Optimal Constraint Collection for Core-Selecting Path Mechanism*

Hao Cheng, Lei Zhang*, Yi Zhang, Jun Wu, ChongJun Wang
State Key Laboratory for Novel Software Technology at Nanjing University
Nanjing, China
njuchengh@163.com,zhangl@nju.edu.cn
mg1733091@smail.nju.edu.cn,{wujun,chjwang}@nju.edu.cn

ABSTRACT

In path auctions, strategic bidders make bids for commodities. Each edge of the graph stands for a commodity and the weight on the edge represents the prime cost. Auctioneer needs to purchase a sequence of edges in order to get a path from one vertex to another at a low cost. Path auctions can be considered as a kind of combinatorial reverse-auctions. Computing prices in core-selecting combinatorial auctions is a computationally hard problem, the same is true in core-selecting path auctions. This problem can be solved by core constraint generation(CCG) algorithm. However, we find that there are many redundant constraints and the constraint collection can be conciser in core-selecting path mechanism. In this paper, 1) we put forward a new approach to get the constraint collection, and reduce the constraint number from exponential $O(2^n)$ to polynomial $O(n^2)$, where n is the network diameter; 2) we prove that the new constraint collection is not only equivalent to the original collection, but also has no redundant constraint in the worst case; 3) we validate our approach on real-world datasets and obtain excellent results. Furthermore, we provide new insights to think over the core-selecting mechanism in combinatorial auctions.

KEYWORDS

path auctions; the shortest path; core; constraint collection

ACM Reference Format:
Hao Cheng, Lei Zhang*, Yi Zhang, Jun Wu, ChongJun Wang. 2018. Optimal Constraint Collection for Core-Selecting Path Mechanism. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

1 INTRODUCTION

Path auctions have been studied extensively [10, 21, 24] since Nisan and Ronen [20] introduced algorithmic mechanism design. In path auctions, auctioneer tries to buy an s-t-path in a directed graph, where the edges of the graph are owned by bidders. The cost of each edge is the private information of its owners. Path auction is an abstract mathematical model of many scenes such as transport routing, power transmission.

The classic mechanism for path auctions is the well-known Vickrey-Clark-Groves(VCG) mechanism, where bidders pay the externalities they impose on all other bidders. VCG mechanism is the unique mechanism that can guarantee effective allocation and incentive compatibility theoretically. However, VCG mechanism has some issues. On one hand, it may result in a low revenue to the auctioneer [2]. On the other hand, VCG mechanism exists false-name bids so that some bidders may register fake accounts to get more profit[24]. These issues have led to considerable interests in core-selecting mechanism [6–8].

Core-selecting mechanism has been well studied in the area of combinatorial auctions [3, 17, 22, 23]. It is false-name-proof and has better revenue performance than VCG mechanism. So it is widely used in auctions such as spectrum auctions [4], procurement auctions [25] and TV advertising auctions [8]. Core-selecting mechanism selects the outcome from the core so that no coalition in the auction can improve upon the outcome. However, it is NP-hard to find an efficient allocation in general combinatorial auction, which is also known as the winner-determination problem[22]. This problem results in that producing an optimal linear objective over the core is also NP-hard [9]. In core-selecting mechanism, it needs to compute the optimal allocation for all the possible coalitions to describe the core, which is complicated. To solve the computation problem, [9] presents an approach of core constraint generation (CCG) algorithm. CCG algorithm reduces the coalitions which requires considering to a moderate number. Nevertheless, complicated computation is still an important reason that hinders its application. Path auctions can be considered as a special case of combinatorial auctions. [24] has designed the core-selecting path mechanism, where computational problem also exists to produce the core.

The remainder of the paper is organized as follows. We begin by discussing related work of path auctions and core-selecting mechanism. Section 2 describes priori knowledge of path auctions, using directed weighted graph for modeling. A new approach to get simplified polynomial constraints is described in section 3 and it also proves that the new constrains is equal to exponential constraints in two aspects of necessity and sufficiency. Section 4 proves strictly that each constraint in the new constraint collection is indispensable, in other words, we get the optimal core constraint collection. In section 5, we design algorithms to verify the correctness above, analyze the efficiency of the two constraint collections and compare them with CCG algorithm. Section 6 presents the results of our experiments. Section 7 concludes with a summary of what we have accomplished and a discussion of future work.

1.1 Related Work

This paper is based on [24], where they designed the core-selecting path mechanism that is false-name-proof and put forward a new

*Lei Zhang is the corresponding author.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

formulation for the core with 2^n core constraints , n is the network diameter. They also proved the maximum core payment can be computed within time of polynomial in n, but they didn't offer a polynomial algorithm.

The problem of designing economic mechanisms for path auctions was first studied in [20], where VCG mechanism is applied to find the shortest path. It is shown that the VCG payments can be computed using n runs of Dijkstra's algorithm in $O(nm + n^2 \log n)$ time. It is later proved that if the underlying graph is undirected, the VCG payments can be computed in only $O(m + n \log n)$ time [13]. Previous work has found that VCG path mechanism can be forced to make arbitrarily high overpayment in the worst case. In fact the result can be generalized to include all truthful path mechanisms [11, 16]. This led to the study of frugal path mechanisms [1]. Previous work has also studied the VCG overpayment in the Internet inter-domain routing graph [12] and large random graphs [15].

In addition to the literature mentioned above, our work is also related to the literature on core-selecting auction [5–7, 19]. As for the problem of computation, [9] formulated the core separation problem, finding the most violated core constraint for any proposed payment vector. Then they designed CCG algorithm with the separation technique, which could achieve a comfortably rapid solution to produce the bidder-Pareto-optimal core pricing. [14] extended the CCG algorithm based on separability in allocative conflicts between participants, and offered fast algorithms for computing core price in large combinatorial auctions.

2 PRELIMINARIES

We represent a social network by a directed weighted graph $G = (V, E)$. The edges represent the commodities for auctions, owned by strategic agents. Each edge has a prime cost $c_e \in R^+$, only known by the agent who owns it. These agents are also the bidders and they will make bids for the edges in path auctions. The auctioneer aims to buy an edge collection to achieve a path from a source vertex s to a target vertex t. The final solution in path auctions is a profile that describes the chosen path and the payments to the chosen edges.

This is a problem of mechanism design. Each bidder makes a bid $b_e > 0$, then the mechanism offers an allocation rule to determine a coalition of winners $E' = \{e_1, e_2, ..., e_n\}$ and a payment vector $P = (p_1, p_2, \ldots, p_n)$ for the winners. So the outcome of path auctions includes E' and P. We use π_e to describe the utility of bidder e, defined as follows.

$$\pi_e = \begin{cases} p_e - c_e & e \in E' \\ 0 & e \notin E' \end{cases} \quad (1)$$

Denote the auctioneer by 0 and

$$\pi_0 = -\sum_{e \in E'} p_e \quad (2)$$

The utility of the system including bidders and auctioneer 0(i.e., social welfare) is denoted as Π,

$$\Pi = \sum_{e \in E'} \pi_e + \pi_0 = -\sum_{e \in E'} c_e \quad (3)$$

Denote the total cost of the shortest path from s to t in the graph G as $d(s, t, G)$, so the maximum social welfare is $-d(s, t, G)$.

In general auctions, assuming bidders are rational and strategic, they will take strategies to increase their utilities such as reporting a fake cost, registering some fake accounts to bid, forming coalitions with other bidders and so on. So the mechanism is expected to be strategy-proof.

Individual rationality means each bidder is willing to participate in a mechanism only if they are guaranteed a non-negative utility. In path auctions, the mechanism should satisfy $\forall e \in E', p_e \geq c_e$. Efficiency means the outcome of meachanism gets the maximum social welfare Π. According to formula (3), it means that the mechanism needs to determine the bidders on the shortest path as the winners. Incentive compatibility means that bidders report the real edge cost is a dominant strategy in path auctions.

Well-known VCG mechanism is a mechanism that satisfies individual rationality,efficiency and incentive compatibility theoretically. But there are numerous issues with VCG[2], which leads to researches of core-selecting mechanism.

2.1 Core-selecting path mechanism

In the case of path auction, core-selecting mechanism is described as follows.

Model path auction as a cooperative game (N, W) and use the core as a solution concept. N represents all the players in this game. It includes bidders and the auctioneer. The auctioneer is denoted by 0. W represents the social welfare. Let L represent subset of N. For each L, the welfare is defined as

$$W(L) = \begin{cases} -d(s, t, L), & 0 \in L \\ 0, & 0 \notin L \end{cases} \quad (4)$$

Definition 2.1 (Core outcome). In path auctions, a core outcome is an allocation and payment profile such that the utility profile $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ satisfies:

$$\sum_{i \in N} \pi_i = W(N) \quad (5)$$

$$\sum_{i \in L} \pi_i \geq W(L), \forall L \in N \quad (6)$$

$W(N) = -d(s, t, N)$, which represents the maximum social welfare. It means we should determine the coalition of the shortest path as the winner coalition. Formula (6) means that the welfare of the subset L under the profile π is not lower than that under the definition of formula (4). And *core* is defined as the total set of core outcomes. Then we can define the core-selecting mechanism.

Definition 2.2 (Core-selecting path mechanism). A path auction mechanism is core-selecting if 1) it selects the shortest path; and 2) the payment vector P is computed so that $P \in core$.

Core-selecting mechanism satisfies individual rationality and efficiency. And it satisfies the core property in the case of bidders reporting their cost truthfully[24]. The core property means no coalition (subset of all players) can form a mutually beneficial renegotiation among themselves. In addition, core-selecting path mechanism relaxes the property of incentive compatibility so that the bidders may not report truthful cost, which leads to some researches[5]. But it is not the focal point in this paper, so we make the assumption that bidders report their cost truthfully ($i.e.\ b_e = c_e$) in the following discussion.

2.2 Core constraints

To get the core of core-selecting path mechanism, the first step is to determine the shortest path in the graph, which is easy to compute. The next step is to generate the constraints in formula (6). However, the number of constraints in (6) is too huge to compute. Fortunately, it has a big space to simplify and [24] has simplified the constraint collection as $(C1)$.

$$(C1): \sum_{e \in x} p_e \leq d(s,t,G-x) - (d(s,t,G) - \sum_{e \in x} c_e) \tag{7}$$

x is the subset of E', E' is the edge set of the shortest path. $d(s,t,G-x)$ represents the total cost of the shortest path in $G-x$ and $d(s,t,G) - \sum_{e \in x} c_e$ is the total cost of the subset $E' - x$.We assume that E' isn't a cut set of graph G, which is the prerequisite of core existence. Then $d(s,t,G-x)$ always exists in (7).

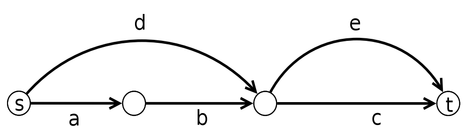

Figure 1: An example of path auctions. There are 5 bidders a, b, c, d, e with cost 1, 1, 1, 5, 3.

The shortest path from s to t is the edge set denoted by $\{a,b,c\}$ and the total cost is 3. According to $(C1)$, we need to find the nonempty subset of $\{a,b,c\}$ and bring into formula (7). Then we get 7 constraints as follows.

$$\begin{cases} a \leq 4, b \leq 4, c \leq 3 \\ a+b \leq 5, b+c \leq 7, a+c \leq 7 \\ a+b+c \leq 8 \end{cases} \tag{8}$$

Let $x = \{a\}$, then the shortest path is 6 in the graph $G - \{a\}$. So we have the constraint $a \leq 6 - (3-1) = 4$. Similarly we can get the other constraints. Moreover, due to individual rationality, we also have the three constraints

$$a \geq 1, b \geq 1, c \geq 1 \tag{9}$$

There are 10 constraints above. An outcome satisfying these constraints is a core outcome, for instance $a = 2$, $b = 1$, $c = 2$. The constraints in $(C2)$ produce a problem of linear programming and core is the feasible domain of this linear programming. However, the computation is also complicated because the number of constraints is $2^n - 1$, n is the length of the shortest path. In order to generate the constraint related to x, we need to compute $d(s,t,G-x)$ by shortest path algorithm. Therefore, $2^n - 1$ constraints in $(C1)$ indicate that the problem of core computation is difficult. To reduce the computational complexity, we put forward a new constraint collection in this paper.

3 A NEW CONSTRAINT COLLECTION

We denote the edge set of the shortest path from s to t as $E(s,t)$[1] and the vertex set of this path as $V(s,t)$[2] including s and t. Then a new constraint collection $(C2)$ is defined as

$$(C2): \sum_{e \in E(a,b)} p_e \leq d(a,b,G-E(a,b)) \tag{10}$$

In $(C2)$, (a,b) is a vertex pair from $V(s,t)$, and b is after a. $d(a,b,G-E(a,b))$ is the total cost of the shortest path from a to b in the graph removing the edges of $E(a,b)$. If there is no path from a to b after removing $E(a,b)$, this pair (a,b) is not included in $(C2)$.

For example, in figure 1, the constraints of $(C2)$ are as follows.

$$\begin{cases} c \leq 3 \\ a+b \leq 5 \\ a+b+c \leq 8 \end{cases} \tag{11}$$

We can see that the constraint number is much smaller than $(C1)$. Given $|E'| = n$, the number of (a,b) is $\frac{n(n+1)}{2}$, which means we can only run $\frac{n(n+1)}{2}$ times shortest path algorithms. So the computational complexity is reduced greatly and could be accepted for practical application.

Theorem 3.1. *The two constraint collections $(C1)$ and $(C2)$ describe the same core.*

This theorem means that $(C2)$ is equivalent to $(C1)$. Next, we will prove theorem 3.1 from two aspects, necessity and sufficiency. Necessity is to prove $(C1) \Rightarrow (C2)$ and sufficiency is to prove $(C2) \Rightarrow (C1)$.

3.1 Necessity: $(C1) \Rightarrow (C2)$

Firstly, we have two lemmas for the shortest path.

Lemma 3.2. *Given s and t, the cost of shortest path is not longer than other paths in the graph.*

Lemma 3.3. *A subpath of a shortest path is itself a shortest path.*

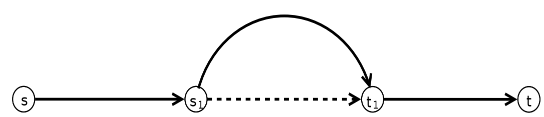

Figure 2: The graph removing $E(s_1,t_1)$.

In figure 2, (s_1,t_1) is an arbitrary vertex pair from $V(s,t)$. The dotted line represents $E(s_1,t_1)$. The curved arrow represents the shortest path from s_1 to t_1 in $G - E(s_1,t_1)$. And the cost of this path is $d(s_1,t_1,G-E(s_1,t_1))$, denoted by d_2. Then we can find a path which is $s \to s_1 \to t_1 \to t$ signed by solid arrows. This path exists after removing $E(s_1,t_1)$ and its cost is $d_2 + d(s,t,G) - \sum_{e \in E(s_1,t_1)} c_e$. According to lemma 3.2, we have

$$d(s,t,G-E(s_1,t_1)) \leq d_2 + d(s,t,G) - \sum_{e \in E(s_1,t_1)} c_e \tag{12}$$

[1] $E(s,t) = \emptyset$ if $s = t$
[2] $V(s,t) = \{s\}$ if $s = t$

In $(C1)$, let $E(s_1, t_1)$ be the subset x, we have

$$\sum_{e \in E(s_1,t_1)} p_e \leq d(s,t,G-E(s_1,t_1)) - (d(s,t,G) - \sum_{e \in E(s_1,t_1)} c_e) \quad (13)$$

Since (s_1, t_1) is arbitrary, combine two formulas (12) and (13) and we can derivate any constraint of formula (10) in $(C2)$. So we prove $(C1) \Rightarrow (C2)$.

3.2 Sufficiency: preparation theorems

We consider arbitrary subset x in $(C1)$. By removing x, the shortest path is divided into several parts. Denote these parts as sets $S(0), S(2), \ldots, S(m)$. Each set represents a subpath of the shortest path except $S(0)$ and $S(m)$[3]. So $s \in S(0), t \in S(m)$. Moreover, the subpath belonging to $S(i)$ is also the shortest in graph $G - x$ according to lemma 3.3.

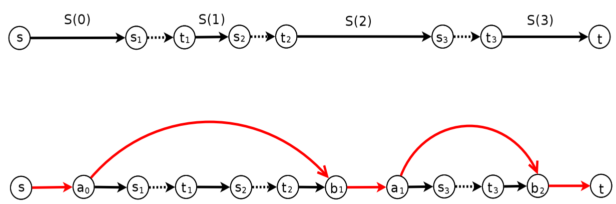

Figure 3: A situation including the shortest path in graph G and $G - x$. Top: The straight path from s to t represents the shortest path in graph G. The dotted arrows represents the removed subset x; Bottom: The path signed with red arrows represents the shortest path in graph $G - x$.

A situation is described as figure 3. In figure 3, we have $x = E(s_1, t_1) \cup E(s_2, t_2) \cup E(s_3, t_3)$. And the shortest path is divided into four parts as $S(1), S(2), S(3), S(4)$. Every set $S(i)$ represents the vertices and edges of a subpath. And the boundary vertices are also in these sets, for example $t_1, s_2 \in S(1)$.

Denote the shortest path from s to t in $G - x$ as E_2, whose cost is denoted as $d(E_2)$. So we have $d(E_2) = d(s, t, G - x)$. In figure 3, E_2 is the s-t-path signed with red arrows. According to the definition, E_2 can't pass any edge in x, but it may pass some edges in $S(i)$. Assuming that a set $S(i), (0 < i < m)$ has common edges with E_2. Here are two theorems for $S(i)$.

Theorem 3.4. *there exists at least one common vertex a in $S(i)$ and E_2, which satisfies*

(1) *in E_2, the edge ending at the vertex a belongs to $S(i)$;*
(2) *in E_2, the edge starting at the vertex a doesn't belongs to $S(i)$.*

Proof. We assume there exists no vertex which meets the requirements. Denote the end vertex of one common edge in $S(i)$ as v_i. In E_2, the edge starting at v_i must belong to $S(i)$, otherwise, v_i is just the vertex we are looking for. Denote the next vertex as v_{i+1} and we have $v_{i+1} \in S(i)$. Similarly the edge starting at v_{i+1} also belongs to $S(i)$, so the next vertex v_{i+2} also belongs to $S(i)$. Keep deriving and we will find that all the vertices after v_i belongs to $S(i)$. Due to $i < m$, This conflicts with the factor that $t \in S(m)$. So the theorem 3.4 is established. □

[3] $S(0)$ may represent $\{s\}$ and $S(m)$ may represent $\{t\}$

We denote the two conditions above as property A. In figure 3, a_1 is a vertex satisfying property A. Similarly we can get theorem 3.5.

Theorem 3.5. *There exists at least one common vertex b in $S(i)$ and E_2, which satisfies*

(1) *in E_2, the edge ending at the vertex b doesn't belong to $S(i)$;*
(2) *in E_2, the edge starting at the vertex b belongs to $S(i)$.*

Proof is similar to the proof of theorem 3.4. Also, denote the two conditions above as property B. And the vertex b_1 satisfies B in the figure 3.

3.3 Suffciency: properties for the vertices

According to the theorem 3.4, 3.5, If $S(i)$ has common edges with E_2, where $0 < i < m$, Then we can find a vertex a satisfying A and a vertex b satisfying B. Moreover, we find the vertex a and b have the properties below. Denote the subpath repersented by $S(i)$ as $t_i \longrightarrow s_{i+1}$, then for the vertex a, we have

Proposition 1. *In E_2, the subpath $a \longrightarrow t$ has no common edges with $E(a, s_{i+1})$, which belongs to $S(i)$.*

Proof. Assuming that property 1 is wrong. The situation can be described as figure 4, where E_2 is represented by the red arrows. And the edge $b' \longrightarrow a'$ is a common edge between $E(a, s_{i+1})$ and the subpath $a \longrightarrow t$ in E_2. We know that the shortest path from b to a' is the straight path belonging to $S(i)$. However, due to that the edge starting at a doesn't belong to $S(i)$, the subpath from b to a' is not the shortest in E_2. This produces a contradiction according to lemma 3.3, so the property is true for vertex a. □

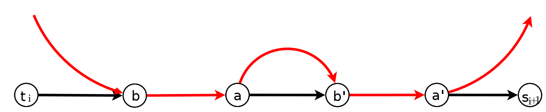

Figure 4: A counter-example for property 1.

In figure 3, as for the vertex a_1, this property means the subpath $a_1 \longrightarrow t$ of E_2 has no common edges with $E(a_1, s_3)$, which belongs to $S(2)$. On the basis of symmetry, we have a similar property for vertex b.

Proposition 2. *In E_2, the subpath $s \longrightarrow b$ has no common edges with $E(t_i, b)$, which belongs to $S(i)$.*

The proof is similar. In figure 3, as for the vertex b_1, this property means the subpath $s \longrightarrow b_1$ of E_2 has no common edges with $E(t_2, b_1)$, which belongs to $S(2)$.

3.4 Sufficiency: path division

Based on the conclusion above, then we consider the traveling process from s to t of path E_2.

At first E_2 starts at $s, s \in S(0)$, then it may pass some edges in $S(0)$ and leave $S(0)$ at a vertex satisfying property A, denote this vertex as a_0. Or E_2 may not pass any edge in $S(0)$ and leave $S(0)$, then denote that $a_0 = s$. After leaving $S(0)$, E_2 will arrive at a set $S(i), (i > 0)$, which is the first set with common edges. If $i \neq m$, firstly E_2 arrives at a vertex denoted as b_1 that satisfies property B, then it passes some edges in $S(i)$ and leave $S(i)$ at a vertex denoted as a_1, which satisfies property A. Then E_2 will arrive at a set $S(j), (j > i)$, which is the first set with common edges after leaving $S(i)$. We can also find the vertex b_2 and a_2 in $S(j)$ if $j \neq m$. Repeat the process until E_2 arrives at $S(m)$. Then E_2 may arrive at a vertex that satisfies property B, denoted as b_k, and E_2 will reach the target vertex t using the subpath in $S(m)$, which is the shortest. Or E_2 may arrive at the target vertex t directly, where we sign that $b_k = t$.

Therefore, we find a division of path E_2 as

$$s \rightarrow a_0 \rightarrow b_1 \rightarrow a_1 ... b_{k-1} \rightarrow a_{k-1} \rightarrow b_k \rightarrow t$$

We denote the set $S(i)$ passed by E_2 as $S'(0), S'(1), ..., S'(k)$ in order, so $S'(0) = S(0), S'(k) = S(m)$.

Take figure 3 as an example, the division of E_2 is $s \rightarrow a_0 \rightarrow b_1 \rightarrow a_1 \rightarrow b_2 \rightarrow t$. And the sets passed by E_2 is $S(0), S(2), S(3)$. We denote them as $S'(0), S'(1), S'(2)$.

According to lemma 3.3, these subpaths are the shortest in graph $G - x$. Then we consider the first part of the subpaths,

$$s \rightarrow a_0, b_1 \rightarrow a_1, ..., b_k \rightarrow t$$

The two endpoints in these subpaths is in the same set $S'(i)$, so the shortest path between them in $G - x$ is the same as G. We denote that $U = \bigcup_{i=0}^{k-1} E(a_i, b_{i+1})$, then the total cost of these paths above is $d(s, t, G) - \sum_{e \in U} c_e$

considering the rest paths:

$$a_0 \rightarrow b_1, a_1 \rightarrow b_2, \ldots, a_{k-1} \rightarrow b_k$$

Denote the total cost of path $a_i \rightarrow b_{i+1}$ as L_i, then we have

$$\sum_{i=0}^{k-1} L_i = d(E_2) - (d(s, t, G) - \sum_{e \in U} c_e) \tag{14}$$

It can be proved that a_0 satisfies the property 1 and b_k satisfies the property 2. In E_2, we know that $a_i \rightarrow b_{i+1}$ is a subpath of the path $s \rightarrow b_{i+1}$ and $a_i \rightarrow t$. Due to the property 1, 2 and that b_{i+1} belongs to the first set $S(j), (j > j')$ which has common edges after $S(j')$, then we can draw a conclusion that subpath $a_i \rightarrow b_{i+1}$ has no common edges with $E(a_i, b_{i+1})$. Therefore, the path $a_i \rightarrow b_{i+1}$ still exists in the graph $G - E(a_i, b_{i+1})$.

Then we consider the vertex pair (a_i, b_{i+1}), According to the constraint in $(C2)$ and lemma 3.2, we have

$$\sum_{e \in E(a_i, b_{i+1})} p_e \leq d(a_i, b_{i+1}, G - x) \leq L_i \tag{15}$$

combining k formulas in (15), we have

$$\sum_{e \in U} p_e \leq \sum_{i=0}^{k-1} L_i \tag{16}$$

Put the formula 14 into 16, we can get the formula as

$$\sum_{e \in U} (p_e - c_e) \leq d(E_2) - d(s, t, G) \tag{17}$$

Then we can see that $x \subseteq U$ because U consists of all the edges which E_2 doesn't pass in the original shortest path. According to individual rationality, we have $p_e - c_e \geq 0$, so

$$\sum_{e \in x} (p_e - c_e) \leq \sum_{e \in U} (p_e - c_e) \tag{18}$$

By combining formula (16) and formula (18), we have

$$\sum_{e \in x} (p_e - c_e) \leq d(E_2) - d(s, t, G) \tag{19}$$

(19) is the same as formula (7) in $(C1)$. It is established for arbitrary subset x ,so we prove the sufficiency. Therefore, theorem 3.1 is true.

4 WORST CASE

Due to the basis of conclusion above, we know that the constraint collection $(C2)$ can produce the core correctly. However, $(C2)$ may have some redundant constraints like the figure 1.

Definition 4.1 (redundant constraint). A constraint is redundant if the feasible domain does not change after removing it from the collection.

To test the redundancy of $(C2)$, we construct a worst case as the figure 5

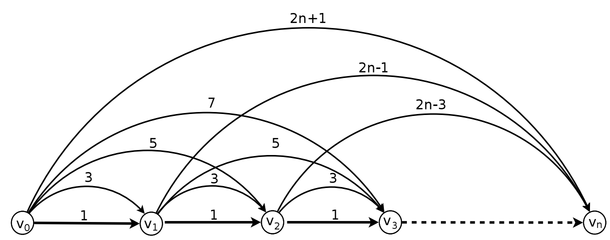

Figure 5: A specific graph, where the number on the edge is the cost.

Figure 5 is a case, where the length of the shortest path is n. Firstly, we consider a sample situation including vertices v_0, v_1, v_2, v_3 and connected directed edges. If auctioneer want to buy a path from v_0 to v_3, the shortest path is the three edges at the cost of 1. Then the bidders owning the three edges will win in core-selecting path auction. Denote the payment vector as $P = (p_1, p_2, p_3)$ in order, then the constraint collection $(C2)$ will be

$$\begin{cases} p_1 \leq 3, \ p_2 \leq 3, \ p_3 \leq 3 \\ p_1 + p_2 \leq 5, \ p_2 + p_3 \leq 5 \\ p_1 + p_2 + p_3 \leq 7 \end{cases} \tag{20}$$

To produce the core, we also need constraints $p_1 \geq 1, p_2 \geq 1, p_3 \geq 1$. These constraints form a core constraint collection according to $(C2)$. And we can find that none of them is redundant.

Theorem 4.2. *For a constraint, if we can find a payment vector out of the core, which is feasible after removing this constraint, then this constraint is not redundant.*

Proof. The appearance of a feasible payment vector out of the core means that the feasible domain change after removing this constraint, so it is not redundant. □

If all the constraints of $(C2)$ in this case is not redundant,then we have the theorem.

THEOREM 4.3. *To produce the core correctly, the number of constraints is at least $\frac{n^2}{2} + \frac{3}{2}n$, where n is the length of the shortest path.*

PROOF. To prove the theorem 4.3, we construct a worst case as figure 5. In this case, the auctioneer aims to buy a path from v_0 to v_n. And we can prove that none of the constraints in $(C2)$ and n constraints related to individual rationality is redundant to produce the core. The number is exactly $\frac{n^2}{2} + \frac{3}{2}n$.

Denote the payment vector as the $(p_1, p_2, ..., p_n)$. Firstly we have n constraints as

$$p_i \geq 1, \ 0 \leq i \leq n \tag{21}$$

Remove the constraint $p_i \geq 1$, we can find an payment vector where $p_i = 0.5$ and other price equals 1, which is out of the core. So these constraints is not redundant.

Secondly, as for following constraints in $(C2)$

$$p_i \leq 3, \ 1 \leq i \leq n \tag{22}$$

They are also not redundant because we find an payment vector where $p_i = 4$ and other payment equals 1, which is not in the core.

Except the constraints above, remaining constraints in $(C2)$ can be described as

$$\sum_{k=i}^{j} p_k \leq 2(j-i)+3, \ 1 \leq i < j \leq n \tag{23}$$

Denote one of the constraints as *RC*, we can find a payment vector out of the core after removing *RC* as

$$p_k = \begin{cases} 3, & k = i, j \\ 2, & i < k < j \\ 1, & other \end{cases} \tag{24}$$

This payment is not in the core as a result of blocking *RC*. And we can prove that it satisfies all the other constraints. Firstly it satisfies the constraints of $p_i \geq 1$. Represent anyone of the remaining constraints in $(C2)$ as

$$\sum_{k=i'}^{j'} p_k \leq 2(j'-i')+3, \quad 0 \leq i' < j' \leq n, \ (i', j') \neq (i, j) \tag{25}$$

We consider these constraints in three cases.

Case 1: $j' - i' > j - i$. According to the payment vector, there are at most 2 payments equal to 3 and $j - i - 1$ payments equal to 2, so

$$\begin{aligned} \sum_{k=i'}^{j'} p_k &\leq 3*2 + 2*(j-i-1) + ((j'-i') - (j-i))*1 \\ &= j' - i' + j - i + 4 \leq 2(j'-i') + 3 \end{aligned} \tag{26}$$

Case 2: $j' - i' = j - i$. Due to $(i', j') \neq (i, j)$, there are at most one payment equal to 3 and $j' - i' - 1$ payments equal to 2, so

$$\sum_{k=i'}^{j'} p_k \leq 3*1 + 2*(j'-i'-1) + 1*1 = 2(j'-i') + 2 \tag{27}$$

Case 3: $j' - i' < j - i$. there are at most one payment equal to 3 and other payments equal to 2, so

$$\sum_{k=i'}^{j'} p_k \leq 3*1 + (j'-i')*2 = 2(j'-i') + 3 \tag{28}$$

Above all, we can see that the payment vector satisfies all the constraints except RC.

This vector is a feasible payment vector out of the core after removing *RC*. Therefore, none of the constraints above is redundant in this worst case. That is, these $\frac{n^2}{2} + \frac{3}{2}n$ constraints are indispensable in this example. So it needs at least $\frac{n^2}{2} + \frac{3}{2}n$ constraints to produce the core correctly, the theorem 4.3 is true. □

In addition, the size of constraint collection based on $(C2)$ and rational individual is exactly $\frac{n^2}{2} + \frac{3}{2}n$, which means it is the optimal core constraint collection for path auctions.

5 PRICING ALGORITHMS

In the experiment, we use the bidder-Pareto-optimal core outcome as our experimental result, which is defined as

Definition 5.1. A core outcome is bidder-Pareto-optimal if there is no other core outcome weakly preferred by every bidder in the winner coalition.

THEOREM 5.2. *An outcome is bidder-Pareto-optimal if it owns the maximum total payment in the core.*

PROOF. The total payment is maximum in the core, so there exists no outcome which could increase one's profit without reducing the profit of others in the winner coalition. Then the outcome with a maximum payment is bidder-Pareto-optimal in the core. □

We use the maximum total payment in the core as our experimental result. And we design the pricing algorithms based on the constraint collection of $(C1)$ and $(C2)$, denoted as C1 and C2 algorithm. Then we implement CCG algorithm for path auctions for the purpose of comparison.

5.1 C1 Algorithm

Review the constraint collection $(C1)$

$$\sum_{e \in x} p_e \leq d(s,t,G-x) - d(s,t,G) + \sum_{e \in x} c_e \tag{29}$$

Let β_x be $d(s,t,G-x) - d(s,t,G) + \sum_{e \in x} c_e$ for each x. So all β_x form a vector $\boldsymbol{\beta}$, then we have

$$\boldsymbol{A}\boldsymbol{p}^T \leq \boldsymbol{\beta}^T \tag{30}$$

(30) represents $(C1)$. $\boldsymbol{A}$ is a $2^n * n$ matrix, n is the edge number of the shortest path. $\boldsymbol{A}_{ij}$ equals 1 only when the i-th set x include the j-th edge e_j, or $\boldsymbol{A}_{ij}$ equals 0. $\boldsymbol{t}$ is the vector of payment profile. Then we can calculate the maximum payment by solving linear programming LP-1

$$\begin{aligned} &\text{LP-1} : \alpha = max\ \boldsymbol{p} \times \boldsymbol{1} \\ &subject\ to : \boldsymbol{A}\boldsymbol{p}^T \leq \boldsymbol{\beta}^T \\ &\boldsymbol{p} \geq \boldsymbol{c} \end{aligned} \tag{31}$$

$\boldsymbol{c}$ is the cost vector and α is the maximum payment. LP-1 has n decision variables and $2^n + n$ constraints. C1 algorithm computes $\boldsymbol{A}$ and $\boldsymbol{\beta}$ and then get the outcome by solving LP-1.

5.2 C2 Algorithm

Review the constraint collection ($C2$)

$$\sum_{e \in E(a,b))} p_e \leq d(a, b, G - E(a, b)) \tag{32}$$

(a, b) is a vertex pair of $E(s, t)$ and b is after a. Similarly, let $\beta'_{(a,b)}$ be $d(a, b, G - E(a, b)$ for each pair (a, b). So all $\beta'_{(a,b)}$ form a vector $\boldsymbol{\beta'}$, then we have

$$\boldsymbol{A'p}^T \leq \boldsymbol{\beta'}^T \tag{33}$$

Similarly, (33) represents ($C2$). $\boldsymbol{A'}$ is a $\frac{n(n+1)}{2} \times n$ matrix, n is the edge number of the shortest path. $\boldsymbol{A'_{ij}}$ equals 1 only when the j-th edge e_j is in the set $E(a, b)$ or $\boldsymbol{A'_{ij}}$ equals 0. Then we can calculate the maximum payment by solving linear programming LP-2

$$\begin{aligned} \text{LP-2} : \alpha = max\ \boldsymbol{p} \times \boldsymbol{1} \\ subject\ to : \boldsymbol{A'p}^T \leq \boldsymbol{\beta'}^T \\ \boldsymbol{p} \geq \boldsymbol{c} \end{aligned} \tag{34}$$

LP-2 has n decision variables and $\frac{n(n+1)}{2} + n$ constraints. The constraint number greatly decreases comparing C1 algorithm. So C1 algorithm computes $\boldsymbol{A'}$ and $\boldsymbol{\beta'}$ and then get the outcome by solving LP-2.

5.3 CCG Algorithm

In CAs, the number of core constraints is exponential in n. To solve the problem of computation, [9] puts forward CCG algorithm in expectation of a moderate number of constraints. CCG algorithm uses the method of constraint generation that considers only the most valuable constraints. Then we design a transmutative CCG algorithm for path auction according to [9].

Definition 5.3 (most blocking path). As for an outcome O including E_m and P, replace the bid in E_m with the payment in P and denote the shortest s-t-path as E'_m, if total cost of edges in E'_m is equal to the total value in P, then O has no blocking paths. Otherwise, E'_m is the most blocking path for the outcome O.

In our CCG algorithm, we denote the shortest path as E_m and the payment vector P as $(p_1, p_2, \ldots, p_n)$. We initialize the payments using VCG payments for each bidders. Moreover, we initialize the constraint set LP for CCG as follows.

$$\begin{cases} p_i \leq p_i^{VCG}, & 1 \leq i \leq n \\ p_i \geq w_i \quad , & 1 \leq i \leq n \end{cases} \tag{35}$$

w_i is the weight of each edge, which represents the quoted price. Then for the outcome with E_m and P, by replacing the bid with P in the graph G we find the most blocking path E'_m. If the total weight of E'_m is not equal to the total payment of P, let $x = Em \cap Em'$, the constraint related to E'_m is

$$\sum_{i \in E_m \setminus x} p_i \leq \sum_{i \in E'_m \setminus x} w_i \tag{36}$$

Then we add the constraint (36) into the constraint set LP, and solve the problem:

$$\begin{aligned} max \quad & \boldsymbol{P} \times \boldsymbol{1} \\ subject\ to: \quad & LP \end{aligned} \tag{37}$$

We use the solution above to update the outcome O, iterate through the above process until the total cost of E'_m is equal to the total payment of P. Then we get the result of O_b, which is a bidder-Pareto-optimal core outcome. The pseudocode of CCG algorithm is given as follows.

Algorithm 1 CCG Algorithm for Path Auction

```
Input: Directed graph G = (V, E, W) (in G each edge has a nonnegative weight w_i, w_i ∈ W), G' = (V, E, W'), W' := W; source vertex s; target vertex t;
Output: Maximum ∑_{i∈E_m} p_i
E_m := edges of the shortest path in the graph G
∀i ∈ E_m, p_i^{VCG} := VCG payment of bidder i
LP := {p_i ≤ p_i^{VCG}, p_i ≥ w_i | i ∈ E_m}
// LP is the constraint set for CCG
∀i ∈ E_m, w_i ∈ W', w_i := p_i
E'_m := edges of the shortest path in the graph G'
while ∑_{i∈E_m} p_i ≠ ∑_{j∈E'_m} w_j do
    x := E_m ∩ E'_m
    LP := LP ∪ {∑_{i∈E_m\x} p_i ≤ ∑_{i∈E'_m\x} w_i}
    p_i := arg max_{i∈E} ∑_{i∈W} p_i, subject to LP
    ∀i ∈ E_m, w_i := p_i
    E'_m := edges of the shortest path in the graph G'
end while
```

6 EXPERIMENT

6.1 Experiment data

We use the network datasets of SNAP[18] datasets to construct the graph in our experiment. The network is described as follows.

- Facebook network. The dataset consists of friend lists from Facebook. The data was collected from survey participants using Facebook app.
- Wikipedia voting network. The network contains voting data for Wikipedia administrators elections . vertices in the network represent wikipedia users and a directed edge from vertex i to vertex j represents that user i voted on user j.
- p2p-Gnutella04 and p2p-Gnutella08. The dataset describes the Gnutella peer-to-peer file sharing network from August 4 2002 and August 8 2002. Vertices represent hosts in the Gnutella network topology and edges represent connections between the Gnutella hosts.
- Twitter network. This dataset consists of friend list from Twitter. The data was crawled from public sources.

The detailed network statistics are given in table 1.

In these networks, the true information of cost is hard to get. So we use reported cost data from a micro-blog advertising platform weiboyi[4], where micro-bloggers are asked to report their cost to

[4] http://www.weiboyi.com/

Table 1: Network statistics.

Networks	Vertices	edges	d_{max}	90-percentile d_{max}
Facebook	4,039	88,234	8	4.7
Wiki-Vote	7,115	103,689	7	3.8
p2p04	10,876	39,994	9	5.4
p2p08	6,301	20,777	9	5.5
Twitter	81,306	1,768,149	7	4.5

make recommendations to friends in their social network. Then we assign the edges randomly with cost dataset.

We use the four networks for our first experiment. In each network, we generate 1000 problem instances where the source vertex s and target vertex t are selected uniformly at random from all vertices. All experiments were ran on 2.3 GHZ Inter Core i5 processor. And we describe the graph with networkx 2.0 and solve the linear programming with SciPy 0.19.1 on a runtime environment of Python 2.7.14. The result is shown in the table 2.

Table 2: Average payment under reported cost distribution.

Networks	Avg. shortest cost	Avg.VCG pay	Avg.C1 max-pay	Avg.CCG max-pay	Avg.C2 max-pay
Facebook	2,601.2	7,898	5,511.50	5511.50	5,511.50
Wiki-Vote	1,014.75	3,005.75	2,630.10	2,630.10	2,630.10
p2p04	5,202	15,572	9,351.58	9,351.58	9,351.58
p2p08	5,606	17,366	10,501.12	10,501.12	10,501.12

From the table 2, it is obvious that the maximum payment computed by CCG, C1 and C2 is always the same in our experiment. The result confirms that the algorithm of CCG and C2 designed is equivalent to C1 algorithm.

6.2 Computational efficiency

The result of average runtime in the first experiment is given in table 3.

Table 3: Average runtime performance (in seconds).

Networks	VCG	C1	CCG	C2
Facebook	0.202	16.050	0.441	0.550
Wiki-Vote	0.297	7.179	0.615	0.602
p2p04	0.468	64.778	0.860	1.420
p2p08	0.205	29.102	0.433	0.665

As is shown in table 3, CCG and C2 algorithm are both much faster than the original C1 algorithm. Moreover, the runtime of CCG and C2 algorithms are approaching runtime of VCG so we conclude that the two algorithms achieve a predominant computation for the core. Furthermore, we experiment with dataset of Wiki-vote network and Twitter network for further comparison. In each network, we randomly selected 50 vertex pairs based on different length of the shortest path. The result is shown in figure 6.

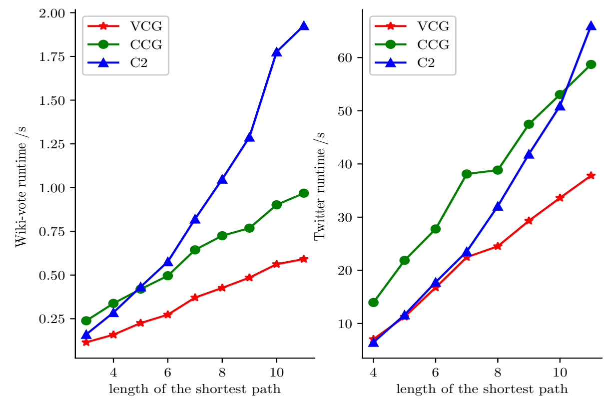

Figure 6: Relationship between runtime and length of the shortest path based on CCG and C2 algorithm and take VCG as baseline. Left: Experiment with Wiki-vote. Right: Experiment with Twitter.

It can be concluded from figure 6 that C2 performs better than CCG in the experiment with dataset of Twitter, but it's opposite in the experiment with dataset of Wiki-vote.Overall, CCG and C2 algorithm are both excellent to compute the core outcome. As for n bidders, CCG algorithm considers all the possible 2^n coalitions and the corresponding exponential set of core constraints. And the solution of CCG is a method constraint genneration. However, C2 algorithm just considers polynomial constraints, which actually reduces problem size for the core-selecting path mechanism. In fact, better performance can be achieved if we combine the two algorithms together.

7 CONCLUSION

In this paper, we reduce the constraint number of core-selecting path mechanism from magnitude of 2^n to n^2, which is proved strictly. Moreover, we prove that the number of constraints is at least $\frac{n^2}{2} + \frac{3}{2}n$ in core-selecting path mechanism. The experimental results illustrate that the new constraint collection is correct meanwhile CCG algorithm and C2 algorithm are both predominant to compute the core outcome.

This approach can be used in combinatorial auctions which have similar structural properties with path auctions. Moreover, our approach offers heuristics for simplified computation in core-selecting combinatorial auctions.

ACKNOWLEDGEMENTS

This paper is supported by the National Key Research and Development Program of China (Grant No. 2016YFB1001102), the National Natural Science Foundation of China (Grant Nos.61502227, 61375069), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University and the Fundamental Research Funds for the Central Universities (Grant Nos.020214380036, 020214380038).

REFERENCES

[1] Aaron Archer and Éva Tardos. 2007. Frugal path mechanisms. *ACM Transactions on Algorithms (TALG)* 3, 1 (2007), 3.

[2] Lawrence M Ausubel and Paul R Milgrom. 2002. Ascending auctions with package bidding. *Advances in Theoretical Economics* 1, 1 (2002).

[3] Edward H Clarke. 1971. Multipart pricing of public goods. *Public choice* 11, 1 (1971), 17–33.

[4] Peter Cramton. 2013. Spectrum auction design. *Review of Industrial Organization* 42, 2 (2013), 161–190.

[5] B Day and Paul Milgrom. 2010. Optimal Incentives in Core-Selecting Auctions. *Handbook of Market Design* (2010).

[6] Robert Day and Paul Milgrom. 2007. Core-selecting auctions. *International Journal of Game Theory, July* (2007).

[7] Robert Day and Paul Milgrom. 2008. Core-selecting package auctions. *international Journal of game Theory* 36, 3-4 (2008), 393–407.

[8] Robert W Day and Peter Cramton. 2012. Quadratic core-selecting payment rules for combinatorial auctions. *Operations Research* 60, 3 (2012), 588–603.

[9] Robert W Day and Subramanian Raghavan. 2007. Fair payments for efficient allocations in public sector combinatorial auctions. *Management Science* 53, 9 (2007), 1389–1406.

[10] Ye Du, Rahul Sami, and Yaoyun Shi. 2010. Path auctions with multiple edge ownership. *Theoretical Computer Science* 411, 1 (2010), 293 – 300. https://doi.org/10.1016/j.tcs.2009.09.032

[11] Edith Elkind, Amit Sahai, and Ken Steiglitz. 2004. Frugality in path auctions. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 701–709.

[12] Joan Feigenbaum, Christos Papadimitriou, Rahul Sami, and Scott Shenker. 2005. A BGP-based mechanism for lowest-cost routing. *Distributed Computing* 18, 1 (2005), 61–72.

[13] John Hershberger and Subhash Suri. 2001. Vickrey prices and shortest paths: What is an edge worth?. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. IEEE, 252–259.

[14] Jean Honorio and Luis Ortiz. 2015. Learning the Structure and Parameters of Large-Population Graphical Games from Behavioral Data. *Journal of Machine Learning Research* 16 (2015), 1157–1210. http://jmlr.org/papers/v16/honorio15a.html

[15] D.R. Karger and E. Nikolova. 2006. On the Expected VCG Overpayment in Large Networks. In *Decision and Control, 2006 45th IEEE Conference on*. 2831–2836. https://doi.org/10.1109/CDC.2006.377149

[16] Anna R. Karlin, David Kempe, and Tami Tamir. 2005. Beyond VCG: Frugality of Truthful Mechanisms. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS '05)*. IEEE Computer Society, Washington, DC, USA, 615–626. https://doi.org/10.1109/SFCS.2005.25

[17] Daniel Lehmann, Liadan Ita Oćallaghan, and Yoav Shoham. 2002. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM (JACM)* 49, 5 (2002), 577–602.

[18] Jure Leskovec, A Krevl, and SNAP Datasets. 2014. Stanford large network dataset collection, 2014. *URL: http://snap. stanford. edu/data/index. html* (2014).

[19] Paul Robert Milgrom. 2004. *Putting auction theory to work*. Cambridge University Press.

[20] Noam Nisan and Amir Ronen. 1999. Algorithmic mechanism design. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM, 129–140.

[21] LC Polymenakos and Dimitri P Bertsekas. 1994. Parallel shortest path auction algorithms. *Parallel Comput.* 20, 9 (1994), 1221–1247.

[22] Michael H Rothkopf, Aleksandar Pekeč, and Ronald M Harstad. 1998. Computationally manageable combinational auctions. *Management science* 44, 8 (1998), 1131–1147.

[23] Makoto Yokoo, Yuko Sakurai, and Shigeo Matsubara. 2004. The effect of false-name bids in combinatorial auctions: New fraud in Internet auctions. *Games and Economic Behavior* 46, 1 (2004), 174–188.

[24] Lei Zhang, Haibin Chen, Jun Wu, Chong-Jun Wang, and Junyuan Xie. 2016. False-Name-Proof Mechanisms for Path Auctions in Social Networks.. In *ECAI*. 1485–1492.

[25] Yuefei Zhu, Baochun Li, Haoming Fu, and Zongpeng Li. 2014. Core-selecting secondary spectrum auctions. *IEEE Journal on Selected Areas in Communications* 32, 11 (2014), 2268–2279.