程的运作方式，称为依托平台的协作配送. 整个过程大致分为三个环节：1）依托平台在线组建联盟；2）集中调度优化完成协作配送；3）计算协作配送成本分摊方案，进行成本分摊. 协作配送的示例如图 1 所示.

由图 1 可知，协作前正如当前物流配送的现状，不同配送企业的顾客分布区域高度重叠，导致不同企业的配送距离均很远. 如果 3 个企业协作配送，即通过相互交换部分离自身配送中心较远，而离其他企业配送中心较近的顾客，能够有效地缩短相互的配送距离. 另外，需要说明的两点是：1）协作配送问题不完全等价于 MDVRP 问题，因为协作配送问题需要在不同企业的配送中心间调配货物，在此将货物调配路线优化近似为解决一个经过多个配送中心的旅行商问题（traveling salesman problem，TSP）；2）根据优化问题的原理可知，协作后的成本必定小于等于协作前各企业的总成本.

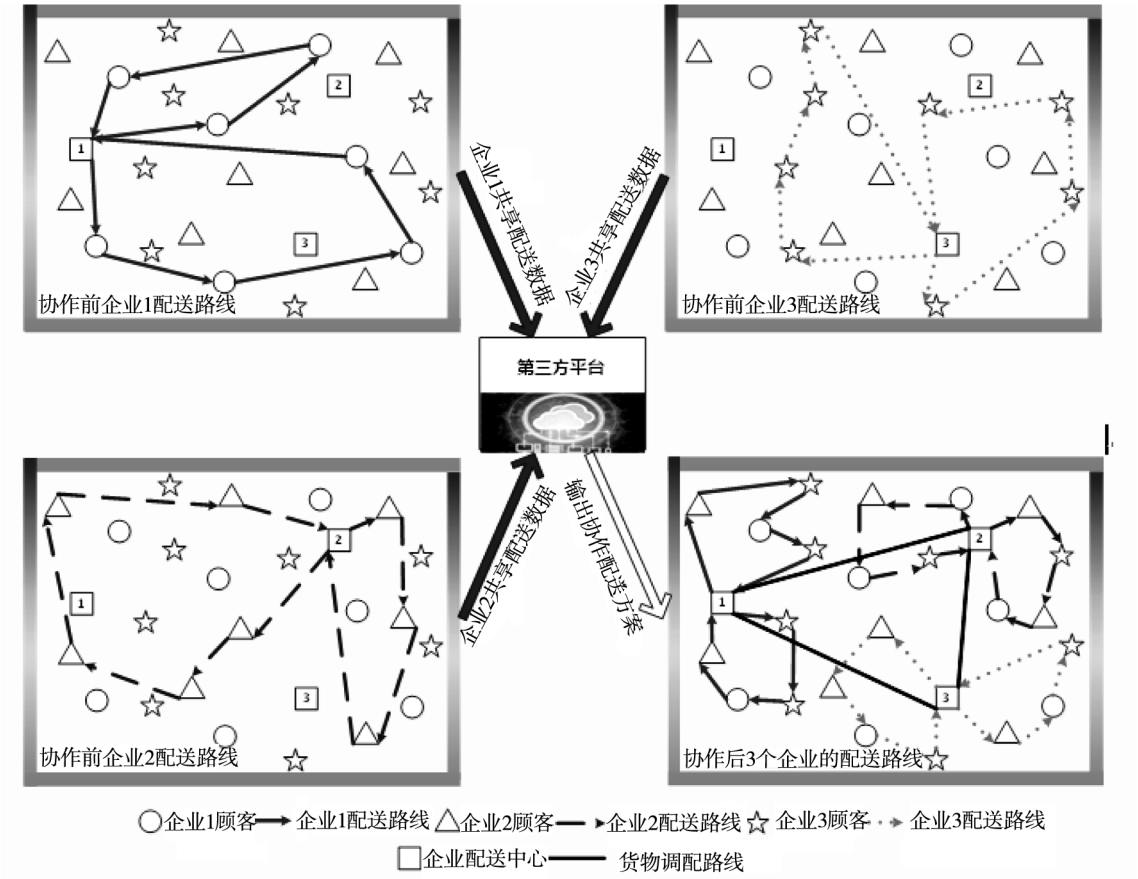

图 1　三个企业协作前和协作后的配送示意图

Fig. 1 Comparison diagram of cooperative vehicle routing problem with three enterprises

1.2　协作配送成本分摊核仁解含义

协作配送的成本分摊方案，应该如何公平分配？当前研究认为核仁解是最为公平的成本分摊方案. 核仁解是 Schmeidler 在 1969 年提出的一种分摊方案[8]，能够使所有子联盟总体满意度达到最高的转归解. 为了方便描述，文章用到的主要

符号解释如表 1 所示.

表 1 符号说明

Table 1 Symbols descriptions

符号	符号解释
N	协作配送联盟$\{1, 2, \cdots, n\}$
S	协作配送子联盟 $S\neq\varnothing$, $S\subseteq N$, 共 2^n-1 个
$c(S)$	联盟 S 的成员协作配送成本
x	成本分摊方案$(x_1, x_2, \cdots, x_n)$
$x(S)$	方案 x 子联盟 S 中成员分摊成本之和
n	协作配送联盟中企业数量
$e(S, x)$	子联盟 S 对成本分摊方案 x 的满意度, $c(S)-x(S)$
$E(c, x)$	成本分摊问题 c 中所有子联盟对成本分摊方案 x 满意度的非降序数列
$BS(i)$	如果成员 i 属于子联盟 S 等于 1,否则为 0

协作配送成本分摊核仁解如何求解? 本部分根据图 1 的示例假设企业 1、企业 2、企业 3 不协作配送的成本分别为 $c(\{1\})=80$, $c(\{2\})=90$, $c(\{3\})=100$. 其中 3 个企业协作的总成本 $c(\{1, 2, 3\})=180$, 并且 3 个企业中任意两个企业之间协作的成本分别为 $c(\{1, 2\})=120$, $c(\{1, 3\})=125$, $c(\{2, 3\})=130$. 那么企业 1、企业 2、企业 3 协作配送产生的 180 单位成本应该如何分配?

核仁解的提出学者 Schmeidler 通过比较不同分摊方案对应的子联盟满意度序列值的大小, 确定哪个方案更为公平. 比较方法如下: 先将成本分摊方案对应的子联盟满意度按升序排列, 如本例分别可得数列 $E(c, x)=(0, 0, 5, 10, 30, 30, 30)$、$E(c, y)=(0, 0, 5, 10, 20, 30, 40)$; 然后比较 $E(c, x)$ 和 $E(c, y)$ 的字典序列值, 字典序列值大的成本分摊方案更好. 字典序列值确定的方法如下: 先比较满意度中的最小元素, 最小元素更大的序列值更大; 如果最小元素相等再比较第二小的元素值;…; 直到比较出字典序更大的序列. 比如通过比较 $E(c, x)$ 和 $E(c, y)$ 发现从第 5 个元素开始不同, $E(c, x)$ 的第 5 个元素 30 大于 $E(c, y)$ 中的第 5 个元素 20, 即认为 $E(c, x)$ 的字典序列值比 $E(c, y)$ 更大. 那么在核仁解规则中认为成本分摊方案 $x=(50, 60, 70)$ 使子联盟的总体满意度比成本分摊方案 $y=(60, 60, 60)$ 的更大. 具体比较过程如图 2 所示.

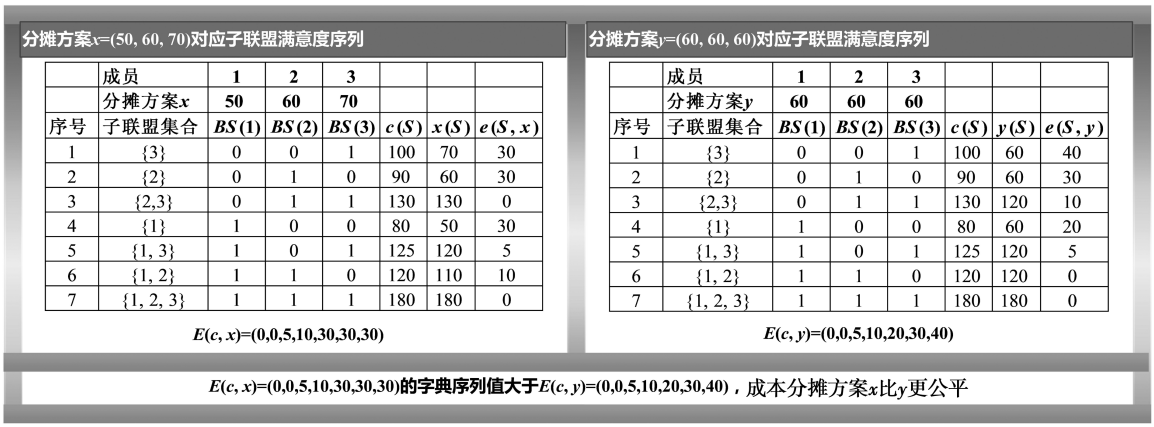

图 2 两个分摊方案子联盟总体满意度对比示意图

Fig. 2 Comparison of the sub-coalition satisfaction with two cost-sharing solutions

如果 $E(c, x^*)$ 的字典序列值最大, 那么 x^* 为本问题的核仁解.

1.3 求解核仁解的传统模型

那么传统方法是怎么求解核仁解的呢? 基本求解思路是: 首先采用线性规划模型得到使 2^n-1 个子联盟满意度的最小值取到最大的转归解集合为 T_1, 然后继续采用线性规划方法, 在 T_1 中令 2^n-1 个子联盟满意度的次小值取到最大的解集合为 T_2, 按此思路, 最后, 当在 T_{k-1} 中令 2^n-1 个子联盟满意度的倒数第 k 大值取到最大的解集合为 T_k 的时候, 当且仅当 T_k 只包含唯一解时, 得到核仁解.

求解核仁解的模型如下所示[29]

LP_1:

$$\max\ \varepsilon_1 \tag{1}$$

$$\text{s.t.}\ \sum_{i\in S} x_i+\varepsilon_1\leqslant c(S)\quad \forall S\subset N, S\notin \Gamma_1=$$

$$F_0 = \{\varnothing\} \tag{2}$$

$$x_1 + x_2 + \ldots + x_n = c(N) \tag{3}$$

$$x_i \leqslant c(\{i\}) \quad \forall i \in N \tag{4}$$

线性规划 LP_1 得到的成本分摊结果，可以最大化子联盟满意度的最小值，从而使总体满意度提升. 当 LP_1 为唯一解时，得到的求解结果即为核仁解，否则必须循环求解模型 LP_k，直至求解出唯一解.

LP_k：

$$\max \varepsilon_k \tag{5}$$

$$\text{s.t.} \sum_{i \in S} x_i + \varepsilon_k \leqslant c(S) \quad \forall S \subset N,$$

$$S \notin \Gamma_k = F_0 \cup F_1 \cup \ldots \cup F_{k-1} \tag{6}$$

$$\sum_{i \in S} x_i + \varepsilon_t = c(S) \quad \forall S \in F_t,$$

$$t \in \{1, 2, \ldots, k-1\} \tag{7}$$

$$x_1 + x_2 + \ldots + x_n = c(N) \tag{8}$$

$$x_i \leqslant c(\{i\}) \quad \forall i \in N \tag{9}$$

在第 k 个线性规划模型 LP_k 中，目标函数(5)表示最大化总体满意度向量中第 k 小的满意度 ε_k，其中约束(7)中的 F_i 表示第 $i(i < k)$ 个线性规划模型 LP_i 的解中，其对应所有子联盟的满意度中取到约束条件(6)中等号的联盟集合(假设第 2 个线性规划模型中，联盟 S 在约束(6)中取到等号，则联盟 $S \in F_2$；同时若联盟 T 在约束(6)中取到等号，则联盟 $T \in F_2$，即 $F_2 = \{S, T\}$). 此时 Γ_k 代表前 $k-1$ 个模型中约束(6)取等号的所有子联盟的简单集合，即 $\Gamma_k = \bigcup_{i \leqslant k-1} F_i$. 通过定义 $\Gamma_1 = \{\varnothing\}$. 约束条件(8)，约束条件(9)表示成本分摊方案是转归解.

线性规划模型 LP_k 中约束条件的总数量是 $2^n + n$ 个，当 n 达到 10 时，其约束条件的个数就会达到数千个. 而求解核仁解还需要反复求解多个这样的线性规划模型. 依托平台的协作配送问题中在线企业的数量可能远大于 10，且成本分摊的时效性要求很高. 因此，有必要研究更加高效的方法，文章拟提出的方法具体应用场景如图 3 所示.

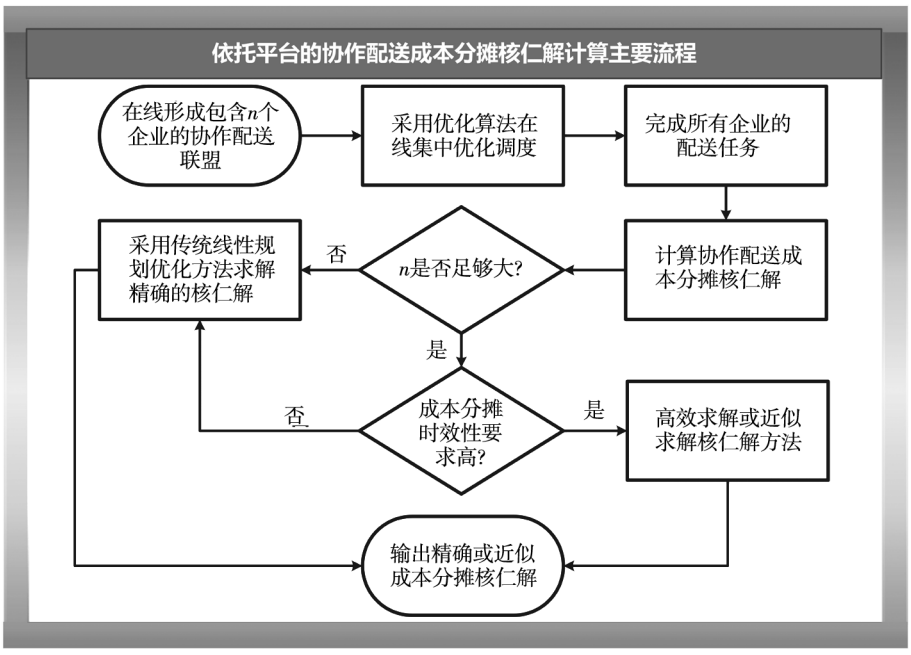

图 3　依托平台的协作配送问题成本分摊流程图

Fig. 3 Flow chart of cost allocation of collaborative distribution problem based on platform

2　子联盟满意度的发现与启示

通过分析和研究子联盟的满意度，发现有定理 1 ~ 定理 3 成立.

定理 1　假设成本分摊方案 x 满足总体理性，那么任意子联盟的满意度 $e(S, x)$ 之和是常数.

证明　要证明定理 1 成立，即证明式(10)的结果为常数.

$$\sum_{S \subseteq N} e(S, x) = \sum_{S \subseteq N} [c(S) - x(S)]$$

$$= \sum_{S \subseteq N} \left[c(S) - \sum_{i \in S} x_i \right] \tag{10}$$

由于 $N=\{1,2,3,\cdots,n\}$ 中的每个成员 $i(1 \leqslant i \leqslant n)$ 只有在子联盟 S 中和不在子联盟 S 中两种情况. 根据排列组合知识可知包含任意成员 i 的子联盟个数为 2^n-1 个. 因此式(10)可以化简为式(11).

$$\sum_{S\subseteq N}\left[c(S)-\sum_{i\in S}x_i\right]=\sum_{S\subseteq N}c(S)-2^{n-1}\times(x_1+x_2+\cdots+x_n) \tag{11}$$

又因为成本分摊方案 x 满足总体理性, 即 $x_1+x_2+\cdots+x_n=c(N)$,故式(11)可以变形为式(12).

$$\sum_{S\subseteq N}\left[c(S)-\sum_{i\in S}x_i\right]=\sum_{S\subseteq N}c(S)-2^{n-1}c(N) \tag{12}$$

显然, 对于一个确定的成本分摊问题, 式(12)是一个常数, 即定理 1 成立. 证毕.

定理 2 假设成本分摊方案 x 满足总体理性, 那么任意子联盟 S 的满意度 $e(S, x)$ 与补联盟 $N\backslash S$ 的满意度 $e(N\backslash S, x)$ 之和是常数.

证明 要证明定理 2 成立, 等价于证明式(13)是常数.

$$e(S,x)+e(N\backslash S,x)=c(S)+c(N\backslash S)-x(S)-x(N\backslash S) \tag{13}$$

因为成本分摊方案 x 满足总体理性, 那么 $x(S)+x(N\backslash S)=x_1+x_2+\cdots+x_n=c(N)$. 则式(13)可以改写为式(14).

$$c(S)+c(N\backslash S)-x(S)-x(N\backslash S)=c(S)+c(N\backslash S)-c(N) \tag{14}$$

显然, 对于一个确定的成本分摊问题, $c(S)$、$c(N\backslash S)$ 和 $c(N)$ 均为确定数, 即式(14)是一个常数, 所以定理 2 成立. 证毕.

根据定理 1 和定理 2 可知, 子联盟的满意度之和是常数, 且任意一对互补联盟满意度也是常数. 那么从另外一方面讲, 寻找核仁解的过程类似于分别将一组总和确定的数据数列 A 中的每个元素一分为二, 从而得到元素数量翻倍的另外一个数列 B, 那么应该如何将 A 中的元素一分为二, 才能使 B 数列序列值最大?通过研究发现, A 数列中的元素均平分为两个相等的元素时, B 数列的序列值达到最大. 也就是说, 有定理 3 成立.

定理 3 如果某个转归解 x 能够使任意子联盟 S 的满意度 $e(S, x)$ 与补联盟 $N\backslash S$ 的满意度 $e(N\backslash S, x)$ 相等, 那么 x 必定为核仁解.

证明 略. 证明过程见附录 1.

根据定理 3 可知, 任意子联盟的满意度与补联盟满意度相等时, 所有子联盟总体满意度必定最高. 由此构造子联盟满意度均衡函数, 如式(15)所示.

$$\begin{cases} f(x_1,x_2,\ldots,x_{n-1})=\sum_{S\subseteq N}\left[e(S,x)-\frac{L_S}{2}\right]^2 \\ e(S,x)=c(S)-x(S) \\ x(S)=\sum_{i\in S}x_i(1\leqslant i\leqslant n) \\ x_n=c(N)-(x_1+x_2+\cdots+x_{n-1}) \\ L_S=c(S)+c(N\backslash S)-c(N) \end{cases} \tag{15}$$

显然, 函数 $f\geqslant 0$, 且 $f=0$ 时等价于任意子联盟的满意度与补联盟满意度相等, 子联盟满意度最均衡, 总体满意度最高或称为子联盟满意度非降序数列的序列值最大. 由此可推知, 某个成本分摊方案 x 对应的函数 $f(x)$ 值与接近核仁解的程度存在负相关关联, 即 $f(x)$ 的值越小, 成本分摊方案 x 越接近核仁解, 如果 $f(x)$ 的值等于 0, 那么 x 就是核仁解.

文章研究需要验证的主要假设: 函数 f 是否存在极小值, 如果存在, 那么使 f 取到最小值的 x^* 将是一个能够使子联盟总体满意度达到高水平的成本分摊方案. 从而达到采用计算式求解或近似求解核仁解的目的.

3 子联盟满意度均衡分摊方法

3.1 满意度均衡模型求解

通过研究发现式(15)中的函数 f 存在唯一极小值. 由于函数 f 虽然包含等式 x_1, x_2, …, x_n 共 n 个变量, 但此处要求 $x_1+x_2+\cdots+x_n=c(N)$, 因此函数 f 实际共包含 $n-1$ 个自由变量. 验证函数 f 存在极小值的过程分三步: 1) 先求解关于每个自由变量的二次偏导得到海瑟矩阵; 2) 证明海瑟矩阵是正定矩阵; 3) 得到函数 f 存在极小值的结论. 详细求解和验证过程如下, 首先对任意自由变量 $x_i(1 \leqslant i \leqslant n-1)$ 求一阶偏导可得式(16).

$$\begin{cases}\dfrac{\partial f}{\partial x_i} = -2\sum\limits_{S\in S_{in}}\left[c(S)-x(S)-\dfrac{L_S}{2}\right]+\\ \quad 2\sum\limits_{S\in S_{ni}}\left[c(S)-x(S)-\dfrac{L_S}{2}\right]\\ S_{in}=\{S\mid S\subseteq N, i\in S, n\notin S\}\\ S_{ni}=\{S\mid S\subseteq N, n\in S, i\notin S\}\\ 1\leqslant i\leqslant n-1\end{cases} \tag{16}$$

基于式(16)可知，S_{in} 和 S_{ni} 集合中均共包含 2^{n-2} 个子联盟，且 S_{in} 中的子联盟包含成员 i 不包含成员 n，S_{ni} 中的子联盟包含成员 n 不包含成员 i，因此不难求得二阶导数如式(17)所示.

$$\begin{cases}\dfrac{\partial^2 f}{\partial x_i^2} = 2\times 2^{n-2}+2\times 2^{n-2}=2^n\\ \dfrac{\partial^2 f}{\partial x_i\partial x_j} = 2\times 2^{n-3}+2\times 2^{n-3}=2^{n-1},\\ \quad 1\leqslant i\leqslant n-1, i\neq j\end{cases} \tag{17}$$

根据式(17),可以得到函数 f 的海瑟矩阵如式(18)所示.

$$2^{n-1}\begin{pmatrix}2 & 1 & \cdots & 1\\ 1 & 2 & \cdots & 1\\ \vdots & \vdots & \ddots & \vdots\\ 1 & 1 & \cdots & 2\end{pmatrix} \tag{18}$$

显然,f 的海瑟矩阵为正定矩阵，因此函数 f 存在极小值. 令一阶导数等于 0，即式(16)等于 0，可得到式(19).

$$\begin{aligned}\frac{\partial f}{\partial x_i} &= -2\sum_{S\in S_{in}}\left[c(S)-x(S)-\frac{L_S}{2}\right]+\\ &\quad 2\sum_{S\in S_{ni}}\left[c(S)-x(S)-\frac{L_S}{2}\right]\\ &=0,\ 1\leqslant i\leqslant n-1\end{aligned} \tag{19}$$

通过对式(19)化简可得到式(20)，详细化简过程请见附录 2.

$$x_i - x_n = \frac{\sum\limits_{S\in S_{in}} c(S) - \sum\limits_{S\in S_{ni}} c(S)}{2^{n-2}},\ 1\leqslant i\leqslant n-1 \tag{20}$$

实际上，式(20)表示的是 $n-1$ 个等式，同时结合成本分摊总体理性等式 $x_1+x_2+\cdots+x_n=c(N)$，可以求解得到使函数 f 取到极小值的分摊方案 x^*. 如计算式(21)所示.

$$\begin{cases}x_i = x_n + M_{in}\\ M_{in}=\dfrac{\sum\limits_{S\in S_{in}} c(S)-\sum\limits_{S\in S_{ni}} c(S)}{2^{n-2}},\quad 1\leqslant i\leqslant n-1\\ x_n = \dfrac{c(N)-\sum\limits_{k=1}^{n-1} M_{kn}}{n}\end{cases} \tag{21}$$

其中 M_{in} 表示的含义是成员 i 比 n 应该分摊的更多成本量. 由于成本分摊中的 n 个变量在 f 中是对称的，因此，从 n 个变量中选择任意其他 $n-1$ 个变量为自由变量均能得到上述结论和 x^* 计算公式. 文章称该方法为满意度均衡成本分摊法(Satisfaction Balance Cost Allocation Method, SBCAM).

3.2　分摊方法合理属性分析

SBCAM 成本分摊方法具有以下 7 个合理属性.

属性 1　分摊结果必定满足总体合理性，即成员的分摊结果和等于 $c(N)$. 对应于实际管理问题，各成员的分摊成本之和必须等于协作总成本.

证明　因为分摊结果是式(20)表示的 $n-1$ 个方程和总体理性等式 $x_1+x_2+\cdots+x_n=c(N)$ 共 n 个方程求解得到，因此属性 1 必定成立.

属性 2　分摊结果具有唯一性，即分摊的结果是单值解. 对应于实际管理问题，分摊结果的唯一性可以保证成员之间不会因为存在多个分摊方案产生分歧.

证明　按照 SBCAM 的计算式(21)可知仅能得到唯一分配方案.

属性 3　分摊结果具有可加性，如果 $c(S)=c_1(S)+c_2(S)$(S 是 N 的子集)，且 $c_1(S)$ 的 SBCAM 分摊结果为 y^*，$c_2(S)$ 的 SBCAM 分摊结果为 z^*，那么 SBCAM 对 $c(S)$ 的分摊结果必定为 $x^*=y^*+z^*$. 对应于实际管理问题，协作产生的各部分成本，可以加和后采用该方法一次性分摊.

证明　按照 SBCAM 的计算式(21)可知，任意子联盟协作成本为 $c_1(S)$ 的成本分摊问题，由 SBCAM 计算得到的分摊结果 y^* 计算过程如式(22)所示.

$$\begin{cases} y_i = y_n + M_{in}^1 \\ M_{in}^1 = \dfrac{\sum\limits_{S\in S_{in}} c_1(S) - \sum\limits_{S\in S_{ni}} c_1(S)}{2^{n-2}}, 1 \leqslant i \leqslant n-1 \\ y_n = \dfrac{c_1(N) - \sum\limits_{k=1}^{n-1} M_{kn}^1}{n} \end{cases} \tag{22}$$

同理，可得 z^* 计算过程如式(23) 所示.

$$\begin{cases} z_i = z_n + M_{in}^2 \\ M_{in}^2 = \dfrac{\sum\limits_{S\in S_{in}} c_2(S) - \sum\limits_{S\in S_{ni}} c_2(S)}{2^{n-2}}, 1 \leqslant i \leqslant n-1 \\ z_n = \dfrac{c_2(N) - \sum\limits_{k=1}^{n-1} M_{kn}^2}{n} \end{cases} \tag{23}$$

根据已知条件 $c(S) = c_1(S) + c_2(S)$ 和上述计算式(22) 和式(23) 可推导得到，有式(24) 成立.

$$\begin{aligned} M_{in}^1 + M_{in}^2 &= \frac{\sum\limits_{S\in S_{in}} c_1(S) - \sum\limits_{S\in S_{ni}} c_1(S)}{2^{n-2}} + \frac{\sum\limits_{S\in S_{in}} c_2(S) - \sum\limits_{S\in S_{ni}} c_2(S)}{2^{n-2}} \\ &= \frac{\sum\limits_{S\in S_{in}} [c_1(S) + c_2(S)] - \sum\limits_{S\in S_{ni}} [c_1(S) + c_2(S)]}{2^{n-2}} \\ &= \frac{\sum\limits_{S\in S_{in}} c(S) - \sum\limits_{S\in S_{ni}} c(S)}{2^{n-2}} \\ &= M_{in}, 1 \leqslant i \leqslant n-1 \end{aligned} \tag{24}$$

结合上述等式，进一步可得式(25) 和式(26) 成立.

$$\begin{aligned} y_n + z_n &= \frac{c_1(N) - \sum\limits_{k=1}^{n-1} M_{kn}^1}{n} + \frac{c_2(N) - \sum\limits_{k=1}^{n-1} M_{kn}^2}{n} \\ &= \frac{c_1(N) + c_2(N) - \sum\limits_{k=1}^{n-1} (M_{kn}^1 + M_{kn}^2)}{n} \\ &= \frac{c(N) - \sum\limits_{k=1}^{n-1} M_{kn}}{n} \\ &= x_n \end{aligned} \tag{25}$$

$$\begin{aligned} y_i + z_i &= y_n + M_{in}^1 + z_n + M_{in}^2 (1 \leqslant i \leqslant n-1) \\ &= x_n + M_{in} \\ &= x_i \end{aligned} \tag{26}$$

因此，$x^* = y^* + z^*$　　证毕.

属性 4　分摊结果具有策略等价相对不变性，令 $c(S) = k \times c_1(S) + a(S)$，其中 $k > 0$，$a = (a_1, a_2, \cdots, a_n)$ 为常数向量，$a(S) = \sum\limits_{i\in S} a_i$. 即 $c(S)$ 与 $c_1(S)$ 对应的成本分摊问题策略等价，并设 $c_1(S)$ 的 SBCAM 分摊结果为 y^*，那么 SBCAM 对 $c(S)$ 的分摊结果必定为 $x^* = k \times y^* + a$. 对应于实际管理问题，以不同博弈等价策略改变子联盟的协作成本，各成员分摊的成本值会发生同规律变化.

证明　按照 SBCAM 的计算式(21)，根据属性 4 中的已知信息，有计算式(27) 成立.

$$\begin{aligned} M_{in} &= \frac{\sum\limits_{S\in S_{in}} c(S) - \sum\limits_{S\in S_{ni}} c(S)}{2^{n-2}}, 1 \leqslant i \leqslant n-1 \\ &= \frac{\sum\limits_{S\in S_{in}} [kc_1(S) + a(S)] - \sum\limits_{S\in S_{ni}} [kc_1(S) + a(S)]}{2^{n-2}} \\ &= k \times \frac{\sum\limits_{S\in S_{in}} c_1(S) - \sum\limits_{S\in S_{ni}} c_1(S)}{2^{n-2}} + (a_i - a_n) \\ &= k \times M_{in}^1 + (a_i - a_n) \end{aligned} \tag{27}$$

那么结合计算式(21) 和式(27)，可得式(28) 和式(29) 必定会成立.

$$\begin{aligned} x_n &= \frac{c(N) - \sum\limits_{k=1}^{n-1} M_{kn}}{n} \\ &= \frac{\left(k \times c_1(N) + \sum\limits_{i=1}^{n} a_i\right) - \sum\limits_{k=1}^{n-1} (k \times M_{kn}^1 + a_k - a_n)}{n} \\ &= k \times \frac{c_1(N) - \sum\limits_{k=1}^{n-1} M_{kn}^1}{n} + a_n \\ &= k \times y_n + a_n \end{aligned} \tag{28}$$

$$\begin{aligned} x_i &= x_n + M_{in} (1 \leqslant i \leqslant n-1) \\ &= k \times y_n + a_n + k \times M_{in}^1 + a_i - a_n \\ &= k \times (y_n + M_{in}^1) + a_i \\ &= k \times y_i + a_i \end{aligned} \tag{29}$$

因此，$x^* = k \times y^* + a$　　证毕.

属性 5　满足一致性原则，即对不同子联盟中具有相同边际贡献的成员分摊结果必定相同. 对应于实际管理问题，当成员的实际贡献相同，其成本分摊值必定相等.

证明　根据式(21)，假设成员 i，$j(i \neq j)$ 对任意子联盟的边际作用相同，即假设对任意 $S \subset N$，且 $i \notin S$，$j \notin S$，均有 $c(S \cup \{i\}) = c(S \cup \{j\})$ 成立，那么 M_{ij} 必定等于 0，因此成员 i，j 的成本分摊结果也必定会相等.

属性 6　满足匿名性原则，即任意成员分摊成本结果与其标识序号顺序无关. 对应于实际管理问题，合作成员的分摊成本量只与分摊成本贡献相关，与联盟中对每个成员的标号顺序没有任何关系.

证明　根据计算式(21)，假设成员 i，$j(i \neq j)$ 互换标识号，并按此调换相应子联盟的分摊成本值，那么 i，j 的分摊结果值也会随之互换，因此，成员的分摊结果与标识顺序号无关.

属性 7　满足可比性原则，即任意成员可以与其他成员比较成本分摊结果的差别. 对应于实际管理问题，成员一般通过与其他成员比较来确定自身分摊成本的公平性，实现科学量化成员间成本分摊的差异，有利于成员接受分摊方案.

证明　根据计算式(21)，是以成员 n 为参照，成员 n 可以对比与其他任意 $n-1$ 个成员的贡献差别，即其他成员与 n 相比的额外贡献 M_{in}. 其实根据式(15)可知，在求解极值过程中可以选择任意 $n-1$ 个变量为自由变量，故计算式(21)可以变形为以任何成员为参照的计算式，因此，SBCAM 分摊方法能够合理解释任意两个不同成员的分摊结果差别.

3.3　SBCAM 计算复杂度分析

根据式(27)可知，SBCAM 的计算量主要体现在需要分别计算 $n-1$ 个 M_{in} 值($1 \leqslant i \leqslant n-1$)，计算任意一个 M_{in} 值需要分别遍历 S_{in} 和 S_{ni} 中的子联盟成本值，而 S_{in} 和 S_{ni} 中的子联盟数量均为 2^{n-2} 个，因此 M_{in} 值的复杂度可由式(30)计算.

$$\Omega(M_{in}) = 2^{n-2} + 2^{n-2} = 2^{n-1} \tag{30}$$

所以SBCAM的计算复杂度是式(30)的n–1倍，其复杂度可由式(31)表示.

$$\Omega(SBCAM) = (n-1)\Omega(M_{in}) = (n-1)2^{n-1} \tag{31}$$

当 n 取值不大时SBCAM可高效计算出成本分摊结果，但随着 n 的增加SBCAM的计算量也会增大.

3.4　SBCAM 有效计算策略

在上述分析 SBCAM 的计算复杂度时，是根据式(20)按先后次序计算 $n-1$ 个 M_{in} 值，从而在计算 M_{in} 值时需要重复遍历相同的子联盟，不妨称其为“串行计算”方式.

在此文章提出了一个实现“并行计算”的策略，主要思路如下：1）采用二进制数组 BS 方式表示子联盟；2）将 M_{in} 值计算公式分解为式(32)所示；3）提出快速计算 H_{in} 值的策略如图4所示.

$$\begin{cases} M_{in} = \dfrac{H_{ni} - H_{in}}{2^{n-2}} \\ H_{in} = \displaystyle\sum_{S \in S_{in}} c(S)\ ,\ 1 \leqslant i \leqslant n-1 \\ H_{ni} = \displaystyle\sum_{S \in S_{ni}} c(S) \end{cases} \tag{32}$$

```
1: Input array cS including 2^n-1 cost value of subcoalition;
2: H_in←0;   //初始化数组 H_in(共包含 n-1 个元素)
3: H_ni←0;   //初始化数组 H_ni(共包含 n-1 个元素)
4: For j=1 to 2^n-1 do
5:       BS←Binary(j, n); //产生第 j 个子联盟对应的包含 n 个元素 0-1 数组(二进制),
                          如 BS(i)=1, 表示第 i 个成员属于子联盟, 反之亦然;
6:       For i=1 to n-1 do
7:            If  BS(i)=1 and BS(n)=0   then   //当前子联盟属于 S_in;
8:               H_in(i)←H_in(i)+ cS(j);
9:            EndIf
10:           If  BS(i)=0 and BS(n)=1   then   //当前子联盟属于 S_ni;
11:              H_ni(i)←H_ni(i)+ cS(j);
12:           EndIf
13:      EndFor
14: EndFor
15: M_in=(H_in - H_ni)/2^(n-2);
16: Return M_in
```

图 4　采用二进制表示子联盟实现并行计算 M_{in} 示意图

Fig. 4 Parallel computing M_{in} by using binary representation sub-coalition

由图4 可知，$n-1$ 个 M_{in} 值的复杂度等价于遍历所有的 2^n-1 个子联盟. 因此结合“并行计算” 策略后 SBCAM 的计算复杂度可以降到 2^n-1，从而计算速度与式(31) 中的复杂度 $(n-1)2^{n-1}$ 相比，可以提高 $(n-1)/2$ 倍. 因当前经典的成本分摊方法均要输入 2^n-1 个子联盟的成本信息，所以均至少需要遍历 2^n-1 个子联盟，因此结合 BS 策略后 SBCAM 计算效率已达到极为高效的水平.

4　数据实验与结果分析

4.1　SBCAM 求解过程

本部分采用 SBCAM 求解了包括协作配送问题在内的众多成本分摊问题. 并将求解结果与核仁解进行了比较，通过计算分摊结果的偏差来量化 SBCAM 的求解质量. 采用 MatlabR2018b 编程，运行平台：CPU 为 Inter（R）core（TM）2 Duo，内存为 2.0G，主频为 2.93GHz.

首先验证符合定理 3 中描述的，如果存在分摊方案使任意子联盟满意度均能与补联盟满意度相等的成本分摊问题，SBCAM 一定能够求解得到该分摊方案，且必为核仁解. 如图 5 所示的物流协作配送问题，假设有配送企业 1，企业 2，企业 3，其中企业 1 有 3 个顾客，企业 2 有 3 个顾客，企业 3 有 6 个顾客，且每个顾客需求量为 30，每个企业的配送车辆载量为 100. 显然，企业 1 和企业 2 需要派送 1 辆配送车辆，企业 3 需要派送 2 辆车进行配送. 并且企业 1，企业 2，企业 3 协作前的配送成本分别为 $c(\{1\})=50$，$c(\{2\})=20$，$c(\{3\})=40$.

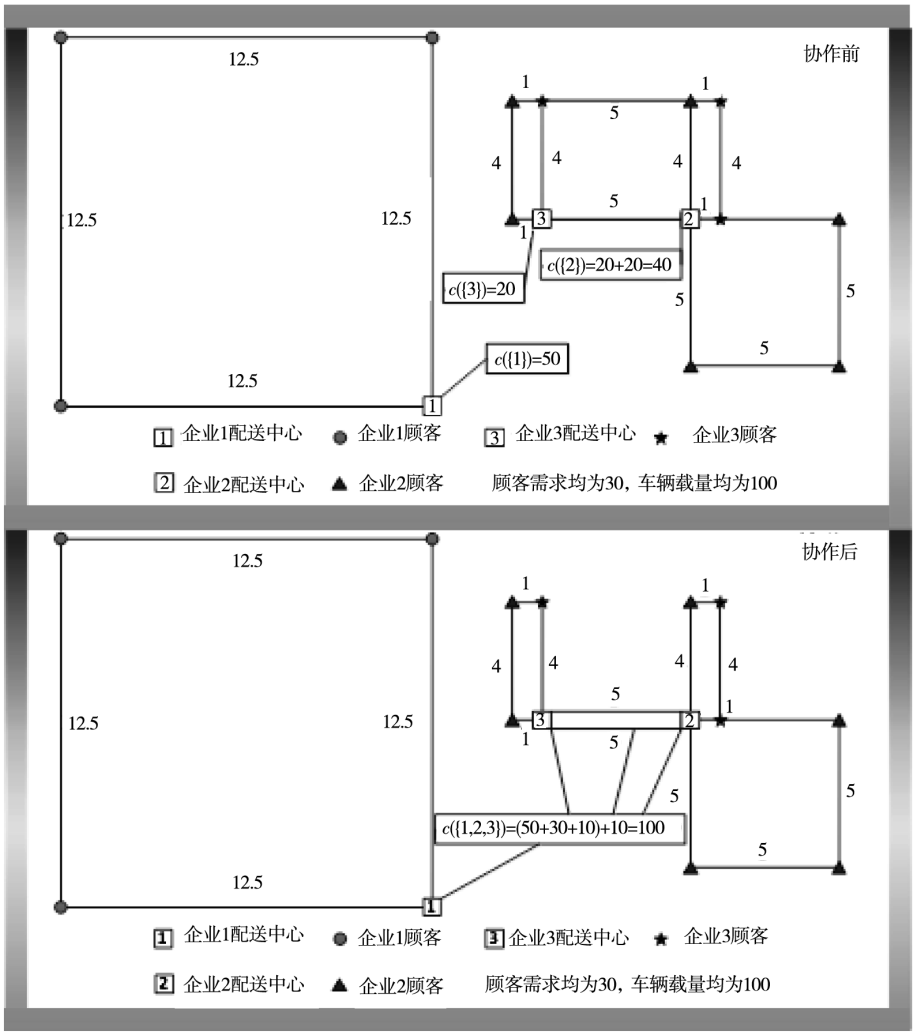

图 5　包含 3 个企业的协作配送问题

Fig. 5 Acollaborative distribution problem with three enterprises

如果 3 个企业协作配送，企业 2 和企业 3 之间可以交换两个顾客，从而能够将企业 2 和企业 3 的配送距离分别缩短至 10 和 30，但需要在企业 2 和企业 3 的配送中心间调配货物，因此需要额外考虑 10 单位（企业 2 和企业 3 的配送中心距离为 5）的货物调配成本. 所以协作后总配送成本 $c(\{1,2,3\}) = (50 + 10 + 30) + 10$. 该协作配送成本分摊问题中所有子联盟协作配送的成本等信息如表 2 所示.

上述协作配送成本分摊问题采用 SBCAM 方法可以得到分摊方案 $x^* = (50, 35, 15)$，计算过程如图 6 所示.

表 2　3 个协作企业的协作配送成本分摊问题子联盟成本

Table 2 The cost of all sub-coalitions of collaborative distribution problem with three enterprises

序号 j	子联盟 S^j	B^j	$c(S^j)$
1	{3}	[0, 0, 1]	20
2	{2}	[0, 1, 0]	40
3	{2,3}	[0, 1, 1]	50
4	{1}	[1, 0, 0]	50
5	{1,3}	[1, 0, 1]	70
6	{1,2}	[1, 1, 0]	90
7	{1,2,3}	[1, 1, 1]	100

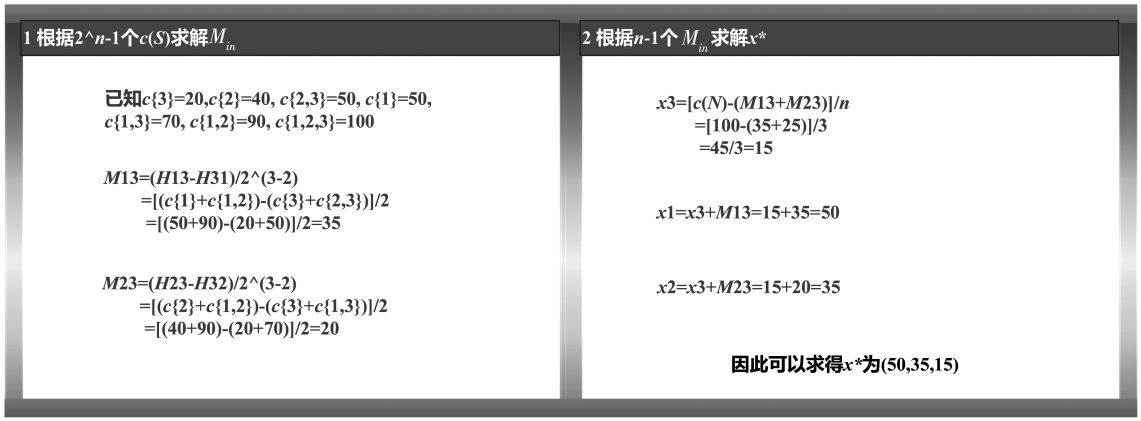

图 6　SBCAM 计算过程示意图

Fig. 6 Diagram of SBCAM calculation process

其实 $x^* = (50, 35, 15)$ 就是本问题的成本分摊核仁解. 当然，不是所有的成本分摊问题都存在一个分摊方案，使任意子联盟和对应的补联盟满意度相等. 也就是说，对于很多成本分摊问题 SBCAM 的求解结果可能不一定是核仁解. 那么 SBCAM 的求解性能与核仁解的有多大差别？

4.2　SBCAM 求解性能验证

此部分主要验证 SBCAM 的求解速度和求解质量性能. 验证算例分为两部分：1）设计规模较大的协作配送问题算例（验证求解速度）；2）当前文献中的成本分摊问题（验证求解质量）.

4.2.1　协作配送问题算例

本部分根据文献[40]的研究成果，采用估算方法，根据顾客数量，需求量，顾客分布面积等信息，估算协作配送问题的协作成本，具体原理请参看文献[40]. 本部分协作配送算例的设计规则如下.

1）共设计 28 个算例，企业数 n 取值为 3 ~ 30，且每个企业的顾客数均为 100；

2）每个企业的顾客和配送中心随机均匀分布在 10 km × 10 km 的区域中，即面积 A = 100 km^2；

3）任意子联盟协作配送的成本由式(33) ~ 式(34) 估算得到.

$$\begin{aligned} EstML &= \sum_{i=1}^{n}\sum_{j}^{K_i} EstTL_{ij} + f(A_n, n) \\ &= \sum_{i=1}^{n}\sum_{j}^{K_i} f(A_{ij}, n_{ij}) + f(A_n, n) \end{aligned} \tag{33}$$

$$\begin{aligned} EstTL &= f(A, n) \\ &= \alpha\sqrt{An} \\ &= (0.712\,4 + a \times n^b) \times \sqrt{n} \times \sqrt{A} \end{aligned} \tag{34}$$

其中 $a = 2.243\,0, b = -0.587\,7$.

设计评估任意两个满足总体理性的分摊方案

偏差指标为 $Dev\%$，计算方法如式(35) 所示.

$$\begin{cases} Dev\% = \dfrac{\sum_{i=1}^{n} |x_i - y_i|}{2C(N)} \times 100\% \\ x_1 + x_2 + \cdots + x_n = C(N) \\ y_1 + y_2 + \cdots + y_n = C(N) \\ x_i, y_i \geqslant 0,\ i = 1,2,\ldots,n \end{cases} \tag{35}$$

可以证明 $Dev\%$ 的取值范围必定为[0%，100%]. 证明如式(36) ~ 式(37) 所示.

$$\sum_{i=1}^{n} |x_i - y_i| \geqslant 0 \tag{36}$$

$$\sum_{i=1}^{n} |x_i - y_i| \leqslant (x_1 + x_2 + \cdots + x_n) + (y_1 + y_2 + \cdots + y_n) = 2C(N) \tag{37}$$

因此，有式(38) 成立.

$$0\% \leqslant Dev\% = \frac{\sum_{i=1}^{n} |x_i - y_i|}{2C(N)} \leqslant 100\% \tag{38}$$

分别采用 SBCAM 和文献[29]中核仁解循环线性规划方法求解了上述 28 个算例，求解耗时及 SBCAM 和核仁解的 $Dev\%$ 信息如表 3 所示.

表 3 SBCAM 与核仁解方法求解 28 个算例的速度和偏差结果

Table 3 The speed and deviation results of 28 instances solved by SBCAM and nucleolus method

No.	n	Times/s		Dev/%	No.	n	Times/s		Dev/%
		SBCAM	Nucleolus				SBCAM	Nucleolus	
1	3	0.000 1	0.380 0	5.13	15	17	1.123 0	4.74E +06	–
2	4	0.000 1	0.620 0	5.74	16	18	2.239 1	1.90E +07	–
3	5	0.000 2	1.060 0	5.92	17	19	4.530 3	7.58E +07	–
4	6	0.000 5	2.800 0	6.15	18	20	8.650 2	3.03E +08	–
5	7	0.000 9	10.440 0	5.40	19	21	16.871 3	1.21E +09	–
6	8	0.001 9	34.211 2	5.01	20	22	33.079 9	4.85E +09	–
7	9	0.003 9	115.913 6	4.95	21	23	62.958 8	1.94E +10	–
8	10	0.008 2	394.241 1	4.03	22	24	123.216 5	7.76E +10	–
9	11	0.015 0	1 297.841 6	3.70	23	25	244.842 9	3.11E +11	–
10	12	0.028 6	4 321.822 2	4.36	24	26	465.636 3	1.24E +12	–
11	13	0.058 4	18 508.348 0	3.79	25	27	917.800 2	4.97E +12	–
12	14	0.124 8	7.40E +04	–	26	28	1 910.177 1	1.99E +13	–
13	15	0.265 2	2.96E +05	–	27	29	4 005.485 6	7.95E +13	–
14	16	0.514 8	1.18E +06	–	28	30	7 710.561 4	3.18E +14	–

由表 3 可知，SBCAM 的求解速度远远快于传统求解核仁解的循环线性规划方法，SBCAM 能够在 1s 之内求解规模为 16 的成本分摊问题，而传统核仁解方法需要耗时 13.7 天的 cpu 时间. SBCAM 求解耗时随着问题规模的增加，耗时量大约以 2 倍的速度递增，而传统方法大概以 4 倍的速度增加耗时量. 因此当问题规模较大时，计算核仁解需要耗费巨大的时间，如 $n = 15$ 时，采用传统线性规划方法求解核仁解需要耗费的 CPU 时间大约是 3.4 天，$n = 20$ 时需要计算 9.6 年，而当 $n = 30$ 时则需要计算 1 008.279 4 万年. 显然，对于依托平台的协作配送模式，要求协作完成后立刻进行成本分摊，在采用传统方法求解核仁解时，当协作的企业超过 15 时所需要的计算时间就已经超过合理范围，而采用文章提出的 SBCAM 计算 $n = 30$ 的超大规模协作配送问题，仅需要 2 个多小时的耗时，这比求解核仁解的传统优化方法快近 412 亿倍. 需要指出的是在表 3 中当 n 小于等于 13 时，耗时量为实际运算结果，当 n 大于 13 时其求解耗时通过估算得到，其估算方法根据循环线性规划方法求解 Nucleolus 耗时随问题规模每增加 1 单位，耗时量大概增加 4 倍的规律计算得到，估算的耗时在表 3 中采用科学计数法格式表示. 因此，当规模大于 13 时无 Shapley

求解结果，所以对应的偏差指标 *Dev*% 用“ – ”标识.

另外通过表 3 不仅能看出 SBCAM 的求解耗时更少，而且通过对比可以发现 SBCAM 的分摊结果与核仁解仅有较小的偏差，28 个算例的综合偏差指标 *Dev*% 大概分布在 3.70% ~6.15% 之间.

4.2.2　文献中成本分摊或利益分配算例

合作博中的成本分摊或利益分配问题已被广泛地研究，文献[30]中研究了当前 5 篇文献采用传统求解核仁解的循环线性规划方法，求解核仁解产生了偏差. 本部分采用 SBCAM 方法求解了 5 篇文献中的例子，并与文献[30]中正确的核仁解进行了对比. 计算的结果如表 4 所示. 需要指出的是表 4 中 $v(S)$ 代表利益 $c(S)$ 代表成本，利益分配问题和成本分摊问题可以等价转换，转换方式 $c(S)=v(N)-v(N\backslash S)$.

由表 4 可知，SBCAM 求解结果与问题核仁解的偏差在 1.36% ~5.06% 之间.

表 4　SBCAM 求解 5 个算例结果与核仁解的偏差

Table 4 The deviation between the results of five instances solved by SBCAM and the method for nucleolus

S	$v(S)$[41]	S	$v(S)$[42]	S	$v(S)$[43]	$c(S)$[44]	$v(S)$[45]	
{1}	0.060	{1}	0.00	{1}	1.275	375 144	46 125.0	
{2}	0.168	{2}	0.00	{2}	3.471	245 280	17 437.5	
{3}	0.030	{3}	0.00	{3}	1.466	211 239	5 812.5	
{4}	0.249	{4}	0.00	{1,2}	7.005	568 232	69 187.5	
{5}	0.000	{1,2}	0.68	{1,3}	4.081	551 055	53 812.5	
{1,2}	0.378	{1,3}	0.24	{2,3}	5.672	452 411	30 750.0	
{1,3}	0.144	{1,4}	0.75	{1,2,3}	11.210	772 500	90 000.0	
{1,4}	0.408	{2,3}	0.26					
{1,5}	0.182	{2,4}	0.51					
{2,3}	0.337	{3,4}	0.07					
{2,4}	0.538	{1,2,3}	1.03					
{2,5}	0.279	{1,2,4}	1.53					
{3,4}	0.383	{1,3,4}	1.02					
{3,5}	0.083	{2,3,4}	0.75					
{4,5}	0.386	{1,2,3,4}	1.89					
{1,2,3}	0.536							
{1,2,4}	0.747	文献[41]	x_1	x_2	x_3	x_4	x_5	*Dev*%
{1,2,5}	0.485	Nucleolus	0.165 000	0.320 500	0.084 500	0.374 500	0.055 500	3.92
{1,3,4}	0.546	SBCAM	0.162 050	0.297 800	0.099 175	0.360 925	0.080 050	
{1,3,5}	0.255	文献[42]	x_1	x_2	x_3	x_4	*Dev*%	
{1,4,5}	0.561	Nucleolus	0.75	0.48	0.18	0.47	5.06	
{2,3,4}	0.706	SBCAM	0.66	0.54	0.19	0.50		
{2,3,5}	0.379	文献[43]	x_1	x_2	x_3	*Dev*%		
{2,4,5}	0.668	Nucleolus	3.089 3	5.285 3	2.835 5	1.84		
{3,4,5}	0.473	SBCAM	3.295 8	5.189 3	2.724 8			
{1,2,3,4}	0.906	文献[44]	x_1	x_2	x_3	*Dev*%		
{1,2,3,5}	0.573	Nucleolus	331 695.3	233 051.3	207 753.5	1.36		
{1,2,4,5}	0.874	SBCAM	342 205.7	227 951.7	202 342.7			
{1,3,4,5}	0.634	文献[45]	x_1	x_2	x_3	*Dev*%		
{2,3,4,5}	0.801	Nucleolus	52 687.5	24 468.8	12 843.8	1.56		
{1,2,3,4,5}	1.000	SBCAM	51 750.0	25 875.0	12 375.0			

这充分说明虽然 SBCAM 方法求解原理是，得到所有子联盟的满意度与其补联盟的满意度总偏差最小的分摊方案，这与使总体满意度序列值最大的核仁解具有一定差别，但这两种原理具有

很强的内在关联性,且当成本分摊问题满足定理 3 时 SBCAM 求解结果即为核仁解. 因此, SBCAM 的求解结果非常接近于甚至等于核仁解.

另外, 为进一步评估 SBCAM 方法性能及可改进性, 对比了一些最新求解核仁解与 Shapley 值的方法, 最新方法分别选择了文献[26]中的 Heuristic-Nucleolus、文献[27]中的 Approximate-Nucleolus 及文献[46]中的 LPSA-Shapley、文献[5]中的 B-T Shapley, 选择这些方法的原因主要是这些方法均是求解核仁解和 Shapley 值的近似快速方法, 并且是在近 2 年内发表的论文. 在表 5 ~ 表 7 中从理论特征与实验结果两方面将 SBCAM 和 B-T SBCAM 与这些最新方法进行了对比, 其中 B-T SBCAM 是采用作者在之前发表的文献[5]中二叉树分解法 Binary Tree 提升 Shapley 速度策略, 所形成的提升 SBCAM 方法速度性能方法.

在表 6 中基于文献[26, 27]中涉及的算例规模 10 和规模 18, 对比了相应方法与 SBCAM 的性能, 其中精度偏差是指与核仁解 Nucleolus 对比的结果; 在表 7 中基于文献[46]中涉及的算例规模 25, 对比了相应方法与 SBCAM 的性能, 其中精度偏差是指与 Shapley 值对比的结果. 需要指出的是, 求解速度的对比可能存在一定误差, 这主要是因为运行平台和采用软件等方面具有差别所造成的.

表 5 SBCAM 与最新求解核仁解及 Shapley 值方法的理论特征对比

Table 5 Comparison of SBCAM with the latest method for nucleolus and Shapley value on theoretical characteristics

方法	基于的原理	求解方式	复杂度	提速策略
Heuristic-Nucleolus[26]	子联盟满意度字典序最大化	优化	2^n 数量级约束的线性规划模型	减少约束条件数量
Approximate-Nucleolus[27]	子联盟满意度字典序最大化	优化	2^n 数量级约束的线性规划模型	减少约束条件数量
Shapley	成员边际贡献	公式	2^n 数量级子联盟成本的公式计算	无
LPSA-Shapley[46]	成员边际贡献	公式	抽样数量级算法 + 公式计算	减少所需子联盟数量
B-T Shapley[5]	成员边际贡献	公式	$2n-1$ 数量级子联盟成本的公式计算	Binary Tree 分级策略
SBCAM	子联盟满意度均衡	公式	$2n$ 数量级子联盟成本的公式计算	图 4 中的并行策略
B-T SBCAM	子联盟满意度均衡	公式	2^n-1 数量级子联盟成本的公式计算	Binary Tree 分级策略

表 6 SBCAM 与最新求解核仁解方法的实验结果对比

Table 6 Comparison of SBCAM with the latest method for nucleolus on experimental result

方法	算例规模/n	精度偏差/%	求解速度/s	与 SBCAM 耗时比
Nucleolus	10 ~ 18	0.00	4 321.82 以上	大于 40 多万
Heuristic-Nucleolus[26]	18	5.54	23.75	10.60
Approximate-Nucleolus[27]	10	5.32	2.00	200.00
SBCAM	10 ~ 18	4.36	0.01 ~ 2.24	1.00

表 7 SBCAM 与最新求解 Shapley 值方法的实验结果对比

Table 7 Comparison of SBCAM with the latest method for Shapley value on experimental result

方法	算例规模/n	精度偏差/%	求解速度/s	与 SBCAM 耗时比
Shapley[5]	25	0.00	15 080.49	61.59
LPSA-Shapley[46]	25	3.65	34.75	0.14
B-T Shapley[5]	25	4.09	0.00	小于 100 万分之一
SBCAM	25	3.08	244.84	1.00
B-T SBCAM	25	3.98	0.00	小于 100 万分之一

通过表 5 ~ 表 7 的对比可以发现，可采用公式计算的 SBCAM 和 Shapley 方法，其求解复杂度明显要低于需要采用复杂优化方法的核仁解 Nucleolus. 如表 6 所示，文献[26，27]中改进的核仁解求解方法的求解精度与 SBCAM 类似，且其求解速度虽然比传统求解核仁解的方法更快，但与 SBCAM 相比明显更慢. 在表 7 中，通过与 Shapley 方法和最近改进的 Shapley 方法(LPSA-Shapley 和 B-T Shapley)比较，可以发现求解精度与改进的 Shapley 方法相比具有类似的精度偏差，SBCAM 速度要快于 Shapley 方法，但比改进的 Shapley 方法 LPSA-Shapley 和 B-T Shapley 均要明显的更慢，同时发现，结合文献[5]中的二叉树分级策略，可以完全应用到 SBCAM 形成 B-T SBCAM，通过实验发现B-T SBCAM 的求解速度与 B-T Shapley 相近.

通过上述对比可知，SBCAM 的求解速度和质量与当前最新方法相比，均具有较强的竞争力，且可改进性强. 此处，需要再次强调的是，文章提出的 SBCAM 方法最终目的不是近似求解核仁解，而是一种具有自身内涵的成本分摊方法，只是产生的科学依据与核仁解具有相似性，接下来继续分析 SBCAM 的求解结果与核仁解的内在关联.

4.3　SBCAM 与核仁解 Nucleolus 的关联性分析

如上所述 SBCAM 本身是一种新的方法，其原理是求解的分摊方案能让所有子联盟满意度达到最均衡状态，即让均衡度函数 f 取到最小值. 传统的核仁解成本分摊方案其目标是让所有子联盟满意度的非降序排列的字典序值最大，即先令 2^n-1 个子联盟满意度的最小值取到最大的转归解集合为 T_1，然后在 T_1 中令 2^n-1 个子联盟满意度的次小值取到最大的解集合为 T_2，按此思路，最后，当在 T_{k-1} 中令 2^n-1 个子联盟满意度的倒数第 k 大值取到最大的解集合为 T_k，且 T_k 只包含唯一解时，得到核仁解.

其实 SBCAM 和核仁解具有类似的原理内涵，目标均是使子联盟满意度尽量均衡，只是采用的衡量指标不同. 前者通过构造均衡度函数 f 表示子联盟均衡程度，核仁解采用的是子联盟满意度字典序值最大的方式. 为了验证两者的内在关联性，基于成本分摊问题通过随机产生成本分摊方案 x 的方式，分别计算分摊方案 x 对应的均衡度函数 f 值、x 与核仁解的偏差 $Dev\%$ 值(计算方法如式(35)所示). 然后，对比均衡度函数 f 值与 $Dev\%$ 值，分析是否具有高度的相关性. 根据式(15)可知，均衡度函数 f 值是关于成本分摊方案 x 中元素的二次方函数. 根据式(35)可知，偏差 $Dev\%$ 值的计算式是关于成本分摊方案 x 中元素的线性式. 所以，为了更加准确的比较 f 值与 $Dev\%$ 值的相关性，选择 f 值的平方根即 $\sqrt{f}$ 作为比较对象. 另外，由于 $Dev\%$ 值的取值范围为 0% ~100%，在此，也将 $\sqrt{f}$ 值采用式(39)中的方法样转化为一个取值范围在 0% ~100% 的相对数.

$$\alpha\% = \frac{\sqrt{f} - \sqrt{f_{\min}}}{\sqrt{f_{\max}} - \sqrt{f_{\min}}} \times 100\% \tag{39}$$

其中 $\sqrt{f_{\min}}$ 表示均衡函数最小值，显然 SBCAM 求解得到的成本分摊方案 x^* 对应的子联盟均衡函数值最小. $\sqrt{f_{\max}}$ 表示均衡函数最大值，在此取与 SBCAM 求解得到的成本分摊方案 x^* 偏离最大的成本分摊方案 y^* 对应的均衡函数值. 根据式(35)，不难得到成本分摊 y^* 的结果，就是将成本分摊方案 x^* 中最小元素值改为 $c(N)$，其他元素值均改为 0.

此处，分别选择图 5 中的协作配送算例，及表 4 中已发表的 5 篇论文中对应的成本分摊或利益分配算例，共 6 个例子. 每个例子随机生成 2 000个成本分摊方案 x(随机生成的成本分摊方法如式 40 所示)，并分别计算对应的 $\alpha\%$ 和 $Dev\%$ 值，6 个算例分别对应的散点图如图 7 所示.

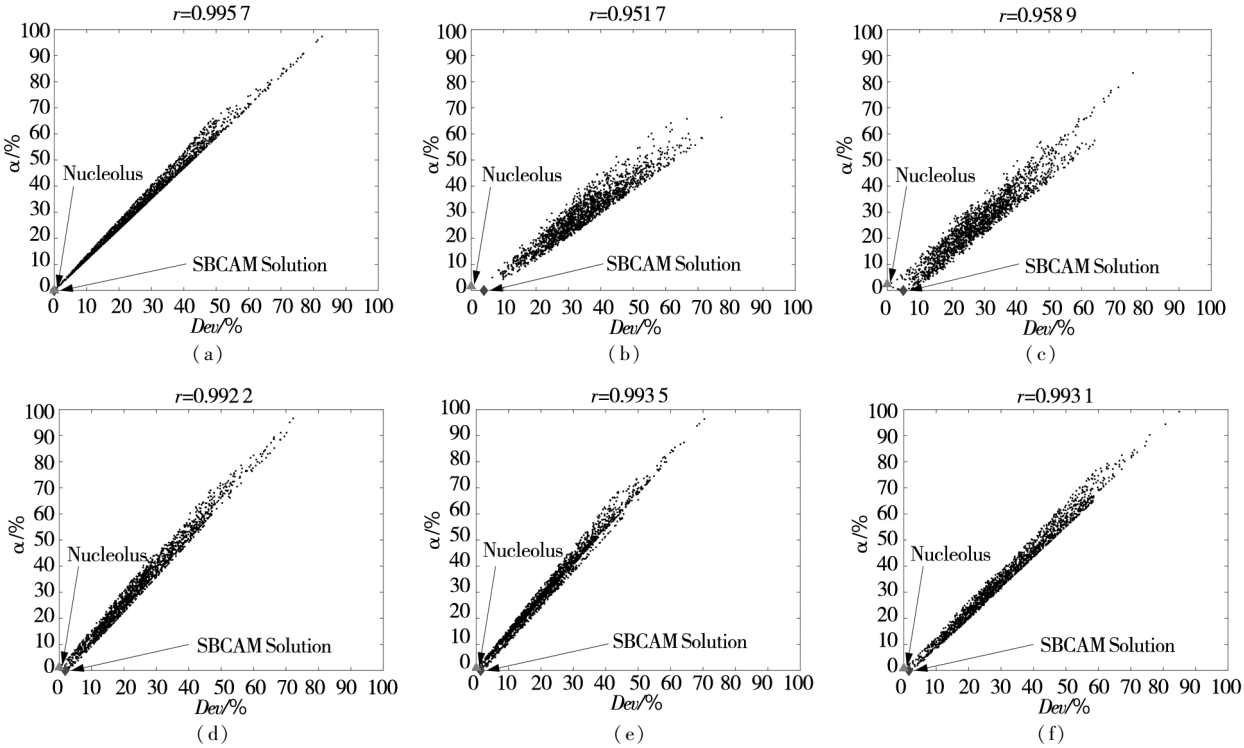

图 7　分摊方案的均衡度 α% 和与 Nucleolus 偏离度 *Dev*% 的相关性分析

Fig. 7 The correlation analysis of α% and *Dev*%

$$\begin{cases}\sigma=(\sigma_1,\sigma_2,\cdots,\sigma_n)\\ 0\leqslant\sigma_i\leqslant 1\\ \omega=\sigma_1+\sigma_2+\cdots+\sigma_n\\ x_i=\dfrac{\sigma_i}{\omega}\times c(N)\\ x=(x_1,x_2,\cdots,x_n)\end{cases}\tag{40}$$

其中 σ 是随机生成的一组数列.

需要指出的是图 7 中的后 5 个算例是表 4 中现有文献中的例子, 在此将利益分配问题统一转换为成本分摊问题, 令 $c(S)=v(N)-v(N/S)$. 另外, r 表示 2 000 对 α% 和 *Dev*% 值的相关系数, 由 6 个算例可知, 该 6 个算例中 α% 和 *Dev*% 的值均高度正相关, 特别是图 7(a),图 7(d),图 7(e)和图 7(f)对应的 4 个算例的相关系数达到了 0.99 以上,另外两个算例的相关系数均超过 0.95. 由图 7 容易发现, 当相关系数 r 越大, SBCAM的求解结果越接近核仁解(图中蓝色菱形越接近原点). 图 7(a)对应的算例 SBCAM 求解结果与核仁解为同一解, 因此代表核仁解的红色三角形和代表 SBCAM 求解方案的蓝色菱形, 在图中原点位置重合.

通过上述分析可知, SBCAM 定义的子联盟满意度均衡的指标函数, 与传统核仁解使子联盟满意度升序排列的字典序值最大的内涵具有高度一致性. 重要的是 SBCAM 方法可以通过公式计算得到, 而传统的核仁解需要经过复杂的优化计算, 并且容易产生错误的计算结果. 因此, SBCAM 不仅具有自身内在含义, 具有独立的科学原理, 可以单独用于求解成本分摊方案, 而且求解结果会非常接近核仁解.

4.4　SBCAM 求解结果的科学内涵及理论属性验证

4.4.1　科学内涵

合作博弈中的 SBCAM 本身是一种新的科学成本分摊方法,求解的分摊方案不仅能让所有子联盟满意度达到最均衡状态, 即使均衡度函数 f 取到最小, 也能够合理地解释成员之间分摊成本量的差别.

在实际中, 成员分摊协作所产生的总成本, 只有当每个成员认可自身分摊的成本量科学合理, 才能保证分摊方案的实施. 然而, 成员确定自身成本分摊量是否合理的主要方式, 通常是通

过与其他成员的分摊量进行比较. 当成员接受自身分摊的成本量与其他成员分摊量的差别时，说明成员认可了成本分摊方案.

例如，图 5 中的 3 个企业协作配送的案例，3 个企业的协作配送总成本为 100，SBCAM 的分摊结果是 $x^*=(50, 35, 15)$. 那么不妨站在企业 2 的角度来评估该方案是否公平，企业 2 就会比较自己为什么比企业 1 少分摊了 15 个成本单位，比企业 3 多分摊了 20 个单位成本？因为，如表 1 所示已知所有子联盟的协作配送成本如下，$c\{3\}=20$，$c\{2\}=40$，$c\{2,3\}=50$，$c\{1\}=50$，$c\{1,3\}=70$，$c\{1,2\}=90$，$c\{1,2,3\}=100$. 首先，分析企业 2 为什么比企业 1 少分摊了 15 个成本，由于 $c\{2\}-c\{1\}=40-50=-10$，$c\{2,3\}-c\{1,3\}=50-70=-20$，那么企业 2 应该比企业 1 少分摊 10 个单位还是 20 个单位？取两者平均值可得到为 -15，即企业 2 应该比企业 1 少分摊 15 个单位，这也相当于式(27)中的 $M_{21}=-15$. 同理可以计算 $M_{23}=20$，因为 $c\{2\}-c\{3\}=40-20=20$，$c\{1,2\}-c\{1,3\}=90-70=20$，取两者平均数可得 $M_{23}=20$. 同理，站在企业 1 和企业 3 的角度可以类似评估 $x^*=(50, 35, 15)$的合理性.

在 SBCAM 的成本分摊方案中，对于任意两个不同成员 i，j，均有 $x_i=x_j+M_{ij}$成立. M_{ij}就是成员 i 应该比成员 j 多分摊的成本量. M_{ij}的计算方法就是，通过计算不同 S 下的 $c(S\cup\{i\})-c(S\cup\{j\})$ 的差值，其中 S 为不包含成员 i，j 的任意子联盟，并将不同 S 下的差值计算算术平均值即得到 M_{ij}. 该计算方式能够非常公平的评估成员 i，j 对子联盟协作成本的贡献差别，所以分摊的结果容易获得成员的认可，能够合理解释成员之间成本分摊量的差别. 因此，SBCAM 分摊方法不仅可以采用公式计算，而且科学内涵能够通过通俗的方式解释，所以容易在实际应用中进行推广.

4.4.2　SBCAM 分摊结果的理论属性验证

由文章第 3.2 节可知，SBCAM 的成本分摊方案满足总体有效性、唯一性、可加性、策略等价相对不变性、一致性、匿名性和可比性，共 7 个合理属性. 现通过一个小规模成本分摊算例验证该 7 个属性，算例的相关数据如表 8 所示.

表 8　3 个成本分摊例子

Table 8 Three instances of cost allocation

序号 j	子联盟 S^j	B^j	$a(S^j)$	$c_1(S^j)$	$c_2(S^j)=2c_1(S^j)+a(S^j)$	$c(S^j)=c_1(S^j)+c_2(S^j)$
1	{3}	[0, 0, 1]	10	35	80	115
2	{2}	[0, 1, 0]	10	35	80	115
3	{2,3}	[0, 1, 1]	20	60	140	200
4	{1}	[1, 0, 0]	10	45	100	145
5	{1,3}	[1, 0, 1]	20	70	160	230
6	{1,2}	[1, 1, 0]	20	70	160	230
7	{1,2,3}	[1, 1, 1]	30	100	230	330

采用 SBCAM 方法分别对 $a(S)$、$c_1(S)$、$c_2(S)=2c_1(S)+a(S)$和 $c(S)=c_1(S)+c_2(S)$对应的成本分摊问题进行求解，可得到分摊的结果分别为 $a=(10, 10, 10)$、$y^*=(40, 30, 30)$、$z^*=(90, 70, 70)$和 $x^*=(130, 100, 100)$. 因为$a(S)$属于可加合作博弈，成员之间进行合作不会降低任何成本，即联盟的成本会等于成员不协作的成本总和，因此最终合理分摊结果显然就是每个成员不协作的成本(10, 10, 10)，SBCAM 求解的结果即为该结果. 另外，根据上述算例求解 SBCAM 结果显然会满足合理属性 1 总体有效性和合理属性 2 唯一性；根据表 8 可知，由于 $c(S)=c_1(S)+c_2(S)$，求解结果表明 $x^*=y^*+z^*$，验证了合理属性 3 可加性；由于 $c_2(S)=2c_1(S)+a(S)$，因此 $c_1(S)$和 $c_2(S)$其实为策略等价合作博弈，其分摊结果 $z^*=2y^*+a$，验证了合理属性 4 策略

等价相对不变性；在该 3 个成本分摊问题中，成员 2 和成员 3 在任意子联盟中的贡献均相同，因此在 x^*，y^*，z^* 中成员 2 和成员 3 的分摊结果均相同，这也验证了 SBCAM 分摊结果遵循合理属性 5 一致性. 通过本实例也不难验证SBCAM可满足属性 6，如将成员 1 和成员 3 互换身份方式改变子联盟的成本，然后采用 SBCAM 可求解得到 $x^* = (100, 100, 130)$，即等价于分摊结果中成员 1 和成员 3 互换了身份，因此合理属性 6 匿名性成立. 最后，可以根据 SBCAM 计算公式求解得到任意两个不同成员的成本贡献差别 M_{ij} 值，比如 M_{12} 等于 30，因此可以解释成员 1 要比成员 2 多分摊 30 个单位，故验证了合理属性 7 可比性.

需要指出的是核仁解和 Shapley 也能满足上述属性的绝大部分，但 SBCAM 在所有成本分摊方法中是唯一能够满足合理属性 7 可比性的方法.

5 结束语

文章以协作配送成本分摊为应用背景，基于使子联盟满意度越均衡，分摊方案越公平的原理，提出了一个新的成本分摊方法 SBCAM. SBCAM 与核仁解原理非常相近，并且 SBCAM 的求解结果非常接近核仁解，重要的是 SBCAM 可以通过公式计算得到，求解复杂度远远低于求解核仁解的现有方法. 主要结论如下所示.

1）文章所提出的 SBCAM 可以采用公式方式计算成本分摊结果，其求解速度能够应用于求解时效性要求高的依托平台协作配送模式成本分摊问题，其中求解协作企业数量达 30 数量级的大规模协作配送成本分摊问题，可在合理时间内完成，如果结合二叉树 Binary Tree 提速策略，可在线实现大规模甚至是超大规模的成本分摊问题的求解.

2）所有子联盟的满意度与对应补联盟的满意度越接近，所有子联盟满意度的序列值趋向越大，当所有子联盟与补联盟满意度相等时，子联盟满意度序列值达到理论最大值.

3）SBCAM 求解结果与核仁解仅有较小偏差，偏差一般在 5% 左右，对于符合定理 3 的成本分摊问题 SBCAM 可以直接求解得到核仁解.

4）SBCAM 可通过公式计算，计算速度是采用传统方法求解核仁解的数万倍，成本分摊方案满足总体理性、唯一性、可加性、策略等价相对不变性、一致性、匿名性和可比性等众多公平特点.

SBCAM 方法不仅能够用于求解或近似求解核仁解，更重要的是 SBCAM 可以求解任何支付可转移的合作博弈成本分摊问题. 当然，SBCAM 也存在自身的不足，如具有与 Shapley 类似的缺点，从理论上不能保证问题核心解存在时，其求解结果属于核心解. 在今后的研究中将继续研究提升 SBCAM 综合性能的策略，以从理论上进一步提高 SBCAM 的合理性及有效性.

参 考 文 献：

[1] 袁 韵，徐 戈，陈晓红，等. 城市交通拥堵与空气污染的交互影响机制研究 —— 基于滴滴出行的大数据分析[J]. 管理科学学报，2020，23(2)：54 – 73.
Yuan Yun，Xu Ge，Chen Xiaohong，et al. Study on the interactive mechanism of urban traffic congestion and air pollution：A big data analysis based on DiDi Chuxing[J]. Journal of Management Sciences in China，2020，23(2)：54 –73.（in Chinese）

[2] Gansterer M，Hartl R F. Collaborative vehicle routing：A survey[J]. European Journal of Operational Research，2018，268(1)：1 – 12.

[3] Chen M，Zhang S，Zhang W，et al. Collaborative vehicle routing problem with rough location using extended ant colony optimization algorithm[J]. Journal of Intelligent & Fuzzy Systems，2019，37(2)：2385 – 2402.

[4] Basso F D，Amours S，Rnnqvist M，et al. A survey on obstacles and difficulties of practical implementation of horizontal collaboration in logistics[J]. International Transactions in Operational Research，2019，(3)：775 – 793.

[5] 饶卫振，朱庆华，金　淳，等. 协作车辆路径成本分摊问题的 B-T Shapley 方法[J]. 管理科学学报，2019，22(1)：107 – 126.
Rao Weizhen，Zhu Qinghua，Jin Chun，et al. A Binary-Tree Shapely method for cost sharing of the collaborative vehicle routing problem[J]. Journal of Management Sciences in China，2019，22(1)：107 – 126. (in Chinese)

[6] Guajardo M，Ronnqvist M. A review on cost allocation methods in collaborative transportation[J]. International Transactions in Operational Research，2016，23(3)：371 – 392.

[7] Shapley L S. A value for n-person games[J]. Contributions to the Theory of Games，1953，28：307 – 317.

[8] Schmeidler D. The nucleolus of a characteristic function game[J]. SIAM Journal on Applied Mathematics，1969，17(6)：1163 – 1170.

[9] Maschler M，Peleg B，Shapley L S. Geometric properties of the kernel，nucleolus，and related solution concepts[J]. Mathematics of Operations Research，1979，(4)：303 – 338.

[10] Liu J C，Sheu J B，Li D F，et al. Collaborative profit allocation schemes for logistics enterprise coalitions with incomplete information[J]. Omega，2020：102237.

[11] Baïou M，Barahona F. On the nucleolus of shortest path games[J]. Algorithmic Game Theory，2017：55 – 66.

[12] Farsani E A，Abyaneh H A，Abedi M，et al. A novel policy for LMP calculation in distribution networks based on loss and emission reduction allocation using nucleolus theory[J]. IEEE Transactions on Power Systems，2016，31(1)：143 –152.

[13] 刘家财，李登峰，胡勋锋. 区间值最小二乘核仁解及在供应链合作利益分配中的应用[J]. 中国管理科学，2017，(12)：78 – 87.
Liu Jiacai，Li Dengfeng，Hu Xunfeng. Interval-valued least square nucleolus and its application in cooperative profit allocation of supply chain[J]. Chinese Journal of Management Science，2017，(12)：78 – 87. (in Chinese)

[14] Gow S H，Thomas L C. Interchange fees for bank ATM networks[J]. Naval Research Logistics，1998，45(4)：407 – 417.

[15] Albizuri M J，Echarri J M，Zarzuelo J M. A non-cooperative mechanism yielding the nucleolus of airport problems[J]. Group Decision and Negotiation，2018，27(1)：153 – 163.

[16] Kohlberg E. The nucleolus as a solution of a minimization problem[J]. SIAM Journal on Applied Mathematics，1972，23(1)：34 – 39.

[17] Owen G. A note on the nucleolus[J]. International Journal of Game Theory，1974，3(2)：101 – 103.

[18] Puerto J，Perea F. Finding the nucleolus of any n-person cooperative game by a single linear program[J]. Computers & Operations Research，2013，40(10)：2308 – 2313.

[19] Behringer F A. A simplex based algorithm for the lexicographically extended linear maxmin problem[J]. European Journal of Operational Research，1981，(3)：274 – 283.

[20] Drăgan I. A procedure for finding the nucleolus of a cooperative n person game[J]. Z. Oper. Res. Ser. A-B，1981，(5)：119 – 131.

[21] Potters J A M，Reijnierse J H，Ansing M. Computing the nucleolus by solving a prolonged simplex algorithm[J]. Mathematics of Operations Research，1996，21(3)：757 – 768.

[22] Derks J，Kuipers J. Implementing the Simplex Method for Computing Prenucleolus of Transferable Utility Games[M]. Maastricht：Maastricht University，1996.

[23] Hallefjord A，Helming R，Jornsten K. Computing the nucleolus when the characteristic function is given implicitly：A constraint generation approach[J]. International Journal of Game Theory，1995，24(4)：357 – 372.

[24] Fromen B. Reducing the number of linear programs needed for solving the nucleolus problem of n-person game theory[J]. European Journal of Operational Research，1997，98(3)：626 – 636.

[25] Nguyen T，Thomas L. Finding the nucleoli of large cooperative games[J]. European Journal of Operational Research，2016，248(3)：1078 – 1092.

[26] Perea F, Puerto J. A heuristic procedure for computing the nucleolus[J]. Computers and Operations Research, 2019, 112(12): 104764.

[27] Lu W, Quadrifoglio L. Fair cost allocation for ridesharing services-modeling, mathematical programming and an algorithm to find the nucleolus[J]. Transportation Research: Part B, 2019, 121: 41 – 55.

[28] Tae H, Kim B, Park J. Finding the nucleolus of the vehicle routing game with time windows[J]. Applied Mathematical Modelling, 2020, 80(4): 334 – 344.

[29] Chardaire P. The core and nucleolus of games: A note on a paper by Gothe-Lundgren[J]. Mathematical Programming, 2001, 90(1): 147 – 151.

[30] Guajardo M, Jornsten K. Common mistakes in computing the nucleolus[J]. European Journal of Operational Research, 2015, 241(3): 931 – 935.

[31] Faigle U, Kern W, Kuipers J. On the computation of the nucleolus of a cooperative game[J]. International Journal of Game Theory, 2001, 30(1): 79 – 98.

[32] Leng M, Mahmut P. Analytic solution for the nucleolus of a three-player cooperative game[J]. Naval Research Logistics Quarterly, 2010, (7): 667 – 672.

[33] Zhang Q, Yang Z, Gui B. Coalitional game with fuzzy payoffs and credibilistic nucleolus[J]. Journal of Intelligent & Fuzzy Systems, 2017, 32(1): 1 – 9.

[34] Lin J, Zhang Q. The least square B-nucleolus for fuzzy cooperative games[J]. Journal of Intelligent & Fuzzy Systems, 2016, 30(1): 279 – 289.

[35] Lin J, Zhang Q. The L-nucleolus and I-Shapley value for cooperative games under incomplete information[J]. Trans. Beijing Inst. Tech, 2015, 35(7): 767 – 770.

[36] Cabral L, Pacheco-De-Almeida G. Alliance formation and firm value[J]. Management Science, 2019, 65(2): 879 – 895.

[37] Huang Y, Zhao L, Powell W B, et al. Optimal learning for urban delivery fleet allocation[J]. Transportation Science, 2019, 53(3): 623 – 641.

[38] Hu C, Tsay M, Yeh C. A study of the nucleolus in the nested cost-sharing problem: Axiomatic and strategic perspectives[J]. Games and Economic Behavior, 2018, 109: 82 – 98.

[39] Sziklai B, Fleiner T, Solymosi T. On the core and nucleolus of directed acyclic graph games[J]. Mathematical Programming, 2017, 163(1 – 2): 243 – 271.

[40] 饶卫振, 朱庆华, 刘从虎. 在线组建协作配送联盟中企业成本节约相对量估算方法研究[J]. 系统工程理论与实践, 2019, 39(3): 659 – 672.
Rao Weizhen, Zhu Qinghua, Liu Conghu. An estimation method for computing cost saving percentage of distribution enterprises in collaborative environment: Used for forming collaborative distribution coalition online[J]. Systems Engineering: Theory & Practice, 2019, 39(3): 659 – 672. (in Chinese)

[41] Sakawa M, Nishizaki I, Uemura Y. Fuzzy programming and profit and cost allocation for a production and transportation problem[J]. European Journal of Operational Research, 2001, 131(1): 1 – 15.

[42] Krus L, Bronisz P. Cooperative game solution concepts to a cost allocation problem[J]. European Journal of Operational Research, 2000, 122(2): 258 – 271.

[43] Satya R P, Radhakrishna C. Use of cooperative game theory concepts for loss allocation in multiple transaction electricity markets[J]. Journal of Electrical Systems, 2009, 5(1): P6.

[44] Oh S C, Shin J. A semantic e-kanban system for network-centric manufacturing: Technology, scale-free convergence, value and cost-sharing considerations[J]. International Journal of Production Research, 2012, 50(19): 5292 – 5316.

[45] Lemaire J. Cooperative game theory and its insurance applications[J]. Astin Bulletin, 1991, 21(1): 17 – 40.

[46] Le P H, Nguyen T D, Bekta T. Efficient computation of the Shapley value for large-scale linear production games[J]. Annals of Operations Research, 2020, 287(2): 761 – 781.

Fair and effective cost-sharing method for collaborative distribution based on a third-party platform

RAO Wei-zhen[1], *XU Feng*[1], *ZHU Qing-hua*[2*], *WEI Fang-fang*[3], *LIU Cong-hu*[3]

1. College of Economics and Management, Shandong University of Science and Technology, Qingdao 266590, China;
2. Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai 200030, China;
3. Sino-US Global Logistics Institute, Shanghai Jiao Tong University, Shanghai 200030, China

Abstract: It is crucial to develop a fair and effective cost-sharing solution within reasonable time for collaborative distribution based on a third-party platform. The nucleolus solution is recognized as a commonly-accepted fair allocation solution, but it needs complicated optimization calculation. In this paper, an efficient method is proposed to calculate the nucleolus solution approximately and quickly by a formula, which finds that for any sub-coalitions with a cost-sharing solution x that satisfies the overall rationality, the sum of $e(S, x)$, or the sum of satisfaction, of all $2^n - 1$ (n is the player number of the big coalition N) sub-coalitions S (S is a subset of N) is a constant, and that the sum of satisfaction of any sub-alliance S and complementary alliance $N\backslash S$ with different x is $e(S, x)$ plus $e(N\backslash S, x)$, which is equal to a constant L_S. Based on the rule that the more balanced the sub-coalitions' satisfaction $e(S, x)$, the more reasonable the cost-sharing solution x, this paper constructs a function $f(x) = \sum [e(S, x) - 0.5L_S]^2$ which quantifies the satisfaction balance of all sub-alliances with the cost-sharing solution x. Apparently, the smaller the value of $f(x)$, the more balanced the sub-coalitions' satisfaction $e(S, x)$. It is proved that there exists a cost-sharing solution x^* that minimizes function $f(x)$, and that x^* satisfies many reasonable attributes of cost-sharing solutions, such as overall rationality, uniqueness, additivity, relative invariance with respect to strategic equivalence, consistency, anonymity, and comparability. Last, many cost-sharing instances from previous publications are calculated by the method proposed in this paper and the traditional methods for finding nucleolus, respectively. The results demonstrate that the calculation speed of the proposed method is tens of thousands of times faster than that of the traditional linear programming for finding the nucleolus solution. The proposed method is also superior to the latest effective algorithm for calculating the nucleolus solution and the Shapley value. In addition, the average deviation is only about 5% between the result calculated by the proposed method and the nucleolus solution. More importantly, the proposed method is able to explain its connotation by scientific principles and can be applied to any cost-sharing problems of cooperative games with transferable payment.

Key words: collaborative distribution; cost-sharing method; satisfaction; nucleolus

附录 1

定理 3 的证明过程

设所有子联盟 $S(S\subset N, S\neq\varnothing)$ 与补联盟 $N\backslash S$ 的满意度之为 L_S，显然 $L_S = L_{N\backslash S}$，且满足总体理性的分摊方案 x，子联

盟 S 对其满意度占 L_S 的比率为 $\alpha(S, x)$，即 $e(S, x) = \alpha(S, x) \times L_S$，故 $\alpha(S, x) + \alpha(N\backslash S, x) = 1$.

证明定理 3 等价于证明当存在一个满足总体理性的分摊方案 x^*，能够使得所有的 $\alpha(S, x^*) = 0.5$，那么 x^* 为核仁解. 根据核仁解的内涵可知，只要证明 x^* 比任意其他 x 的总体满意度都大.

不妨将不同分摊方案的 $\alpha(S, x^*)$ 根据对应的 L_S 值从小到大的顺序排列，根据定理 3 已知信息可知 $[\alpha(S_1, x^*), \alpha(N\backslash S_1, x^*), \alpha(S_2, x^*), \alpha(N\backslash S_2, x^*), \cdots, \alpha(Sw, x^*), \alpha(N\backslash Sw, x^*)] = [0.5, 0.5, 0.5, 0.5, \cdots, 0.5, 0.5]$，其中 $w = 2^{n-1} - 1$，且 $L_{Si} \leqslant L_{Si+1}$. 令 $x \neq x^*$，那么 $[\alpha(S_1, x), \alpha(N\backslash S_1, x), \alpha(S_2, x), \alpha(N\backslash S_2, x), \cdots, \alpha(S_w, x), \alpha(N\backslash S_w, x)]$ 中至少存在一对子联盟与补联盟的满意度比值不等于 0.5，不妨假设第 $i(i \geqslant 1)$ 对的 $\alpha(S_i, x)$ 和 $\alpha(N\backslash S_i, x)$ 是首个不等于 0.5 比值，且 $\alpha(S_i, x) < 0.5$，$\alpha(N\backslash S_i, x) > 0.5$.

那么分别将 x^* 和 x 使子联盟的满意度从小到大排序可知，x^* 的前 $2(i-1)$ 个最小满意度必然为 $[0.5L_{S1}, 0.5L_{S1}, 0.5L_{S2}, 0.5L_{S2}, \cdots, 0.5L_{Si-1}, 0.5L_{Si-1}]$，$x$ 的前 $2(i-1)$ 个最小满意度，因为存在比值小于 0.5 的情况，所以不大于 $[0.5L_{S1}, 0.5L_{S1}, 0.5L_{S2}, 0.5L_{S2}, \cdots, 0.5L_{Si-1}, 0.5L_{Si-1}]$.

情况 1 当 x 的其他子联盟满意度存在小于 $0.5L_{Si-1}$ 的情况时，那么可以直接确定 x^* 的总体满意度必然大于 x. 定理 3 成立. 证毕.

情况 2 当 x 的其他子联盟满意度不存在小于 $0.5L_{Si-1}$ 的情况时，那么 x 的前 $2(i-1)$ 个最小满意度也等于 $[0.5L_{S1}, 0.5L_{S1}, 0.5L_{S2}, 0.5L_{S2}, \cdots, 0.5L_{Si-1}, 0.5L_{Si-1}]$. x^* 的第 $2(i-1)+1 = 2i-1$ 个满意度为 $0.5L_{Si}$，然而，x 的第 $2(i-1)+1 = 2i-1$ 个满意度为 $\min\{\alpha(S_i, x)L_{Si}, \alpha(N\backslash S_i, x)L_{Si}, \alpha(S_{i+1}, x)L_{Si}, \alpha(N\backslash S_{i+1}, x)L_{Si}, \cdots, \alpha(S_w, x)L_{Sw}, \alpha(N\backslash S_w, x)L_{Sw}\} \leqslant \alpha(S_i, x)L_{Si}$，又 $\alpha(S_i, x) < 0.5$，因此 x^* 的总体满意度必然大于 x. 定理 3 成立. 证毕.

附录 2

一阶偏导数等式化简过程.

$$\frac{\partial f}{\partial x_i} = -2\sum_{S\in S_{in}}\left[c(S) - x(S) - \frac{L_S}{2}\right] + 2\sum_{S\in S_{ni}}\left[c(S) - x(S) - \frac{L_S}{2}\right] = 0,\ 1 \leqslant i \leqslant n-1$$

$$\Leftrightarrow -\sum_{S\in S_{in}}[2c(S) - 2x(S) - L_S] + \sum_{S\in S_{ni}}[2c(S) - 2x(S) - L_S] = 0$$

$$\because L_S = c(S) + c(N\backslash S) - c(N)$$

$$\therefore \Rightarrow \sum_{S\in S_{ni}}[c(S) - c(N\backslash S) - 2x(S)] - \sum_{S\in S_{in}}[c(S) - c(N\backslash S) - 2x(S)] = 0$$

将上述等式拆开可得

$$\sum_{S\in S_{ni}}[c(S) - c(N\backslash S) - 2x(S)] - \sum_{S\in S_{in}}[c(S) - c(N\backslash S) - 2x(S)] = 0$$

$$\Leftrightarrow \sum_{S\in S_{ni}}[c(S) - c(N\backslash S)] - \sum_{S\in S_{in}}[c(S) - c(N\backslash S)] - 2\sum_{S\in S_{ni}}x(S) + 2\sum_{S\in S_{in}}x(S) = 0$$

$$\because \sum_{S\in S_{ni}}x(S) = 2^{n-2}\cdot x_n + 2^{n-3}[x(N) - x_i - x_n] = 2^{n-2}\cdot x_n + 2^{n-3}[c(N) - x_i - x_n]$$

$$\sum_{S\in S_{ni}}x(S) = 2^{n-2}\cdot x_i + 2^{n-3}[c(N) - x_i - x_n]$$

$$\therefore \Rightarrow 2^{n-1}(x_i - x_n) = \sum_{S\in S_{in}}[c(S) - c(N\backslash S)] - \sum_{S\in S_{ni}}[c(S) - c(N\backslash S)]$$

$$\because \sum_{S\in S_{in}}c(N\backslash S) = \sum_{S\in S_{ni}}c(S), \sum_{S\in S_{in}}c(S) = \sum_{S\in S_{ni}}c(N\backslash S)$$

$$\therefore \Rightarrow 2^{n-1}(x_i - x_n) = 2\left[\sum_{S\in S_{in}}c(S) - \sum_{S\in S_{ni}}c(S)\right]$$

$$\Rightarrow x_i - x_n = \frac{\sum_{S\in S_{in}}c(S) - \sum_{S\in S_{ni}}c(S)}{2^{n-2}}$$