Cost Sharing and Strategyproof Mechanisms for Set Cover Games*

Xiang-Yang Li† Zheng Sun‡ Weizhao Wang Wei Lou§

January 15, 2009

Abstract

We develop for set cover games several general cost-sharing methods that are approximately budget-balanced, in the core, and/or group-strategyproof. We first study the cost sharing for a single set cover game, which does not have a budget-balanced mechanism in the core. We show that there is no cost allocation method that can always recover more than $\frac{1}{\ln n}$ of the total cost and in the core. Here n is the number of all players to be served. We give a cost allocation method that always recovers $\frac{1}{\ln d_{max}}$ of the total cost, where d_{max} is the maximum size of all sets. We then study the cost allocation scheme for all induced subgames. It is known that no cost sharing scheme can always recover more than $\frac{1}{n}$ of the total cost for every subset of players. We give an efficient cost sharing scheme that always recovers at least $\frac{1}{2n}$ of the total cost for every subset of players and furthermore, our scheme is cross-monotone. When the elements to be covered are selfish agents with privately known valuations, we present a strategyproof charging mechanism, under the assumption that all sets are simple sets; further, the total cost of the set cover is no more than $\ln d_{max}$ times that of an optimal solution. When the sets are selfish agents with privately known costs, we present a strategyproof payment mechanism to them. We also show how to *fairly* share the payments to all sets among the elements.

Keywords: Set cover, selfish agent, mechanism design, pricing.

---

*The preliminary version of the paper was published at STACS 2005 [19].

†Xiang-Yang Li is with Department of Computer Science, Illinois Institute of Technology, Chicago, IL. Email: `xli@cs.iit.edu`. The research of Xiang-Yang Li is partially supported by NSF CNS-0832120, NSF CCF-0515088, National Basic Research Program of China (973 Program) under grant No. 2006CB30300, the National High Technology Research and Development Program of China (863 Program) under grant No. 2007AA01Z180, Hong Kong RGC HKUST 6169/07, the RGC under Grant HKBU 2104/06E. Li and Lou is also partially supported by CERG under Grant PolyU-5232/07E.

‡Zheng Sun and Weizhao Wang are with Google. Emails: {`sunzheng`, `weizhao.wang`}`@gmail.com`

§Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. Email: `csweilou@comp.polyu.edu.hk`

1 Introduction

In designing efficient, centralized or distributed algorithms and network protocols, the computational agents are typically assumed to be either *correct/obedient* or *faulty* (also called adversarial). Here agents are said to be *correct/obedient* if they follow the protocol correctly, *i.e.*, act as instructed. In contrast, economists design market mechanisms in which it is assumed that agents are *rational*, *i.e.*, they respond to well-defined incentives and will deviate from the protocol only if it improves their gain. Designing efficient algorithms with provable performances for a set of autonomous agents gained considerable research attentions recently. In this paper, we study the cost sharing and designing of strategyproof mechanisms for set cover games.

1.1 Generalized Set Cover Problem

Let $U = \{e_1, e_2, \cdots, e_n\}$ be a finite set, and let $\mathcal{S} = \{S_1, S_2, \cdots, S_m\}$ be a collection of multisets (or *sets* for short) of U. For each $e_i \in U$ and each $S_j \in \mathcal{S}$, we denote the multiplicity of e_i in S_j by $k_{j,i}$. Each S_j is associated with a cost c_j. For any $\mathcal{X} \subseteq \mathcal{S}$, let $C(\mathcal{X})$ denote the total costs of the sets in $\mathcal{X}$, *i.e.*, $C(\mathcal{X}) = \sum_{S_j \in \mathcal{X}} c_j$. For a given $k > 0$ and a set of *element coverage requirements* $\{r_1, r_2, \cdots, r_n\}$, a *k-partial-cover* $\mathcal{C}$ is defined to be a subset $\{S_{j_1}, S_{j_2}, \cdots, S_{j_l}\}$ of $\mathcal{S}$ such that $\sum_{i=1}^{n} \min\{r_i, \sum_{t=1}^{l} k_{j_t,i}\} \geq k$. The *generalized set cover problem* is to compute an optimum k-partial-cover $\mathcal{C}_{opt}$ with the minimum cost $C(\mathcal{C}_{opt})$.

This problem becomes the traditional multicover problem [4, 7] when we set $k = \sum_{i=1}^{n} r_i$ and $k_{j,i} = 1$ for all S_j and e_i, as each element e_i should be *fully* covered and each set S_j is a simple set. When we set $r_i = 1$, it becomes the traditional partial cover problem [29]. This problem is therefore a natural extension of the classic set cover problem by allowing partial cover, multiset, and element coverage requirement greater than 1. Accordingly, the greedy algorithm for this problem is a combination of the algorithms designed for partial cover and multicover [4, 7, 29].

1.2 Set Cover Game

Consider the following scenario: a company can choose from a set of service providers $\mathcal{S} = \{S_1, S_2, \cdots, S_m\}$ to provide services to a set of service receivers $U = \{e_1, e_2, \cdots, e_n\}$.

- With a fixed cost c_j, each service provider S_j can provide services to a fixed subset of service receivers.
- There may be a limit $k_{j,i}$ on the number of units of service that a service provider S_j can provide to a service receiver e_i. For example, each service provider may be a cargo company that is transporting goods to various cities (the service receivers), and the amount of goods that can be transported to a particular city daily is limited by the number of trains/trucks that are going to that city everyday.
- Each service receiver e_i may have a limit r_i on the number of units of service that it desires to receive (and is willing to pay for).
- There may be a limit k on the total number of units of service that the service providers shall provide to the service receivers. For example, a manufacturer

company hires various cargo companies to distribute the products to various cities daily, and the total number of units of service required is determined by the daily production of the manufacturer company.

The problem can be modeled by the generalized set cover problem defined in Subsection 1.1. There may be different types of games according to various conditions:

1. Each service receiver e_i has to receive at least r_i units of service, and the costs incurred by the service providers will be shared by the service receivers.
2. Each service receiver e_i declares a bid $b_{i,r}$ for the r-th unit of service it shall receive, and is willing to pay for it only if the assigned cost is at most $b_{i,r}$.
3. Each service provider S_j declares a cost c_j, and is willing to provide the service only if the payment received is at least c_j.

There are different algorithmic issues for these games. For example, for Game 1, we shall define a cost allocation method so that every subset of service receivers feel that the total cost they need to pay is "fair" according to certain criteria. For Games 1 and 2, the cost allocation method, by charging service receivers, needs to recover (either entirely or a constant fraction of) the total cost of the chosen service providers. For Games 2 and 3, we need a mechanism (for determining costs charged to service receivers and payments paid to service providers) that can guarantee that the players are truthful with their declaration of bids/costs.

1.3 Terminologies

Cost Sharing: We first study how we share the total cost of the selected service providers among the service receivers such that some fairness criteria are met. Let $\varkappa(T)$ be the cost of a set cover for a subset of service receivers T. Let $\xi(i,T)$ be the shared cost of the service receiver e_i by a cost sharing method ξ. A number of properties could be desired for a cost sharing method. We list a few here.

1. **Budget-Balanced:** A cost sharing method is called *budget-balanced* if $\sum_{e_i \in T} \xi(i,T) = \varkappa(T)$. Obviously, there are many budget-balanced cost sharing methods.
2. **Fair:** A further criterion is that the sharing method should be *fair*. While the definition of budget-balance is straightforward, defining fairness is more subtle: many fairness concepts were proposed in the literature, such as *max-min* [21], *min-max* [27], *core* and *bargaining set* [24]. In this paper, we study fair cost sharing using the concept of *core*. A sharing mechanism ξ is in the core if, for any subset $T_1 \subseteq T$ of players, the total shared cost $\sum_{e_i \in T_1} \xi(i,T)$ is at most the minimum cost of all subsets of service providers covering T_1.
3. **Cross-Monotone:** The last criterion for a cost sharing method is *cross-monotone*: $\xi(i,T_1) \le \xi(i,T_2)$ for any two subsets T_1 and T_2 with $T_1 \supseteq T_2$.

It is easy to show that there is no cost sharing method that can simultaneously achieve all these three criteria: budget-balance, core and cross-monotone. We thus relax the budget-balance criterion to α-budget-balance: $\alpha \cdot \varkappa(T) \le \sum_{e_i \in T} \xi(i,T) \le \varkappa(T)$.

Mechanism Design: In addition to fair cost sharing, another important task is to design greedy set cover methods that are cognizant of the fact that the service providers or the service receivers are selfish and rational. By "selfish," we mean that they only care about their own benefits without consideration for the global performances or fairness issues. By "rational," we mean that when the methods of computing the output for the set cover game are instituted, they will always choose their actions to maximize their benefits. The study of selfish and rational agents participating in a cooperative or non-cooperative game is central to game theory.

Two fundamental concepts in game theory are *Nash Equilibrium* and *dominant strategy*. Assume that there are n players. Given a set of actions $a = (a_1, a_2, \cdots, a_n)$, where player i chooses the action a_i, let $u(a) = (u_1(a), u_2(a), \cdots, u_n(a))$ be the payoffs vector: $u_i(a)$ is the payoff (or called profit, benefit) to the player i. An action vector a is called a *Nash Equilibrium* if no player can unilaterally switch its action to improve its benefit when the actions of other players are fixed. An action a_i is called a *dominant strategy* for player i if it maximizes its payoff regardless of the actions chosen by other players.

1.4 Our Results

We first present a cost sharing method that is $\frac{1}{\ln d_{max}}$-budget-balanced and in the core, where d_{max} is the maximum set size. The bound $\frac{1}{\ln d_{max}}$ is tight. We also present a cost sharing method that is $\frac{1}{2n}$-budget-balanced, in the core, and cross-monotone, which is almost the optimum [16].

We then design greedy set cover methods that are cognizant of the fact that the service providers or the service receivers are selfish and rational. When the elements to be covered are selfish agents with privately known valuations, we present a strategyproof charging mechanism, under the assumption that all sets are simple sets, such that each element maximizes its profit when it reports its valuation truthfully; further, the total cost of the set cover is no more than $\ln d_{max}$ times that of an optimal solution. When the sets are selfish agents with privately known costs, we present a strategyproof payment mechanism in which each set maximizes its profit when it reports its cost truthfully. We also show how to *fairly* share the payments to all sets among the elements.

1.5 Organization of Paper

The remainder of the paper is organized as follows. In Section 2, we give the exact definitions for fair cost sharing and mechanism design. In Section 3, we study how to fairly share the cost of the service providers among the covered service receivers when the receivers must receive the service. We then show in Section 4 how to charge the cost of service providers to the selfish service receivers when each receiver has a valuation on the r-th cover received. We then show in Section 5 how we compensate the service providers, when they are selfish and each has a privately known cost, such that each service provider maximizes its benefit when it declares its true cost. We conclude our paper in Section 6.

2 Preliminaries and Related Work

2.1 Preliminaries

Algorithm Mechanism Design: A standard economic model for the design and analysis of scenarios in which the participants act according to their own self-interests is as follows. Assume that there are n agents. Each agent i, for $i \in \{1, \cdots, n\}$, has some *private* information t_i, called its *type*. All agents' types define a type vector $t = (t_1, t_2, \cdots, t_n)$. A mechanism defines, for each agent i, a set of strategies A_i. For each strategy vector $a = (a_1, \cdots, a_n)$, *i.e.*, agent i plays a strategy $a_i \in A_i$, the mechanism computes an *output* $o = \mathcal{O}(a_1, \cdots, a_n)$ and a *payment* vector $\mathcal{P}(a) = (p_1, \cdots, p_n)$, where $p_i = \mathcal{P}_i(a_1, \cdots, a_n)$ is the amount of money given to the participating agent i. For each possible output o, agent i's preferences are given by a valuation function v_i that assigns a real monetary number $v_i(t_i, o)$ to output o. Let $u_i(t_i, o(a), p_i(a))$ denote the *utility* of agent i at the outcome (o, p) of the game, given its type t_i and strategy profile $a = (a_1, \cdots, a_n)$ selected by all agents. A common assumption in mechanism design literature, and one which we will follow in this paper, is that agents are *rational* and have quasi-linear utility functions. The utility function is *quasi-linear* if $u_i(t_i, o) = v_i(t_i, o) + p_i(t)$. An agent is called *rational* if it always adopts its best strategy (called *dominant strategy*) that maximizes its utility regardless of what other agents do.

It is well-known that it suffices to design a *direct-revelation* mechanism in which the only actions available to agents are to make direct claims about their preferences v_i to the mechanism. A mechanism is *incentive compatible* (IC) if reporting valuation truthfully is a dominant strategy. Another very common requirement in the literature for mechanism design is the so called *individual rationality* or *voluntary participation*: the agent's utility of participating in the output of the mechanism is not less than the utility of the agent if it did not participate at all. For convenience, let $t|^i b = (t_1, \cdots, t_{i-1}, b, t_{i+1}, \cdots, t_n)$, *i.e.*, each agent $j \neq i$ reports its type t_j except that the agent i reports type b. Direct revelation implies that the actions by agents are to report its type (although they may report falsely). Then, IC implies that, for each agent i, $v_i(t_i, o(t)) + p_i(t) \geq v_i(t_i, o(t|^i b)) + p_i(t|^i b)$; and IR implies that, for each agent i, $v_i(t_i, o(t)) + p_i(t) \geq 0$. A mechanism is called *truthful* or *strategyproof* if it satisfies both IC and IR properties. To make the mechanism tractable, the output method $\mathcal{O}()$, and the payment method $\mathcal{P}()$ should be computable in polynomial time.

Arguably the most positive result in mechanism design is what is usually called the generalized Vickrey-Clarke-Groves (VCG) mechanism [31, 5, 12]. The VCG mechanism applies to maximization problems where the objective function is simply the sum of all agents' valuations. A mechanism $M = (\mathcal{O}(t), \mathcal{P}(t))$ belongs to the VCG family if (1) the output $\mathcal{O}(t)$ computed based on the type vector t maximizes the objective function $g(o, t) = \sum_i v_i(t_i, o)$, and (2) the payment to agent i is $\mathcal{O}_i(t) = \sum_{j \neq i} v_j(t_j, o(t)) + h_i(t_{-i})$. Here $h_i()$ is an arbitrary function of t_{-i} and $t_{-i} = (t_1, \cdots, t_{i-1}, t_{i+1}, \cdots, t_n)$ denotes the vector of strategies of all other agents except i. A VCG mechanism is always incentive compatible [12]. Under mild assumptions, VCG mechanisms are the *only* incentive compatible implementations to maximize the total valuations [11].

Although the family of VCG mechanisms is powerful, but it has its limitations. To use a VCG mechanism, we have to compute the exact solution that maximizes the total valuation of all agents. This makes the mechanism computationally intractable in many cases. Replacing the optimal algorithm with non-optimal approximation usually leads to untruthful mechanisms if VCG payment method is used [23]. To make the mechanism tractable, the output method $\mathcal{O}()$, and the payment method $\mathcal{P}()$ should be computable in polynomial time.

Definition 1 *A mechanism $M = (\mathcal{O}, \mathcal{P})$ is said to be β-efficient if for any t,*

$$\sum_{i=1}^{n} v_i(t_i, \mathcal{O}(t)) \geq \beta \cdot \sum_{i=1}^{n} v_i(t_i, OPT(t)),$$

where $OPT(t)$ is the output that maximizes the total valuations of all players when their type vector is t.

Obviously for the set cover game, we cannot design an $o(\ln n)$-efficient polynomial-time computable strategyproof mechanism unless $NP \subset DTIME(n^{\log \log n})$ [7].

In summary, we want to design strategyproof set cover protocols with the following properties. 1) *Incentive Compatibility (IC)*: an agent will reveal its true cost to maximize its utility no matter what the other agents do; 2) *Individual Rationality (IR)*: an agent is guaranteed to have a non-negative utility if it reports its cost truthfully; and 3) *Polynomial Time Computability (PC)*: all computations (the computation of the output and the payment) are done in polynomial time.

Cost Sharing: Consider a set U of n players. For a subset $T \subseteq U$ of players, let $\varkappa(T)$ be the *cost* of providing service to T defined by the system. Here $\varkappa(T)$ could be computed using the minimum cost, denoted by $\mathrm{OPT}(T)$, or the cost computed by some algorithm $\mathcal{A}$, denoted by $\mathcal{A}(T)$, or some arbitrary cohesive function. We always assume that the cost function $\varkappa(T)$ is *cohesive*, *i.e.*, for any two disjoint subsets T_1 and T_2, $\varkappa(T_1 \cup T_2) \leq \varkappa(T_1) + \varkappa(T_2)$.

A cost sharing scheme is simply a function $\xi(i, T)$ with $\xi(i, T) = 0$ for $i \notin T$, for every set $T \subseteq U$ of players. An obvious criterion is that the sharing method should be *fair*. While the definition of budget-balance is straightforward, defining fairness is more subtle: many fairness concepts were proposed in the literature, such as *max-min* [21], *min-max* [27], *core* and *bargaining set* [24]. Typically, the following three properties are required by a cost sharing scheme.

1. (**α-budget-balance**) For all players U, $\alpha \cdot \varkappa(U) \leq \sum_{i \in U} \xi(i, U) \leq \varkappa(U)$, for some given parameter $\alpha \leq 1$. Equivalently, if we divide the shares by α, we would require that the total cost shares of all agents are at least the cost of providing the service, but do not exceed $\frac{1}{\alpha}$ of that. If $\alpha = 1$, we call the cost sharing scheme *budget-balanced*.
2. (**fairness under core**) For any subset $T \subseteq U$, $\sum_{i \in T} \xi(i, U) \leq \mathrm{OPT}(T)$. In other words, the cost shares paid by any subset of players should not exceed the minimum cost of providing the service to them alone, hence they have no incentives to secede.

3. (**Cross-monotonicity**) For any two subsets $T_1 \subseteq T_2$ and $i \in T_1$, $\xi(i, T_1) \geq \xi(i, T_2)$. In other words, the cost share of a player i should not go up if more players require the service. This is also called *population monotone*.

When a cost sharing scheme satisfies α-BB and is in the core, we call it to be in the α-*core*. When each player i has a valuation v_i on getting the service, a mechanism should first decide the output of the game (who will get the service), and then decide what is the share of each selected player (what is the payment method). It is well-known that a cross-monotone cost sharing scheme implies a *group-strategyproof* mechanism [22]. Notice that the cross-monotone property is not the necessary condition for group-strategyproof. Naturally, several additional properties are required for a cost sharing scheme when every player has a valuation on getting the service.

1. (**Strategyproofness**) Assume that the valuation by player i on getting the service is v_i. Let $b = (b_1, b_2, \cdots, b_n)$ be the bidding vector of n players. Let $\mathcal{O}(b) = (o_1, o_2, \cdots, o_n)$ denote whether a player is selected to get the service or not and $\mathcal{P}(b)$ be the charge to player i, *i.e.*, the mechanism is $M = (\mathcal{O}(b), \mathcal{P}(b))$. It is strategyproof if every player maximizes its profit $v_i \cdot o_i - p_i$ when it reports its true valuation, *i.e.*, $b_i = v_i$. This is also called incentive compatibility.
2. (**No Positive Transfer**) For every player i, $p_i \geq 0$. We will not pay players to participate in the game.
3. (**Voluntary Participation**) For every player i, its profit is non-negative, *i.e.*, $v_i \cdot o_i - p_i \geq 0$. This is also called individual rationality.
4. (**Consumer Sovereignty**) If the bids of all other players are fixed, for every player i, there exists a threshold τ_i such that player i is guaranteed to get the service when its bid is at least τ_i.

2.2 Prior Arts on Cost Sharing and Algorithm Mechanism Design

Routing has been an important part of the algorithmic mechanism-design from the very beginning. Strategyproof unicast and the efficient computing of the payment were addressed in [23, 9, 13, 32]. Several results were proposed in the literature to deal with multicast in selfish networks. Feigenbaum *et al.* [10], by assuming a *fixed* multicast structure, designed a strategyproof mechanism that selects a subset of receivers (each with a privately known willing payment) and then shares the cost of the multicast tree providing the service among the selected receivers so budget-balance is achieved. Maximizing profit in strategyproof multicast was studied in [17, 3]. Sharing the *cost* of the multicast structure among receivers was studied in [20, 10, 22, 28, 8, 2, 14] so some fairness is accomplished.

Although the traditional set cover problem (without multisets and partial-cover requirement) can be viewed as a special case of multicast, several results were proposed specifically for set cover in selfish environment. Devanur *et al.* [6] studied, for the set cover game and facility location game, how the cost of shared resource is to be distributed among its users in such a way that revealing the true valuation is a dominant strategy for each user. Their cost sharing method is not in the *core* of the game. One of the open questions left in [6] is to design a strategyproof cost sharing method for multicover game in which the bidders might want to get covered multiple times. Pál and Tardos [25] gave a cost sharing method that can recover $\frac{1}{3}$ of the total cost for facility location game, and recently, Immorlica *et al.* [16] showed that this is the best achievable upper bound for any cross-monotonic cost sharing method. Hoefer [15] studied non-cooperative games coming from combinatorial covering and facility location problems. Penna [26] showed that mechanisms satisfying all requirements (voluntary participation, no positive transfer, consumer sovereignty, budget-balance and group-strategyproof) must obey certain algorithmic properties (which basically specify how the serviced users are selected). Albers [1] studied network design games where n self-interested agents have to form a network by purchasing links from a given set of edges. She considered Shapley cost sharing mechanisms that split the cost of an edge in a fair manner among the agents using the edge. It shows that using coordination, the price of anarchy drops from linear to logarithmic bounds. Sun *et al.* [30] studied the mechanism design and payment (or cost) sharing problems for set cover games when each element to be covered is an individual autonomous agent.

3 Cost Sharing Among Unselfish Service Receivers

In this section, we study how to share the cost of the service providers among a given set of service receivers. For this scenario, it is difficult to find realistic examples where a partial cover is desired. Therefore, in the remainder of this section, we only consider the multiset multicover problem, *i.e.*, $k = \sum_{i=1}^{n} r_i$. However, the results presented in this section can easily be generalized to the partial cover case, should such a scenario arise.

3.1 α-Core

Given a subset of elements X, let $\mathrm{OPT}(X)$ denote the cost of an optimum cover $\mathcal{C}_{opt}(X)$ of X. This cost function clearly is *cohesive*: for every partition $T_1, T_2, \cdots, T_t$ of U, $\mathrm{OPT}(U) \leq \sum_{i=1}^{t} \mathrm{OPT}(T_i)$. A *cost allocation* for U is a n-dimensional vector $x = (x_1, x_2, \cdots, x_n)$ that specifies for each element $e_i \in U$ the share $x_i \geq 0$ of the total cost of serving U that e_i shall pay.

Ideally, when the set of elements to be covered is fixed to be U, we want the cost allocation x to be budget-balanced and fair, *i.e.*, being in core. However, the following simple example shows that there is no budget-balanced core for the set-cover game. Let U be $\{1, 2, 3\}$ and the sets be $S_1 = \{1, 2\}$, $S_2 = \{1, 3\}$, and $S_3 = \{2, 3\}$ with costs 2, 2 and 2 respectively. For any allocation $x = \{x_1, x_2, x_3\}$ we have $x_1 + x_2 + x_3 = 4$ (from the budget-balance condition), $x_1 + x_2 \leq 2$, $x_1 + x_3 \leq 2$, and $x_2 + x_3 \leq 2$ (from the core requirement). This is clearly impossible. We then relax the notion of budget-balance to the notion of α-budget-balance for some $\alpha \leq 1$, which means that $\alpha \cdot \mathrm{OPT}(U) \leq \sum_{i=1}^{n} x_i \leq \mathrm{OPT}(U)$. We have the following result on the achievable α-core.

Theorem 1 *For the generalized set cover game, there is* no *cost allocation method that is α-core for $\alpha > \frac{1}{\ln n}$ for every set-cover game.*

PROOF. It suffices to prove this for the traditional set cover game, where $k = n$ and $r_i = 1$ for all e_i. We will build a connection between the cost allocation for a set cover game and the solution to the dual of the LP for set cover problem. Let the non-negative integer $y_j \in \{0,1\}$ denote whether the set S_j is selected in $\mathcal{C}_{opt}(U)$. Then we can represent the set cover problem as the following integer programming (IP): $Z_{IP} = \min \sum_{j=1}^{m} y_j c_j$ subject to (1) $\sum_{j=1}^{m} y_j \cdot k_{j,i} \geq 1$ for every element $e_i \in U$, and (2) $y_j \in \{0,1\}$.

To maximize the α for an α-core allocation is equivalent to maximize $\sum_{i=1}^{n} x_i$ subject to $\sum_{e_i \in T} x_i \leq \mathrm{OPT}(T)$ for every subset $T \subseteq U$. Clearly the maximum value achieved above is *at most* the maximum value achieved by the following linear programming (LP): $Z_{LP}^{*} = \max \sum_{i=1}^{n} x_i$ subject to $\sum_{e_i \in S} x_i \leq \mathrm{OPT}(S)$ for every $S \in \mathcal{S}$. This LP is obviously a dual of the relaxed IP for set cover problem. It is well-known that the integrality gap of set cover problem is $\frac{Z_{IP}}{Z_{LP}^{*}} = \ln n$ [7]. Thus, there is a set cover game such that the total recovered cost of an α-core is at most $Z_{LP}^{*} \leq \frac{Z_{IP}}{\ln n} = \frac{\mathrm{OPT}(U)}{\ln n}$. The theorem then follows. $\square$

Notice that this theorem does not exclude some better α-core cost sharing method (with $\alpha > 1 \ln n$) for some special set-cover games. We then give a cost allocation method that can recover $\frac{1}{\ln d_{max}}$ of the total cost $\mathrm{OPT}(U)$ for a multiset multicover game, where $d_{max} = \max_{1 \leq j \leq m} |S_j|$. Without loss of generality, we assume that $d_{max} \leq \sum_{i=1}^{n} r_i$.

The basic approach of our cost allocation method is as follows. We first run a greedy algorithm (see Algorithm 1) to find a set cover $\mathcal{C}_{grd}$ with an approximation ratio of $\ln d_{max}$ for the multiset multicover game. Starting with $\mathcal{C}_{grd} = \emptyset$, the greedy algorithm adds to $\mathcal{C}_{grd}$ a set $S_{j_{t'}}$ at each round t'. After the s-th round, we define the *remaining required coverage* r'_i of an element e_i to be $r_i - \sum_{t'=1}^{s} k_{j_{t'},i}$. For any $S_j \notin \mathcal{C}_{grd}$, the *effective coverage* $k'_{j,i}$ of e_i by S_j is defined to be $\min\{k_{j,i}, r'_i\}$, the *value* v_j of S_j is defined to be $\sum_{i=1}^{n} k'_{j,i}$, and the *effective average cost* of S_j is defined to be $\frac{c_j}{v_j}$.

Algorithm 1 Greedy algorithm for multiset multicover problem.

$\mathcal{C}_{grd} \leftarrow \emptyset$.
$r'_i \leftarrow r_i$ for each e_i.
while $U \neq \emptyset$ **do**
    pick the set $S_{t'}$ in $\mathcal{S} \setminus \mathcal{C}_{grd}$ with the minimum effective average cost.
    $\mathcal{C}_{grd} \leftarrow \mathcal{C}_{grd} \cup \{S_{t'}\}$.
    for all $e_i \in U$ **do**
        $r'_i \leftarrow \max\{0, r'_i - k_{t',i}\}$.
        if $r'_i = 0$ **then**
            $U \leftarrow U \setminus \{e_i\}$

The greedy algorithm will select a set S_j with the least effective average cost. For any e_i and r such that $r_i - r'_i + 1 \leq r \leq r_i - r'_i + k'_{j,i}$, we let $\mathrm{price}(i,r) = \frac{c_j}{v_j}$. Let $x'_i = \sum_{r=1}^{r_i} \mathrm{price}(i,r)$ and $x_i = \frac{x'_i}{\ln d_{max}}$. We will show that x is indeed a $\frac{1}{\ln d_{max}}$-core.

Theorem 2 *The above-defined cost allocation x is a $\frac{1}{\ln d_{max}}$-core.*

PROOF. We first show that $\sum_{e_i \in X} x_i \leq \mathrm{OPT}(X)$ for every subset $X \subseteq U$. We prove this as follows. For our convenience, we assume that $\mathcal{C}_{opt}(X) = \{S_1, S_2, \cdots, S_l\}$ is an optimum cover of X. Let $\mathcal{C}_{grd} = \{S_{i_1}, S_{i_2}, \cdots, S_{i_t}\}$ be the cover of U computed by the greedy algorithm. We order the sets in $\mathcal{C}_{grd}$ according to the order they are added into $\mathcal{C}_{grd}$.

For each element $e_i \in X$, we give the r_i copies of e_i covered by $\mathcal{C}_{grd}$ distinct labels $e_{i,1}, e_{i,2}, \cdots, e_{i,r_i}$, respectively, according to the order they are covered by $\mathcal{C}_{grd}$. Each element copy $e_{i,r} \in S_{j_{t'}}$ is assigned with a cost $\frac{\mathrm{price}(i,r)}{\ln d_{max}}$, where $\mathrm{price}(i,r)$ is equal to the effective average cost $\frac{c_{j_{t'}}}{v_{j_{t'}}}$ of $S_{j_{t'}}$ at the time $S_{j_{t'}}$ is added into $\mathcal{C}_{grd}$. Therefore, to prove that $\sum_{e_i \in X} x_i \leq \mathrm{OPT}(X)$, we need to show that $\sum_{e_i \in X} \sum_{r=1}^{r_i} \mathrm{price}(i,r) \leq \mathrm{OPT}(X) \cdot \ln d_{max}$.

For any set $S_j \in \mathcal{C}_{opt}(X)$ and any element $e_i \in U$, we give the $k_{j,i}$ copies of e_i in S_j distinct labels $e_{i,1}^{(j)}, e_{i,2}^{(j)}, \cdots, e_{i,k_{j,i}}^{(j)}$ respectively. For any element copy $e_{i,r}^{(j)}$, we define its lexicographic order $O(e_{i,r}^{(j)}) = t'$ if the $(\max\{0, r_i - k_{j,i}\} + r)$-th copy of e_i is covered by $\mathcal{C}_{grd}$ at the t'-th round, or set S_j is added into $\mathcal{C}_{grd}$ at the t'-th round, whichever is earlier. The lexicographic order of $e_{i,r}^{(j)}$ defines the number of round after which $e_{i,r}^{(j)}$ becomes "obsolete" (*i.e.*, no longer useful). For example, if $r_i = 3$ and $k_{j,i} = 2$, the first copy of e_i in S_j becomes obsolete after $\mathcal{C}_{grd}$ has covered e_i twice, because now only one element copy of e_i is needed to satisfy the coverage requirement of e_i.

For each $1 \leq j \leq l$, we place all the element copies of S_j into an array L_j according to the above-defined lexicographic order. If $O(e_{i,r}^{(j)}) = O(e_{i',r'}^{(j)})$, $e_{i,r}^{(j)}$ is placed in front of $e_{i',r'}^{(j)}$ if and only if either $i < i'$ or $i = i'$ and $r < r'$. We denote the element copy at the q-th position of L_i by $L_j[q]$, for $q = 0, 1, \cdots, |S_j| - 1$. Each element copy $L_j[q]$ in the array is associated with a cost $c_j/(|S_j| - q)$. The total associated cost of all element copies in S_j is therefore $(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{|S_j|}) \cdot c_j$, which is bounded by $\ln |S_j| \cdot c_j$. And the total associated cost of all array is therefore bounded by $\mathrm{OPT}(X) \cdot \ln d_{max}$.

The proof technique is to uniquely "charge" each $e_{i,r}$ for $e_i \in X$ by an element copy $e_{i,r'}^{(j)}$ in some array L_j such that the associated cost of $e_{i,r'}^{(j)}$ is no less than $\mathrm{price}(i,r)$. To do that, we maintain a pointer for each array L_j to identify the next element copy that will become "obsolete" (that is, it is no longer useful). Initially, the pointer is pointing to the first element copy in L_j. After the t'-th round, the pointer will move to the first element copy in L_j whose lexicographic order is greater than t'.

At the t'-th round, we examine all useful element copies of X in $S_{j_{t'}}$ (in the increasing order of the element indices) as if they are added into $\mathcal{C}_{grd}$ one by one. Suppose $e_{i,r}$ is the element copy currently being examined. Note that here $r \leq r_j$ as we have already excluded the useless element copies from $\mathcal{C}_{grd}$.

Right before $e_{i,r}$ is added into $\mathcal{C}_{grd}$, the pointer of each array containing at least one copy of e_i must be pointing to a copy of e_i. Once $e_{i,r}$ is added into $\mathcal{C}_{grd}$, we have the following two different cases regarding the movements of these pointers:

Case 1: no pointer is moved. This happens if and only if $r_i' \geq \max_{1 \leq j \leq l} k_{j,i}$. We will charge $e_{i,r}$ to a element copy in some L_j in a later round (see Case 2).

Case 2: p pointers are moved. Let $e_{i,r'}, e_{i,r'+1}, \cdots, e_{i,r}$ be the previously uncharged element copies of e_i. We assign each of $e_{i,r'}, e_{i,r'+1}, \cdots, e_{i,\min\{r,r'+p\}}$ to a pointer move in this round (and leave the remaining uncharged element copies, if $r' + p < r$, to the future rounds). For any $e_{i,s}$, $r' \leq s \leq \min\{r, r'+p\}$, let $L_j[q]$ be the corresponding pointer move (*i.e.*, $L_j[q]$ is the element copy to which the pointer was pointing to before moving). Assume that $e_{i,s}$ was covered when $S_{i_{t''}}$ was added into $\mathcal{C}_{grd}$. Since set S_j has not been selected into $\mathcal{C}_{grd}$ before this round, according to the selection criterion of the greedy algorithm we know that the effective average cost of S_j is no less than that of $S_{i_{t''}}$ when $S_{i_{t''}}$ was added into $\mathcal{C}_{grd}$. The effective average cost of S_j is no less than $c_j/(|S_j| - q)$, as at the start of the t'-th round there are at least $|S_j| - q$ useful element copies in S_j. The effective average cost of $S_{i_{t''}}$ is exactly $\mathrm{price}(i, r)$. Therefore, we have $c_j/(|S_j| - q) \geq \mathrm{price}(i, s)$.

When the r_i-th copy of e_i is covered by $\mathcal{C}_{grd}$, all element copies of e_i in arrays $L_1, L_2, \cdots, L_l$ have become obsolete. Since $\sum_{j=1}^{l} e_{j,i} \geq r_i$, all useful element copies of e_i should have been charged. Therefore, the total associated cost of all useful element copies of X in $\mathcal{C}_{grd}$ should be no more than the total associated cost of all element copies in the arrays $L_1, L_2, \cdots, L_l$, implying that $\sum_{e_i \in X} \sum_{r=1}^{r_i} \mathrm{price}(i, r) \leq C(\mathcal{C}_{opt}(X)) \cdot \ln d_{max}$.

It remains to show that x is $\frac{1}{\ln d_{max}}$-budget-balanced. Let $\mathrm{GRD}(U)$ be the total cost of $\mathcal{C}_{grd}$ computed by the greedy algorithm. Obviously, $\frac{\mathrm{OPT}(U)}{\ln d_{max}} \leq \frac{\mathrm{GRD}(U)}{\ln d_{max}} = \frac{\sum_{i=1}^{n} x'_i}{\ln d_{max}} = \sum_{i=1}^{n} x_i$. Further, by letting $X = U$, we have $\sum_{i=1}^{n} x_i \leq \mathrm{OPT}(U)$. This finishes the proof. □

Recall that the core we defined in this paper requires that, given a set of players U, the total cost sharing $\sum_{e_i \in T} \xi(i, U)$ of a subset of elements T is at most the *optimum* cost of providing service to elements in T. For a set cover game, clearly it is NP-hard to find the optimum cost of covering T. Naturally, one may define the α-core as follows: a cost sharing method $\xi(i, \cdot)$ is called a *relaxed* α-core if (1) $\alpha \cdot \mathcal{C}_{grd}(U) \leq \sum_{i \in U} \xi(i, U) \leq \mathcal{C}_{grd}(U)$; and (2) $\sum_{i \in T} \xi(i, U) \leq \mathcal{C}_{grd}(T)$ for every subset of elements $T \subseteq U$. Even we relax the definition of the core to this, we can still prove the following theorem.

Theorem 3 *With the cost function computed by the greedy algorithm, there is no cost sharing method that is a relaxed α-core for $\alpha = \Omega(\frac{1}{\ln n})$.*

PROOF. We prove this by presenting an example as shown in Figure 1. The key idea behind this example is that $\mathcal{C}_{grd}(U) = H_n \cdot \mathcal{C}_{opt}(U)$ and for a particular subset $T \subset U$, $\mathcal{C}_{grd}(T) = \mathcal{C}_{opt}(T)$. There are n elements $U = \{e_1, e_2, \cdots, e_n\}$ and $n + 1$ sets: $S_i = \{e_i\}$ with cost $c_i = \frac{1}{i+1}$ for $1 \leq i \leq n - 2$, $S_{n-1} = \{e_{n-1}\}$ with cost $c_{n-1} = \frac{1}{n-1}$, $S_n = \{e_1, e_2, \cdots, e_{n-1}\}$ with cost $c_n = 1 - \epsilon$, and $S_{n+1} = \{e_{n-1}, e_n\}$ with cost $c_{n+1} = \frac{2-3\epsilon}{n-1}$. Here $0 < \epsilon < \frac{1}{n-1}$. The coverage requirement r_i is 1 for each e_i. It is not difficult to show that $\mathcal{C}_{grd}(U) = \{S_{n+1}, S_{n-2}, S_{n-3}, \cdots, S_1\}$ with total cost $\frac{2-3\epsilon}{n-1} + \sum_{i=1}^{n-2} \frac{1}{i+1} = \frac{2-3\epsilon}{n-1} + H_{n-1} - 1$. Consider a subset $T = \{e_1, e_2, \cdots, e_{n-1}\}$ of elements. Clearly, the greedy cover will be $\mathcal{C}_{grd}(T) = \{S_n\}$ with total cost $1 - \epsilon$. Assume that $x = (x_1, x_2, \cdots, x_n)$ be a cost allocation method that is a relaxed α-core. Then this requires that $\sum_{e_i \in T} x_i \leq C(\mathcal{C}_{grd}(T)) = 1 - \epsilon$. For a single element e_n,

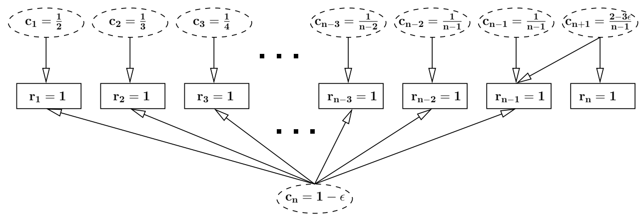

Figure 1: Worst case example for greedy algorithm. Sets are represented by ovals while elements are represented by rectangles. A link (with arrow) between an oval and a rectangle denotes that the set contains one copy of the element.

we have $x_n \le C(\mathcal{C}_{grd}(\{e_n\})) = \frac{2-3\epsilon}{n-1}$. Then for all elements in U, $\sum_{e_i \in U} x_i \le 1 - \epsilon + \frac{2-3\epsilon}{n-1}$. Thus, $\alpha \le \frac{1-\epsilon+\frac{2-3\epsilon}{n-1}}{\frac{2-3\epsilon}{n-1}+H_{n-1}-1} \simeq \frac{1}{H_{n-1}}$. $\square$

3.2 Cross-monotone α-Core

Recall that the definition of α-budget-balance only requires that $\alpha \cdot \varkappa(U) \le \sum_{e_i \in U} x_i \le \varkappa(U)$. A cost sharing scheme ξ is called *cross-monotone α-core* if (1) $\alpha \cdot \varkappa(T) \le \sum_{e_i \in T} \xi(i, T) \le \varkappa(T)$ for every $T \subseteq U$, (2) $\sum_{e_i \in T_1} \xi(i, T_2) \le \mathrm{OPT}(T_1)$ for any subsets T_1 and T_2 with $T_1 \subseteq T_2$, and (3) $\xi(i, T_2) \le \xi(i, T_1)$ for any two subsets T_1 and T_2 with $i \in T_1 \subseteq T_2$. Clearly, if a cost sharing scheme is cross-monotone α-core then every cost allocation method $\xi(\cdot, T)$ induced on a subset T of players is always α-core, but the reverse is not true. Directly from Theorem 1, we know that there is *no* cost sharing scheme for the set cover game that is cross-monotone α-core for $\alpha = \frac{1}{\ln n}$. Recently, Immorlica *et al.* [16] claimed the following result.

Theorem 4 *[16] For set cover game, there is* no *cost sharing scheme that is cross-monotone α-core for $\alpha = \frac{1}{n} + \epsilon$.*

In the remainder of this section, we show that this bound is almost tight for generalized set cover games: there exists a cross-monotone cost sharing scheme $\xi(i, T)$ that can recover $\frac{1}{2n}$ of the total cost. Further, the bound is tight for set cover games without multisets (but still allowing multicover requirements): there exists a cross-monotone cost sharing scheme $\xi(i, T)$ that can recover $\frac{1}{n}$ of the total cost.

Our cost sharing scheme, for each element e_i, finds the set with the minimum cost ratio to cover e_i, then updates the covering requirement and then repeats the above process till the covering requirement is satisfied. We assume that each set S_j is selected to cover the element e_i $Y(i, j)$ times. The cost c_j is proportionally shared by the elements covered by S_j: an element e_i will share a $\frac{Y(i,j)}{\sum_{1 \le i \le n} Y(i,j)}$ fraction. We then describe our cost sharing scheme in Algorithm 2.

Theorem 5 *The cost sharing scheme $\xi(\cdot, \cdot)$ defined by Algorithm 2 is a cross-monotone $\frac{1}{2n}$-core.*

Algorithm 2 Cross-monotone cost sharing for multiset multicover game.

Assume that the set of elements to be covered is $T \subseteq U$.

Initialize $Y(i,j) = 0$ for $1 \le i \le n$ and $1 \le j \le m$. Here $Y(i,j)$ will store how many cover requirements of element e_i are provided by set S_j. $\zeta(i,j) = 0$ for $1 \le i \le n$ and $1 \le j \le m$. Here $\zeta(i,j)$ will store the fraction cost of set S_j shared by the element e_i.

Set $\mathcal{C}_{\mathcal{A}} \leftarrow \emptyset$.

for all element $e_i \in T$ **do**

  Set $r_i' \leftarrow r_i$;

  while $r_i' > 0$ **do**

    Find the set S_t with the minimum ratio $\min_{S_j \in \mathcal{S} - \mathcal{C}_{\mathcal{A}}} \frac{c_j}{\min(k_{j,i}, r_i')}$;

    Set $Y(i,t) \leftarrow \min(k_{j,i}, r_i')$ and $r_i' \leftarrow r_i' - Y(i,t)$.

    Set $\mathcal{C}_{\mathcal{A}} \leftarrow \mathcal{C}_{\mathcal{A}} \cup \{S_t\}$.

for all set S_j **do**

  If $\sum_{1 \le i \le n} Y(i,j) > 0$ (set S_j is used to cover some elements in T), then let $\rho_j = \frac{c_j}{\sum_{1 \le i \le n} Y(i,j)}$;

  for all element $e_i \in T$ **do**

    Set $\zeta(i,j) = Y(i,j) \cdot \rho_j$.

for all element $e_i \in T$ **do**

  Set $\xi'(i,T) = \sum_{1 \le j \le m} \zeta(i,j)$ and $\xi(i,T) = \frac{\sum_{1 \le j \le m} \zeta(i,j)}{2|T|}$.

PROOF. We have to prove that the cost-sharing scheme is $\frac{1}{2n}$-budget-balance for every $T \subseteq U$ and monotone.

Cross-monotone Property: First of all, the cost sharing scheme $\xi(\cdot,\cdot)$ is cross-monotone because adding new element covering requirements (from covering a set elements T_1 to covering a set of elements $T_2 \supset T_1$) will not affect $Y(i,j)$ for element $e_i \in T_1$. It will only change $Y(i,j)$ (for element e_i in $T_2 - T_1$) from 0 to positive. Thus, ρ_j of a set S_j cannot increase when T_2 instead of T_1 is to be covered. Consequently, $\xi'(i,T_1) \ge \xi'(i,T_2)$ for any $i \in T_1 \subset T_2$. This implies that ξ is cross-monotone.

$\frac{1}{2n}$**-budget-balance Property**: It is easy to show that $\sum_{e_i \in T} \xi'(i,T)$ is the total cost of all sets $\mathcal{C}_{\mathcal{A}}$ that are selected to cover some element in T. Given a set T of elements to be covered, let $\mathcal{C}_{opt}(T)$ be the optimum set cover with the minimum cost. In the remainder of the proof, we will only consider an element $e_i \in T$. Let S_{a_1}, S_{a_2}, $\cdots$, S_{a_x} be the sets selected by Algorithm 2 to cover element e_i *in this order*. In other words, every set S_{a_j} ($1 \le j < x$) provides the maximum coverage $Y(i,a_j) = k_{a_j,i}$ to element e_i; while the set S_{a_x} provides a coverage $Y(i,a_x) = r_i - \sum_{j=1}^{x-1} k_{a_j,i}$ to element e_i. Let S_{b_1}, S_{b_2}, $\cdots$, S_{b_y} be the sets in the optimum solution $\mathcal{C}_{opt}(T)$ that satisfies the cover requirement of the element e_i. We will show that the total cost of the sets in $X_a = \{S_{a_1}, S_{a_2}, \cdots, S_{a_x}\}$ is at most twice of the total cost of the sets in $X_b = \{S_{b_1}, S_{b_2}, \cdots, S_{b_y}\}$. In other words, we will first prove that

$$C(X_a) \le 2 \cdot C(X_b) \le 2 \cdot C(\mathcal{C}_{opt}(T)).$$

Let X be the common sets (except the set S_{a_x} if it is a common set) in X_a and X_b, *i.e.*, $X = X_a \cap X_b - \{S_{a_x}\}$. Let $r = \sum_{S_j \in X} k_{j,i}$, and $\tilde{r}_i = r_i - r$. For the moment, we assume that the cover requirement of the element e_i is $\tilde{r}_i$ and the set of sets to be chosen from is $\mathcal{S}' = \mathcal{S} - X$. Clearly $X_a - X$ is still the set of sets selected by Algorithm 2 to satisfy the new cover requirement $\tilde{r}_i$ of element e_i and $X_b - X$ is a still a *valid*[1] solution (not necessarily optimum now) that satisfies the cover requirement.

There are two scenarios here: $S_{a_x} \in X_b$ or $S_{a_x} \notin X_b$.

Case 1: We first analyze the case that $S_{a_x} \in X_b$. Notice that, Algorithm 2 selects a set S_t with the ratio $\min_{S_j \in \mathcal{S}'} \frac{c_j}{\min(k_{j,i}, r_i')}$. Then for every set $S_t \notin X_b - X$ (and S_t is not the last picked set S_{a_x}) selected by Algorithm 2, we have $k_{t,i} = Y(i,t)$. Furthermore, since the sets in $X_b - X$ are not selected, we have

$$\frac{c_t}{k_{t,i}} \leq \min_{S_j \in X_b - X} \frac{c_j}{k_{j,i}} \leq \frac{\sum_{S_j \in X_b - X} c_j}{\sum_{S_j \in X_b - X} k_{j,i}} \leq \frac{C(X_b - X)}{\tilde{r}_i}.$$

The last inequality comes from $\sum_{S_j \in X_b - X} k_{j,i} \geq \tilde{r}_i$. This implies that

$$C(X_a - X - \{S_{a_x}\}) \leq \frac{C(X_b - X)}{\tilde{r}_i} \cdot \sum_{S_t \in X_a - X - \{S_{a_x}\}} k_{t,i} \leq C(X_b - X).$$

Then $C(X_a - X) \leq 2C(X_b - X)$ since $S_{a_x} \in X_b$.

Case 2: We then analyze the case that $S_{a_x} \notin X_b$. For the last picked set S_{a_x}, $Y(i, a_x) \leq k_{a_x,i}$ is the coverage to e_i provided by set S_{a_x}. There are two subcases here.

- **Subcase** 1: There exists a set S_t in $X_b - X$ such that $k_{t,i} \geq Y(i, a_x)$. Then S_t can also provide coverage $Y(i, a_x)$ to the element e_i when we pick the set S_{a_x}. The fact that we pick the set S_{a_x} implies that $\frac{c_{a_x}}{Y(i,a_x)} \leq \frac{c_t}{Y(i,a_x)}$. Thus, $c_{a_x} \leq c_t \leq C(X_b - X)$.

- **Subcase** 2: For every set S_t in $X_b - X$, $k_{t,i} < Y(i, a_x)$. Then every set S_t in $X_b - X$ can only provide a coverage $k_{t,i}$ to the element e_i when we pick the set S_{a_x}. Algorithm 2 picking the set S_{a_x} implies that, for every set $S_t \in X_b - X$, we have $\frac{c_{a_x}}{Y(i,a_x)} \leq \frac{c_t}{k_{t,i}}$. Thus, $\frac{c_{a_x}}{Y(i,a_x)} \leq \frac{\sum_{S_t \in X_b - X} c_t}{\sum_{S_t \in X_b - X} k_{t,i}} \leq \frac{C(X_b - X)}{\tilde{r}_i}$. Remember that, we already proved in **Case** (1) that $C(X_a - X - \{S_{a_x}\}) \leq \frac{C(X_b - X)}{\tilde{r}_i} \cdot \sum_{S_t \in X_a - X - \{S_{a_x}\}} k_{t,i}$. Then we have

$$C(X_a - X) \leq \frac{C(X_b - X)}{\tilde{r}_i} \cdot \sum_{S_j \in X_a - X} Y(i,j) = C(X_b - X).$$

Summarizing the above proofs, we have $C(X_a - X) \leq 2C(X_b - X)$, which implies that $C(X_a) \leq 2C(X_b)$.

[1]The statement is *not* true if we include the possible common set S_{a_x} in X since we may only use part of $k_{a_x,i}$ copies of element e_i in the set S_{a_x} to provide coverage to e_i by Algorithm 2 while the optimum solution may use all these $k_{a_x,i}$ appearances to cover e_i.

Then for all elements in T, the total cost of the sets $\mathcal{C}_{\mathcal{A}}$ selected by Algorithm 2 is at most $2|T| \cdot C(\mathcal{C}_{opt}(T))$. In other words, we have $C(\mathcal{C}_{opt}(T)) \leq \sum_{e_i \in T} \xi'(i,T) = C(\mathcal{C}_{\mathcal{A}}) \leq 2|T| \cdot C(\mathcal{C}_{opt}(T))$. Thus, $\frac{C(\mathcal{C}_{opt}(T))}{2|T|} \leq \sum_{e_i \in T} \xi(i,T) = \frac{C(\mathcal{C}_{\mathcal{A}})}{2|T|} \leq C(\mathcal{C}_{opt}(T))$. This finishes the proof. □

We show by an example that the bound $\frac{1}{2n}$ is tight for Algorithm 2. Assume that there are $n \cdot r + 1$ sets $S_{1+(i-1)\cdot r} = \{e_i\}, S_{2+(i-1)\cdot r} = \{e_i\}, \cdots, S_{r-1+(i-1)\cdot r} = \{e_i\}$, and $S_{r+(i-1)\cdot r} = \{e_i, \cdots, e_i\}$ (with r copies of e_i), for $1 \leq i \leq n$, and a set $S_0 = \{e_1, \cdots, e_1, \cdots e_n, \cdots, e_n\}$ (S_0 has r copies of each element e_i) with the following costs (1) the cost of each set S_j is 1 for $1 \leq j \leq n \cdot r$ and $j \neq 0 \mod r$, (2) the cost of each set S_j is $r(1+\epsilon)$ for $1 \leq j \leq n \cdot r$ and $j = 0 \mod r$, and (3) the cost of S_0 is $r(1+2\epsilon)$. Assume that the cover requirement of each element e_i is $r_i = r$. It is not difficult to show that Algorithm 2 will pick all these sets except S_0 to cover r copies of e_i and the total cost of picked sets are $n(r-1) + n \cdot r(1+\epsilon)$. The optimum solution is to use the set S_0 only with cost $r(1+2\epsilon)$. The ratio $\frac{n(r-1+r(1+\epsilon))}{r(1+2\epsilon)}$ could be arbitrarily close to $2n$ by selecting sufficiently large r and sufficiently small ϵ.

There is still a gap between the above result and the upper bound [16]. However, by dropping the multiset assumption, it is not difficult to prove the following theorem.

Theorem 6 *The cost sharing scheme $\xi(\cdot,\cdot)$ defined by Algorithm 2 is cross-monotone $\frac{1}{n}$-core for set cover game when every set S_j is a simple set.*

4 Cost Sharing Among Selfish Service Receivers

In Section 3 we assumed that all elements (service receivers) are unselfish and all their coverage requirements are to be satisfied. In this section, we consider the problem of selecting service providers under the constraint of a collection of bids $B = B_1 \cup B_2 \cup \cdots \cup B_n$. Each B_i contains a series of bids $b_{i,1}, b_{i,2}, \cdots, b_{i,r_i}$, where $b_{i,r}$ denotes the declared price that element e_i is willing to pay for the r-th coverage (*i.e.*, the valuation of the r-th coverage). In this scenario, we may also consider partial cover, as the total number of units of service available may be limited by a constant k.

We assume that $b_{i,1} \geq b_{i,2} \geq \cdots \geq b_{i,r_i}$. This is often true in realistic situations: the marginal valuations are usually decreasing. A bid $b_{i,r}$ will be served (and the subsequent bid $b_{i,r+1}$ will be considered) only if $b_{i,r} \geq \mathrm{price}(i,r)$, where $\mathrm{price}(i,r)$ is the cost to be paid by e_i for its r-th coverage. Further, to guarantee that the mechanism is both strategyproof and budget-balanced, we assume that each set is a simple set.

We use a greedy algorithm (see Algorithm 3) similar to the one for the traditional set cover game [6]. Informally speaking, we start with $y = 0$, where y is the cost to be shared by each bid served. We raise y until there exists a set S_j whose cost can be sufficiently covered by the element copies in S_j, if each element copy needs to pay y. To adapt to the multicover scenario, we make the following changes:

⋆ For any set $S_j \notin \mathcal{C}_{grd}$ and any e_i, we define the *collection of alive bids* $B_i^{(j)}$ of e_i with respect to S_j to be $\{b_{i,r_i-r'_i+1}\}$ if $k'_{j,i} > 0$ (*i.e.*, $k'_{j,i} = 1$ since S_j is a simple set) and $b_{i,r_i-r'_i+1} \geq y$, or $\emptyset$ if otherwise. That is, if y is the cost to be paid for each bid served, $B_i^{(j)}$ contains the bid of e_i covered by S_j that can afford the cost (if any).

⋆ Define the value v_j of S_j as $\sum_{i=1}^{n} |B_i^{(j)}|$, and its effective average cost as $\frac{c_j}{v_j}$.

Algorithm 3 Cost sharing for multicover game with selfish receivers.

1: $\mathcal{C}_{grd}(B) \leftarrow \emptyset$; $A \leftarrow \emptyset$; $y \leftarrow 0$; $k' \leftarrow k$; $B' = \emptyset$;
2: **while** $A \neq U$ and $k' > 0$ **do**
3:     Raise y until one of the two events happens:
4:     **if** $B_i^{(j)} = \emptyset$ for all S_j **then** $U \leftarrow U \setminus \{e_i\}$;
5:     **if** $c_j \leq v_j \cdot y$ for some set S_j **then**
6:         $\mathcal{C}_{grd}(B) \leftarrow \mathcal{C}_{grd}(B) \bigcup \{S_j\}$; $k' \leftarrow k' - v_j$;
7:         **for all** element e_i with $B_i^{(j)} \neq \emptyset$ **do**
8:             $\mathrm{price}(i, r_i - r_i' + 1) \leftarrow \frac{c_j}{v_j}$; $B' \leftarrow B' \cup \{b_{i, r_i - r_i' + 1}\}$;
9:             $r_i' \leftarrow r_i' - 1$;
10:             **if** $r_i' = 0$ **then** $A \leftarrow A \bigcup \{e_i\}$;
11:         update all $B_i^{(j')}$ for all $S_{j'} \notin \mathcal{C}_{grd}$ and $e_i \in S_j \bigcap S_{j'}$;

When the algorithm terminates, B' contains all bids (of all elements) that are served. We first prove the following property about this mechanism:

Lemma 7 *For each e_i, $\mathrm{price}(i, r)$ is non-decreasing with respect to r.*

PROOF. It suffices to show that right after a set $S_{t'}$ is added into $\mathcal{C}_{grd}$ and all relevant r_i''s and $B_i^{(j')}$'s are updated (as stated in Line 9 and Line 11 of Algorithm 3), there is no $S_j \notin \mathcal{C}_{grd}$ with cost $c_i < v_i \cdot y$. For each e_i, $b_{i,r}$ is non-increasing with respect to r. Therefore, after a bid of e_i is served by $S_{t'}$, the bid (*i.e.*, $b_{r_i - r_i' + 1}$) assigned to the element copy of e_i in S_j will not increase, or even no longer be alive if either $r_i' = 0$ or $y > b_{r_i - r_i' + 1}$). Therefore, the value v_j can only decrease, implying $c_j \geq v_j \cdot y$. ◻

The following lemma directly follows Lemma 7:

Lemma 8 *For any set $S_{t'} \in \mathcal{C}_{grd}$ and any $e_i \in S_{t'}$, if no bid of e_i is served by $S_{t'}$, then no bid of e_i will be served in the subsequent rounds.*

Theorem 9 *Algorithm 3 defines a strategyproof mechanism. Further, the total cost of the sets selected is no more than $\ln d_{max}$ times that of an optimal solution.*

PROOF. We first prove that the mechanism is $\ln d_{max}$-efficient. More specifically, we need to show that the total cost of the sets in $\mathcal{C}_{grd}(B)$ computed by Algorithm 3 is no more than $\ln d_{max}$ times that of the optimum cover $\mathcal{C}_{opt}(B')$. Again, here B' is the collection of bids that are actually covered by $\mathcal{C}_{grd}(B)$.

In Theorem 2 we showed that $\sum_{e_i \in U} x_i \leq \mathrm{OPT}(U)$ for the cost allocation method defined in Section 3. Recall that, for Theorem 2, in order to make the cost allocation method $\ln \frac{1}{d_{max}}$-core, each element e_i only pays $\ln \frac{\mathrm{price}(i,r)}{\ln d_{max}}$ for its r-th coverage. Therefore, for the cost allocation method defined in this section, we have $\sum_{e_i \in U} x_i \leq \ln d_{max} \cdot \mathrm{OPT}(U)$.

However, we still need to modify the proof of Theorem 2 to take into consideration the introduction of bids. It is easy to see that $\mathcal{C}_{grd}(B') = \mathcal{C}_{grd}(B)$. In other words, if the collection of bids given is B' instead of B, Algorithm 3 will still pick exactly the same set of sets (with the exactly same order). With B' as the set of bids to be served, since eventually every bid of B' is served, this problem is the same as the set cover game without bids. Therefore, we have $C(\mathcal{C}_{grd}(B)) = C(\mathcal{C}_{grd}(B')) \leq \ln d_{max} \cdot C(\mathcal{C}_{opt}(B'))$.

Now we prove that the mechanism is strategyproof. Recall that the profit of e_i is defined to be the total value of all served bids of e_i minus the total cost e_i has to pay. Suppose for the sake of contradiction that element e_i can benefit (*i.e.*, achieve a higher profit) from lying about its truthful bids. Among these "profit-increasing" lies, let $\overline{B}_i = \{\overline{b}_{i,1}, \overline{b}_{i,2}, \cdots, \overline{b}_{i,r_j}\}$ be one of the "most truthful lies" of e_i with the maximum q such that $\overline{b}_{i,r} = b_{i,r}$ for all $r < q$ and $\overline{b}_{i,q} \neq b_{i,q}$.

There are two cases:

Case 1: $\overline{b}_{i,q} < b_{i,q}$. There are two subcases.

- **Subcase** 1.a: When e_i is truthful, bid $b_{i,q}$ is served by $S_j \in \mathcal{C}_{grd}$. By reporting $\overline{b}_{i,q}$ instead of $b_{i,q}$, e_i must have caused this bid not to be served by S_j. Otherwise e_i does not need to lie, a contradiction to the assumption that $\overline{B}_i$ is the most truthful lie. We claim that e_i will not benefit from this. First of all, e_i will not gain any profit in S_j. (In contrast, e_i will gain a nonnegative profit in S_j if it is truthful.) Further, by Lemma 8, bids $b_{i,q}, b_{i,r_3+1}, \cdots, b_{i,r_i}$ will not be served by $\mathcal{C}_{grd}$ in later rounds, and thus e_i cannot make any more profit. Therefore, it is more advantageous for e_i to bid truthfully.
- **Subcase** 1.b: When e_i is truthful, bid $b_{i,q}$ is not served by any $S_{t'} \in \mathcal{C}_{grd}$. Since $\overline{b}_{i,q} < b_{i,q}$, bid $\overline{b}_{i,q}$ cannot be served either, and therefore there is no point for e_i to lie about its q-th bid.

Case 2: $\overline{b}_{i,q} > b_{i,q}$. There are also two subcases.

- **Subcase** 2.a: When e_i is truthful, bid $b_{i,q}$ is not served by any $S_{t'} \in \mathcal{C}_{grd}$. If by reporting $\overline{b}_{i,q}$ instead of $b_{i,q}$ this bid is still not served (and thus all subsequent bids will not be served either, according to Lemma 8), it does not make any difference because the outcome is exactly the same as when e_i is truthful, and hence e_i does not need to lie at all. If e_i is served by a set $S_j \in \mathcal{C}_{grd}$, e_i is profit-losing in this round because $\mathrm{price}(i,q) > b_{i,q}$; further, it cannot make any profit in later rounds, because $\mathrm{price}(i,r)$'s are non-decreasing while $b_{i,r}$'s are non-increasing with respect to r.
- **Subcase** 2.b: When e_i is truthful, bid $b_{i,q}$ is served by a set $S_j \in \mathcal{C}_{grd}$. Since $\overline{b}_{i,q} > b_{i,q}$, bid $\overline{b}_{i,q}$ will be served either, and therefore the only incentive for e_i to report $\overline{b}_{i,q}$ instead of $b_{i,q}$ is that it also needs to lie about the next bid $\overline{b}_{i,q+1}$ such that $\overline{b}_{i,q+1} > b_{i,q}$ (since we enforce that the bids are non-increasing). In this sequence of lies, there must be one bid, say, $b_{i,q'}$, that makes a difference: the q'-th bid is served by a set by reporting $\overline{b}_{i,q'}$ but will not be served by any set by reporting $b_{i,q'}$. However, as already shown in Subcase 2.a, e_i will not benefit from this.

This finishes the proof. □

In [6] multicover game was also considered. However, the algorithm used is different from ours and also they did not assume that the bids by the same element are non-increasing, and their mechanism is not strategyproof.

5 Selfish Service Providers

In the previous sections, we studied how the costs of the service providers are shared among service receivers such that approximate budget-balance and some fairness are achieved. The underline assumption made so far is that the service providers are truthful in revealing their costs of providing the service. In this section, we will address the scenario when service providers are selfish in revealing their costs.

5.1 Strategyproof Mechanism

Remember that in the generalized set cover problem, there is a set U of n elements that need to be covered: each element e_i need to be covered r_i times, and each agent $1 \leq j \leq m$ can cover a subset of elements S_j with a cost c_j. Let $c = (c_1, c_2, \cdots, c_m)$. We want to find a subset of agents D such that $\bigcup_{j \in D} S_j$ has r_i copies of element e_i for every element $e_i \in U$. The social efficiency of the output D is $-\sum_{j \in D} c_j$, which is the objective function to be maximized. Clearly a VCG mechanism [31, 5, 12] can be applied if we can find the subset of $\mathcal{S}$ that satisfies the multicover requirement of elements in U with the minimum cost. Unfortunately this is NP-hard. We showed that the greedy method presented in Algorithm 1 has an approximation ratio of $\ln d_{max}$.

Let $\mathcal{C}_{grd}(\mathcal{S}, c, U, r)$ be the sets selected from $\mathcal{S}$ (with cost specified by a cost vector $c = (c_1, \cdots, c_m)$) by the greedy algorithm to cover elements in U with cover requirement specified by a vector $r = (r_1, \cdots, r_n)$ (see Algorithm 1). Notice that the output set is a function of $\mathcal{S}$, c, U, and r. The type of an agent could be each set c_j, *i.e.*, the elements in S_j are assumed to be a public knowledge. Here, we consider a more general case: the type of an agent is (S_j, c_j). In other words, we assume that every service provider j could lie not only about its cost c_j but also about the elements it could cover. This problem now looks very similar to the combinatorial auction with single minded bidder studied in [18], but with the following differences: in the set cover problem here we want to cover all the elements with at least a given cover requirement and the sets chosen can have some overlap while in combinatorial auction we want to maximize the sum of the cost of all sets and the chosen sets are disjoint.

Assume that we use Algorithm 1 to find a set cover, and want to apply VCG mechanism to compute the payment to the selected agents. The payment to an agent j is 0 if $S_j \notin \mathcal{C}_{grd}$. Otherwise, the payment to a set $S_j \in \mathcal{C}_{grd}$ is

$$\mathcal{P}_j^{VCG} = C(\mathcal{C}_{grd}(\mathcal{S} \setminus \{S_j\}, c|^j \infty, U, r)) - C(\mathcal{C}_{grd}(\mathcal{S}, c, U, r)) + c_j.$$

Here $C(\mathcal{X})$ is the total cost of the sets in ξ for $\mathcal{X} \subseteq \mathcal{S}$. We show that this payment scheme based on VCG is not truthful by the following example. Consider the universal set $U = \{e_1, e_2, \cdots, e_n\}$ and $\mathcal{S} = \{S_1, S_2, \cdots, S_{n+1}\}$, where $S_i = \{e_i\}$ for $1 \leq$

$i \leq n$, and $S_{n+1} = \{e_1, e_2, \cdots, e_n\}$. The cost $c_i = \frac{1}{n-i+1}$, for $1 \leq i \leq n$, and $c_{n+1} = 1 + \epsilon$, where ϵ is a small positive number. It is easy to show that the payment to agent 1 is $\mathcal{P}_1^{VCG} = 1 + \epsilon - H_n + 1/n$, which is less than its cost $1/n$. In other words, the mechanism $M = (\mathcal{C}_{grd}, \mathcal{P}^{VCG})$ is not truthful.

For the moment, we assume that agent j won't be able to lie about its element S_j. We will drop this assumption later. Clearly, the greedy set cover method presented in Algorithm 1 satisfies a monotone property: if a set S_j is selected with a cost c_j, then it is still selected with a cost less than c_j. Monotonicity guarantees that there exists a strategyproof mechanism for generalized set cover games using the greedy method to compute its output. We then show how to compute the payment to each service provider efficiently. We assume that for any set S_j, if we remove S_j from $\mathcal{S}$, $\mathcal{S}$ still satisfies the coverage requirements of all elements in U. Otherwise, we call the set cover problem to be *monopoly*: the set S_j can charge an arbitrarily large cost in the monopoly game. The following presents our strategyproof mechanism for multiset multicover set cover problem.

Algorithm 4 Strategyproof payment $\mathcal{P}_j^{grd}$ to service provider $S_j \in \mathcal{C}_{grd}$.

$\mathcal{C}_{grd} \leftarrow \emptyset$ and $s \leftarrow 1$;
$k' \leftarrow k$, $r'_i = r_i$ for each e_i;
while $k' > 0$ **do**
  pick the set $S_t \neq S_j$ in $\mathcal{S} \setminus \mathcal{C}_{grd}$ with the minimum effective average cost;
  Let v_t and v_j be the values of the sets S_t and S_j at this moment;
  $\kappa(j,s) \leftarrow \frac{v_j}{v_t} c_t$ and $s \leftarrow s+1$;
  $\mathcal{C}_{grd} \leftarrow \mathcal{C}_{grd} \cup \{S_t\}$;
  $k' \leftarrow k' - v_t$;
  for each e_i, $r'_i \leftarrow \max\{0, r'_i - k_{t,i}\}$;
$\mathcal{P}_j^{grd} = \max_{t=1}^{s-1} \kappa(j,t)$ is the payment to selfish service provider S_j.

It is easy to show that $\mathcal{P}_j^{grd}$ computed by Algorithm 4 is the threshold cost for set S_j such that it is selected in the greedy set cover if and only if it reports a cost less than $\mathcal{P}_j^{grd}$. Thus, the mechanism $M = (\mathcal{C}_{grd}, \mathcal{P}^{grd})$ is strategyproof (when the agent j does not lie about the set S_j of elements it can cover) and the payment $\mathcal{P}_j^{grd}$ is the minimum payment to the selfish service provider j among any strategyproof mechanism using the greedy set cover method described in Algorithm 1 as its output.

We now consider the scenario when agent j can also lie about S_j. Assume that agent j cannot lie upward[2], *i.e.*, it can only report a $S'_j \subseteq S_j$. We argue that agent j will not lie about its elements S_j. Notice that the value $\kappa(j,s)$ computed for the s-th round is $\kappa(j,s) = \frac{v_j}{v_t} c_t = \frac{\sum_{1 \leq i \leq n} \min(r'_i, k_{j,i})}{\sum_{1 \leq i \leq n} \min(r'_i, k_{t,i})} c_t$. Obviously v_j cannot increase when agent j reports any set $S'_j \subseteq S_j$. Thus, falsely reporting a smaller set S'_j will not improve the payment of agent j.

[2]This can be achieved by imposing a large enough penalty if an agent could not provide the claimed service when it is selected.

Theorem 10 *Algorithm 1 and 4 together define a $\ln d_{max}$-efficient strategyproof mechanism $M = (\mathcal{C}_{grd}, \mathcal{P}^{grd})$ for multiset multicover set cover game.*

Notice that so far we assumed that each set S_j is an individual agent. In practice, it is possible that a selfish agent controls several different sets in $\mathcal{S}$. Assume that there are g agents $\{1, 2, \cdots, g\}$. Each agent i controls a subset $\mathcal{S}_i \subset \mathcal{S}$ such that $\bigcup_{i=1}^{g} \mathcal{S}_i = \mathcal{S}$, and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ for $i \neq j$. We still use Algorithm 1 to find a greedy set cover. Assume that we want to compute a payment to a set S_j owned by agent a. The payment computing algorithm has to be changed as follows: the line 4 of Algorithm 4 is replaced by "*pick the set S_t in $\mathcal{S} - \mathcal{C}_{grd} - \mathcal{S}_a$ with the minimum effective average cost*".

5.2 Sharing the Payment Fairly

In the previous subsection, we only define what is the payment to a selfish service provider S_j. A remaining question is how the payment should be charged. A natural way is to charge the payments to all service receivers fairly (under some subtle definitions) to encourage cooperation among service receivers. One natural way of defining fair payment sharing is to extend the fair cost sharing method. Consider a strategyproof mechanism $M = (\mathcal{O}, \mathcal{P})$. Let $\mathcal{P}(T)$ be the total payment to the selfish service providers when T is the set of service receivers to be covered. A payment sharing scheme is simply a function $\pi(i, T)$ such that $\pi(i, T) = 0$ for any element $e_i \notin T$. A payment sharing scheme is called *α-budget-balanced* if $\alpha \cdot \mathcal{P}(T) \leq \sum_{e_i \in T} \pi(i, T) \leq \mathcal{P}(T)$. A payment sharing scheme is said to be a *core* if $\sum_{e_i \in S} \pi(i, T) \leq \mathcal{P}(S)$ for any subset $S \subset T$. A payment sharing scheme is said to be a *α-core* if it is α-budget-balanced and it is a core.

Let us first consider the strategyproof payment method $\mathcal{P}^{grd}$. We first prove the following theorem.

Theorem 11 *There is no α-core payment sharing scheme for the payment method $\mathcal{P}^{grd}$ if $\alpha > \frac{1}{\ln n}$.*

PROOF. Consider the example used in the proof of Theorem 3: we duplicate every set used in that example with the same cost. It is easy to show that $\mathcal{P}^{grd}(U) = H_{n-1} - 1 + \frac{2-3\epsilon}{n-1}$. Consider a set $T = \{e_1, \cdots, e_{n-1}\}$. The payment $\mathcal{P}^{grd}(T)$ is $1 - \epsilon$. From the core property, we have $\sum_{e_i \in T} \pi(i, U) \leq \mathcal{P}^{grd}(T)$ and $\pi(i, \{e_n\}) \leq \mathcal{P}^{grd}(\{e_n\}) = \frac{2-3\epsilon}{n-1}$. Thus, $\alpha \leq \frac{\sum_{e_i \in U} \pi(i,U)}{\mathcal{P}^{grd}(U)} \simeq \frac{1}{H_{n-1}}$. This finishes the proof. □

It is easy to show that if we share the payment to a service provider equally among all service receivers covered by this set, the scheme is not in the core of the game. We leave it as an open problem whether we can design an α-core payment sharing scheme for the payment $\mathcal{P}^{grd}$ with $\alpha = O(\frac{1}{\ln n})$.

In the next, we study the cross-monotone payment sharing scheme. A payment sharing scheme is said to be *cross-monotone* if $\pi(i, T) \leq \pi(i, S)$ for any two subsets $S \subset T$ and $i \in S$. A payment sharing scheme is said to be a *cross-monotone α-core* if it is α-budget-balanced, it is a core, and it is cross-monotone.

Similar to Theorem 4, we propose the following conjecture.

Conjecture 1 *For the strategyproof mechanism $M = (\mathcal{C}_{grd}, \mathcal{P}^{grd})$ of a set cover game, there is no payment sharing scheme $\pi(\cdot,\cdot)$ that is cross-monotone α-core for $\alpha = \frac{1}{n} + \epsilon$.*

In the remaining of this section we will present a cross-monotone budget-balanced payment sharing scheme for a strategyproof payment scheme of the set cover game. Our payment sharing scheme is coupled with the following payment scheme. The strategyproof mechanism uses the output called *least cost set*: for each service receiver e_i, we find the service provider S_j with the least cost efficiency $\frac{c_j}{\min(r_i, k_{j,i})}$ to cover the element e_i. New cost efficient sets are found till the cover requirement of e_i is satisfied. The output method of the mechanism is described in Algorithm 5.

Algorithm 5 Least cost set greedy for multiset multicover game.

```
Let C_lcs ← ∅.
for all element e_i ∈ T do
    Let r'_i ← r_i;
    while r'_i > 0 do
        Find the set S_t with the minimum ratio min_{S_j ∈ S − C_lcs} c_j / min(k_{j,i}, r'_i);
        Let r'_i ← r'_i − min(k_{j,i}, r'_i).
        Let C_lcs ← C_lcs ∪ {S_t}.
```

We then show how we define the mechanism $M = (\mathcal{C}_{lcs}, \mathcal{P}^{lcs})$. The payment, denoted by p_j^i, of an element e_i to a selected set S_j is the largest cost that S_j can declare while S_j is still selected to cover e_i. If the set S_j is not selected to cover e_i, then $p_j^i = 0$. The final payment to a set S_j is defined as $\mathcal{P}_j^{lcs} = \max_{e_i \in U} p_j^i$. We call this mechanism as the *least cost set* mechanism. Algorithm 6 describes our payment method using Algorithm 5 to compute the output.

Algorithm 6 Compute the payment $\mathcal{P}_j^{lcs}$ to a set S_j in $\mathcal{C}_{lcs}$.

```
Let C_lcs ← ∅, p_j^i = 0 for 1 ≤ i ≤ n and s = 1;
for all element e_i ∈ T do
    Let r'_i ← r_i;
    while r'_i > 0 do
        Find the set S_t ≠ S_j with the minimum ratio min_{S_x ∈ S − C_lcs − {S_j}} c_x / min(k_{x,i}, r'_i);
        Let κ(j, i, s) = (min(k_{j,i}, r'_i) / min(k_{t,i}, r'_i)) c_t;
        Let r'_i ← r'_i − min(k_{j,i}, r'_i);
        Let C_lcs ← C_lcs ∪ {S_t} and s ← s + 1;
    p_j^i ← max_{1 ≤ x < s} κ(j, i, s);
P_j^lcs ← max_{1 ≤ i ≤ n} p_j^i;
```

Theorem 12 *The mechanism $M = (\mathcal{C}_{lcs}, \mathcal{P}^{lcs})$ is $\frac{1}{2n}$-efficient and strategyproof.*

PROOF. The proof that the mechanism is $\frac{1}{2n}$-efficient can be directly derived from the proof of Theorem 5. To show that it is strategyproof, we first show that the payment to a set S_j is indeed the largest possible cost it could declare while it is still selected by Algorithm 5. This can be easily verified from the description of our method. Then we show that a set agent j cannot lie its cost c_j to improve its payment. When it is not selected originally, we have $c_j \geq \mathcal{P}_j^{lcs}$. If it lies a cost larger than $\mathcal{P}_j^{lcs}$, its profit does not change; If it lies a cost smaller than $\mathcal{P}_j^{lcs}$, its profit becomes negative: $\mathcal{P}_j^{lcs} - c_j$. Similarly, when it is originally selected, lying cannot improve its profit. This finishes the proof. $\square$

We then study how we charge the service receivers so that a budget-balance is achieved and the charging scheme also is fair under some concepts. Notice that, given a subset of elements T, we can view the total payments $\mathcal{P}(T)$ to all service providers covering T as a "cost" to T. The payment computed by mechanism $M = (\mathcal{C}_{lcs}, \mathcal{P}^{lcs})$ clearly is cohesive. Then naturally, we could use the cost-sharing schemes studied before to share this special cost among elements. However, it is easy to show by example that the previous cost-sharing schemes (studied in Section 3) are not in the core and also not cross-monotone.

Roughly speaking, our payment sharing scheme works as follows. Notice that a final payment to a set S_j is the maximum of payments p_j^i by all elements. Since different elements may have different value of payment to set S_j, the final payment $\mathcal{P}_j^{lcs}$ should be shared *proportionally* to their values, not *equally* among them as cost-sharing. Figure 2 illustrates the payment sharing scheme that follows.

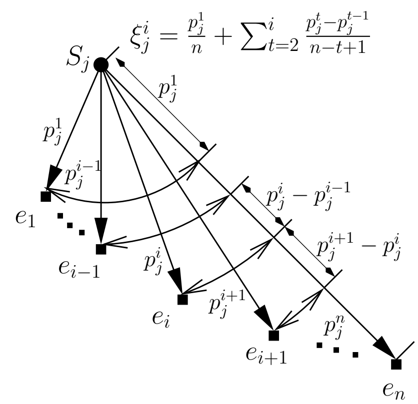

Figure 2: Share the payment to service providers among service receivers fairly.

Without loss of generality, assume that $0 \leq p_j^1 \leq p_j^2 \leq \cdots \leq p_j^n$, *i.e.*, $p_j = p_j^n$. We then divide the payment p_j into n portions: $p_j^1, p_j^2 - p_j^1, \cdots, p_j^i - p_j^{i-1}, \cdots, p_j^n - p_j^{n-1}$. Each portion $p_j^i - p_j^{i-1}$ is then equally shared among the last $n-i+1$ elements, which have the largest $n-i+1$ payments to S_j.

Our payment sharing method applies to a more general cost function. A cost function $\mathcal{P}$ is said to be *maximum-view cost* (MV cost) if it is defined as $\mathcal{P}_j = \max_{e_i \in U} p_j^i$ where p_j^i is the *view* of the cost of set S_j by element e_i. Obviously, the traditional cost

c is a MV cost function by setting $p_j^i = c_j$ for each element e_i. The payment function $\mathcal{P}^{lcs}$ is also a MV cost function.

Algorithm 7 Sharing MV cost $\mathcal{P}$ among receivers.

1: Initialize $\xi(i,U) = 0$ and $\zeta_j(i,U) = 0$. Here $\zeta_j(i,U)$ denotes the payment to set S_j shared by the element e_i when the set of elements is U.
2: **for all** $S_j \in \mathcal{S}$ **do**
3: For all elements e_i, we compute the payment p_j^i. Sort the payments p_j^i, $1 \le i \le n$, in an increasing order. Assume that $p_j^{\sigma(1)}, p_j^{\sigma(2)}, \cdots, p_j^{\sigma(n-1)}, p_j^{\sigma(n)}$ are the sorted list of payments in an incremental order.
4: For elements $e_{\sigma(1)}, \cdots, e_{\sigma(n)}$, let $\zeta_j(\sigma(i),U) \leftarrow \sum_{t=1}^{i} \frac{p_j^{\sigma(t)} - p_j^{\sigma(t-1)}}{n-t+1}$. Here we assume that $p_j^{\sigma(0)} = 0$. Update the payment sharing as follows: $\xi(i,U) = \xi(i,U) + \zeta_j(i,U)$ for each $e_i \in U$.
5: $\xi(i,U)$ is the final payment sharing of service receiver e_i.

A service receiver is called *free-rider* in a payment sharing scheme if its shared total payment is no more than $\frac{1}{n}$ of its total payment it has to pay if it acts alone. Notice that, when a service receiver acts alone, the same mechanism is applied to compute the payment to the service providers.

Theorem 13 *The payment sharing scheme described in Algorithm 7 is budget-balanced, cross-monotone, in the core and does not permit free-rider.*

PROOF. It is easy to see that the payment sharing scheme is budget-balanced: the payment difference $p_j^{\sigma(i)} - p_j^{\sigma(i-1)}$ is equally shared among $n-i+1$ service receivers that have the largest $n-i+1$ payments to the set S_j. It also doe not permit free-rider since, for a service receiver $a = \sigma(i)$, the shared payment of p_j is $\zeta_j(\sigma(i),U) = \sum_{t=1}^{i} \frac{p_j^{\sigma(t)} - p_j^{\sigma(t-1)}}{n-t+1} \ge \frac{p_j^{\sigma(i)}}{n}$. The total shared payment by an element e_i is $\xi(i,U) = \sum_{S_j \in \mathcal{S}} \zeta_j(i,U) \ge \sum_{S_j \in \mathcal{S}} \frac{p_j^i}{n} = \frac{\xi(i,\{i\})}{n}$.

We only need to show that it is cross-monotone and in the core. It is obviously in the core since for any subset of elements X and any set S_j, the total shared payments $\sum_{i \in X} \zeta_j(i,U) \le \max_{i \in X} p_j^i$. Notice that $\max_{i \in X} p_j^i$ is the payment to set S_j if X is the actual set of elements. Then $\sum_{i \in X} \xi(i,U) = \sum_{S_j \in \mathcal{S}} \sum_{i \in X} \zeta_j(i,U)$ is at most the total payment to all sets by X when the subset X of elements plays alone. Clearly the cost sharing method $\xi(\cdot,\cdot)$ is cross-monotone: more receivers added to a given subset of players T will not increase $\zeta_j(i,T)$, thus not increase $\xi(i,T)$. This finishes the proofs. ◻

6 Conclusion

We studied cost sharing and strategyproof mechanisms for various set cover games [19]. We gave an efficient cost allocation method that always recovers $\frac{1}{\ln d_{max}}$ of the total cost, where d_{max} is the maximum size of all sets. We further gave an efficient cost sharing scheme that is $\frac{1}{2n}$-budget-balanced, core and cross-monotone. When the elements to be covered are selfish agents with privately known valuations, we presented a strategyproof charging mechanism. When the sets are selfish agents with privately known costs, we presented two strategyproof payment mechanisms in which each set maximizes its profit when it reports its cost truthfully. We also showed how to *fairly* share the payments to all sets among the elements.

There are a number of open questions left for future research. Are the bounds on the α-budget-balanced cost sharing schemes tight, although we proved that they are asymptotically tight? Consider the strategyproof mechanism $M = (\mathcal{C}_{grd}, \mathcal{P}^{grd})$. Is there a payment sharing method that is $\frac{1}{\ln n}$-core? Is there a payment sharing method that is cross-monotone $\frac{1}{n}$-core? Is this $\frac{1}{n}$ a tight lower bound?

References

[1] ALBERS, S. On the value of coordination in network design. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2008), Society for Industrial and Applied Mathematics, pp. 294–303.

[2] ARCHER, A., FEIGENBAUM, J., KRISHNAMURTHY, A., SAMI, R., AND SHENKER, S. Approximation and collusion in multicast cost sharing. *Games and Economic Behavior 47* (2004), 36–71.

[3] CHAWLA, S., KITCHIN, D., RAJAN, U., RAVI, R., AND SINHA, A. Profit maximizing mechanisms for the extended multicasting game. Tech. Rep. CMU-CS-02-164, Carnegie Mellon University, 2002.

[4] CHVÁTAL, V. A greedy heuristic for the set-covering problem. *Mathematics of Operation Research 4* (1979), 233–235.

[5] CLARKE, E. H. Multipart pricing of public goods. *Public Choice 11* (1971), 17–33.

[6] DEVANUR, N., MIHAIL, M., AND VAZIRANI, V. Strategyproof cost-sharing mechanisms for set cover and facility location games. In *Proceedings of the 4th ACM Conference on Electronic Commerce* (2003), pp. 108–114.

[7] FEIGE, U. A threshold of for approximating set cover. *Journal of the ACM 45* (1998), 634–652.

[8] FEIGENBAUM, J., KRISHNAMURTHY, A., SAMI, R., AND SHENKER, S. Hardness results for multicast cost sharing. *Theoretical Computer Science 304* (2003), 215–236.

[9] FEIGENBAUM, J., PAPADIMITRIOU, C., SAMI, R., AND SHENKER, S. A BGP-based mechanism for lowest-cost routing. In *Proceedings of the 21st Annual ACM Symposium on Principles of Distributed Computing* (2002), pp. 173–182.

[10] FEIGENBAUM, J., PAPADIMITRIOU, C. H., AND SHENKER, S. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences 63* (2001), 21–41.

[11] GREEN, J., AND LAFFONT, J. J. Characterization of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica 45* (1977), 427–438.

[12] GROVES, T. Incentives in teams. *Econometrica 41* (1973), 617–631.

[13] HERSHBERGER, J., AND SURI, S. Vickrey pricing in network routing: Fast payment computation. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science* (2001), pp. 252–259.

[14] HERZOG, S., SHENKER, S., AND ESTRIN, D. Sharing the cost of multicast trees: An axiomatic analysis. *IEEE/ACM Transactions on Networking 5* (1997), 847–860.

[15] HOEFER, M. Non-cooperative facility location and covering games. *Proc 17th Intl Symp Algorithms and Computation (ISAAC)* (2006), 369–378.

[16] IMMORLICA, N., MAHDIAN, M., AND MIRROKNI, V. S. Limitations of cross-monotonic cost-sharing schemes. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms* (2005). To appear.

[17] JAIN, K., AND VAZIRANI, V. V. Applications of approximation algorithms to cooperative games. In *Proceedings of the 3rd ACM Conference on Electronic Commerce* (2001), pp. 364–372.

[18] LEHMANN, D. J., OĆALLAGHAN, L. I., AND SHOHAM, Y. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM 49* (2002), 577–602.

[19] LI, X.-Y., SUN, Z., AND WANG, W. Cost sharing and strategyproof mechanisms for set cover games. In *In Proceedings of the 22nd International Symposium on Theoretical Aspects of Computer Science (STACS 2005), volume 3404 of LNCS* (2005), pp. 218–230.

[20] LIBMAN, L., AND ORDA, A. Atomic resource sharing in noncooperative networks. *Telecommunication Systems 17* (2001), 385–409.

[21] MARBACH, P. Priority service and max-min fairness. *IEEE/ACM Transactions on Networking 11* (2003), 733–746.

[22] MOULIN, H., AND SHENKER, S. Strategyproof sharing of submodular costs:budget balance versus efficiency. *Economic Theory 18* (2001), 511–533.

[23] NISAN, N., AND RONEN, A. Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing* (1999), pp. 129–140.

[24] OSBORNE, M. J., AND RUBINSTEIN, A. *A course in game theory*. The MIT Press, 1994.

[25] PÁL, M., AND TARDOS, E. Group strategyproof mechanisms via primal-dual algorithms. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science* (2003), pp. 584–593.

[26] PENNA, P., AND VENTRE, C. The algorithmic structure of group strategyproof budgetbalanced cost-sharing mechanisms. *Proc. of the Annual Symposium on Theoretical Aspects of Computer Science (STACS 3884*, 337–348.

[27] RADUNOVIĆ, B., AND BOUDEC, J.-Y. L. A unified framework for max-min and minmax fairness with applications. In *Proceedings of 40th Annual Allerton Conference on Communication, Control, and Computing* (2002).

[28] SHENKER, S., CLARK, D., ESTRIN, D., AND HERZOG, S. Pricing in computer networks: Reshaping the research agenda. *ACM Computer Communication Review 26* (1996), 19–43.

[29] SLAVÍK, P. Improved performance of the greedy algorithm for partial cover. *Information Processing Letters 64* (1997), 251–254.

[30] SUN, Z., LI, X.-Y., , WANG, W., AND CHU, X. Mechanism design for set cover games when elements are agents. In *The First International Conference on Algorithmic Applications in Management, (AAIM)* (2005).

[31] VICKREY, W. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance 16* (1961), 8–37.

[32] WANG, W., AND LI, X.-Y. Truthful low-cost unicast in selfish wireless networks. In *Proceedings of the 4th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks* (2004).