On the Generalized Vickrey Auction in the Combinatorial Allocation Problem.

M. Keinänen

14 March 2002

Abstract

This is an essay for the course Seminar on Theoretical Computer Science (T-79.194). We review basic concepts of Generalized Vickrey Auction (GVA) as considered in chapter two of Parkes' [Par] thesis. We examine the application of it to so-called Combinatorial Allocation Problem. To exemplify the discussion, we define a GVA based approach to a typical scheduling problem.

1 Introduction

The main theme of this essay is to study classic mechanism design. The exposition is build very much on chapter 2 of Parkes' Ph.D. thesis [Par]. Especially, we will concentrate on Generalized Vickrey Auction (GVA), which will be discussed and explained in section 3. We will also call a little attention to the issue of applications of the GVA. Perhaps the most intriguing application of the conceptual tools offered by GVA is to the so-called Combinatorial Allocation Problem (CAP), which is introduced in section 2. To illustrate the essential concepts involved in GVA we will sketch, in section 4, a GVA approach to a certain scheduling problem, which turns out to be a typical example of the CAP. This formulation is intended to exemplify the issues presented in chapters 2 and 3.

In the following, it is assumed that the reader is familiar with basic game theoretical concepts like strategy, utility etc. (for the general expositions see e.g. [Osb&Rub] or [Fud&Tir]).

2 Combinatorial Allocation Problem

The general combinatorial allocation problem is defined in terms of the following elements:

- $\mathcal{G}$, a set of n discrete goods,
- $\mathcal{I}$, a set of m agents
- agents preferences for goods, so that each agent $i \in \mathcal{I}$ has a valuation function

  $$v_i : 2^{\mathcal{G}} \rightarrow \mathbb{R}_+$$

  defining its value $v_i(S) \geq 0$ for bundles of items $S \subseteq \mathcal{G}$
  (we assume that for all $S \subset S'$, $v_i(\emptyset) = 0$ and $v_i(S) \leq v_i(S')$) .

We assume that agents have quasilinear utility functions, meaning that their valuations can be measured in terms of a common numeraire, $\mathbb{R}_+$, which can be taken to be money. Thus, we can directly compare the utility of different agents, and meaningfully treat the sum as a measure of global value. Agent i gets a utility $v_i(S)$ for holding the bundle of goods $S \subseteq \mathcal{G}$. Then, the valuation function v_i defines the type θ_i of agent i and its preferences over different outcomes.

An outcome simply indicates which agent, if any, gets each good. An outcome defines thus an allocation, a distribution of items to agents. Of course, we assume that no item gets allocated to two different agents in a feasible allocation. Let $S = (S_1, ..., S_I)$ denote a feasible allocation, meaning that the set of goods $S_i \subseteq \mathcal{G}$ is allocated to agent i.

The global value of an allocation, $S = (S_1, ..., S_I)$, is then the sum of the agent values achieved. Furthermore, an allocation is optimal if no other allocation has higher value. Formally, an optimal allocation is defined as

$$\begin{aligned} \max_{(S_1,\ldots,S_I)} \quad & \sum_{i \in \mathcal{I}} v_i(S_i) \\ \text{s.t. } \forall i, j \in \; & \mathcal{I} : S_i \cap S_j = \{\} \end{aligned}$$

With the received notations, we can now state the general form of combinatorial allocation problems (CAPs):

CAP Given set $\mathcal{G}$ of n discrete goods, set $\mathcal{I}$ of m agents and a valuation v_i for all agents in $\mathcal{I}$, the combinatorial allocation problem is to find an optimal allocation $S = (S_1, ..., S_I)$.

In other words, the essence of the CAP is to compute a feasible allocation of given goods to agents, in such way that the allocation maximizes the total value over all agents. In this way, a solution to CAP gives us a desired allocation of goods to agents: no other solution has higher value.

In the remainder of this essay, we present so-called Generalized Vickrey Auctions for this very general CAP. However, the example we present in chapter 4 focuses on a particular scheduling problem, where each agent has one job to complete.

3 Generalized Vickrey Auctions for CAP

We will give in this section a short presentation of a Generalized Vickrey Auction (GVA) for combinatorial allocation problem. Chapter 2 of Parkes' PhD thesis [Par, pp. 23-61] is our basic point in reference, but we will keep in minds that the GVA is originally introduced in [Vic61].

The concept of GVA belongs to the theory of mechanism design, which is a subfield of Microeconomics. One of the main issues studied in mechanism design is the problem to find optimal system-wide solutions to decentraliced optimization problems with self-interested agents having privative information of their preferences. Briefly, a mechanism defines essentially strategies available to agents and a way to select the outcome based on these strategies. A GVA is intended to be a mechanism, which is guaranteed to find efficient solutions for general combinatorial allocation problem described in the previous section. It is often said that GVA is superior to many other auction mechanisms in that it produces optimal solutions to all problems having the form of CAP. Next, we will have a closer look at the structure of GVA mechanism (subsection 3.1), and then turn to discuss some of its more detailed properties (subsection 3.2).

3.1 On the structure of the GVA

The mechanism design problem is, as Parkes puts it [Par, p.29],

> to implement an optimal system-wide solution to a decentralized optimization problem with self-interested agents with private information about their preferences for different outcomes.

Typically mechanism design is studied as a three-step game of incomplete information, where agents' types, i.e. their preferences over goods, are privative information. Roughly speaking, the principal agent (or auctioneer) designs in the first step a mechanism, which is a game in which the agents send messages or bids, and a final allocation depends on the realized messages. In step two, the agents simultaneously accept or reject the mechanism. Finally, in step three, the agents who accept the mechanism play the game specified by the mechanism. Setting up a Generalized Vickrey Auction is essentially nothing more but a way to implement these three steps of this general mechanism design scheme.

The GVA mechanism is a one-shot direct-revelation mechanism. What this means is that, in the GVA game, agents are first asked to report their valuation functions. In other words, each agent bids a value for all possible sets of items. Then, the GVA mechanism computes an allocation based on this information, and finally computes possible payments to all participating agents.

More formally, the course of the GVA game goes in the following way. Recall that $v_i(S)$ is agent i's valuation function for bundles of items $S \subseteq \mathcal{G}$. Each agent $i \in \mathcal{I}$ submits a (possibly untruthful) valuation function, $v_i(S)$, to the auctioneer. After receiving these bids, the GVA returns an allocation. There is a set of choices

$$\mathcal{K} = \{(S_1, ..., S_I) : S_i \cap S_j = \{\} \text{ and } S_i \subseteq \mathcal{G}\}$$

where S_i is an allocation of a bundle of items to agent i. The GVA returns exactly the allocation of items that maximizes reported value over all agents. The allocation obtained is denoted by S^* and defined as follows:

$$\begin{aligned} S^* &= \arg \max_{S=(S_1,...,S_I)} \sum_i v_i(S_i) \\ \text{s.t. } \forall i, j &\in \mathcal{I} : S_i \cap S_j = \{\} \end{aligned}$$

In the final step, the mechanism goes on to compute the payments. The payment relevant here (not the only one, however) is the Groves-Clarke payment and is computed as follows. Let $p_{vick}(i)$ denote the payment for agent $i \in \mathcal{I}$. Let $(V_{-i})^*$ denote the value of second best allocation, that is the value of the best allocation without agent i. Then, given the allocation S^*, the payment for agent i is computed as below:

$$p_{vick}(i) = (V_{-i})^* - \sum_{j \neq i} v_j(S_j^*).$$

In words, to compute the payment for agent $i \in \mathcal{I}$, the auctioneer must first compute the best allocation so that i is not considered. Then, the auctioneer must compute the overall value, $(V_{-i})^*$, of this second best allocation. Finally, the auctioneer computes the value of the best allocation S^* (with i) so that the value of i is not considered, $\sum_{j \neq i} v_j(S_j^*)$. And the payment for agent i is obtained simply by the above subtraction. In other words, as Parkes [Par, 48] well puts it "an agent pays the marginal negative effect that its participation has on the (reported) value of the other agents".

The payments make truth revelation and the dominant strategy (these notions will be explained in detail later on) for an agent, whatever the information reported by other agents in the game. Recall that a dominant strategy maximizes the expected utility for an agent, whatever the strategies of other agents involved in the game. In this way, the GVA mechanism successfully obtains the system-wide goal of computing an efficient allocation.

In the combinatorial allocation problem, the GVA has (at least) the following nice properties:

Theorem 1 (Generalized Vickrey Auction) *[Par, p.48] The GVA is efficient, strategy-proof, individual-rational, and weak budget-balanced for agents with quasi-linear preferences in the combinatorial allocation problem.*

Perhaps a little explanation is needed here. In words, the concepts and properties involved in the above theorem can be spelled out as follows [Par, p.11]:

> **Allocative efficiency.** The mechanism implements the efficient allocation, the allocation that maximizes total value across agents, given rational agent strategies.
>
> **Strategy-proofness.** Truth-revelation is optimal for an agent whatever the strategies of other agents, there can be no successful (unilateral) manipulation of the outcome.
>
> **Individual-rationality.** The expected utility from participation is non-negative (with a rational strategy), whatever the strategies of other agents.
>
> **(Weak) Budget-balance.** Each agent makes a non-negative payment to the auctioneer, so the net revenue collected by the auctioneer is non-negative.

Let us next try to give an explicit, more formal analysis of these GVA properties.

3.2 Some properties of the GVA

In this subsection we merely summarize some key concepts and technical definitions of mechanism design contained in sections 2.2-2.3 of Parkes' thesis; the reader familiar with these questions may well skip the discussion, and turn directly to an example application of GVA in the next section. Unfortunately, in order to keep the essay within manageable length, it is not possible to give full-detailed explanations. This subsection remains therefore somewhat technical.

We begin with some relevant concepts of mechanism design as well as stating the required underlying notations and definitions.

The system goal in mechanism design is typically defined in terms of a social choice function, that selects the optimal outcome given all agent types. This is defined as follows.

Definition 2 (Social choice function) *[Par, p.30] Social choice function $f : \Theta_1 \times ... \times \Theta_I \to \mathcal{O}$ chooses an outcome $f(\theta) \in \mathcal{O}$ given agent types $\theta = (\theta_1, ..., \theta_I)$.*

Intuitively, the type θ_i of an agent i simply determines i's preferences over various possible outcomes. Given agent types, we want to choose the outcome $f(\theta)$, and the mechanism design problem is to define the rules of the game. A mechanism defines the strategies available to players (usually the bidding language) and the method used to select the final outcome based on agent strategies. The notion of mechanism is made precise with the following definition.

Definition 3 (Mechanism) *[Par, p.30] A mechanism $\mathcal{M} = (\sum_1, ..., \sum_I, g(\cdot))$ defines the set of strategies Σ_i available to each agent, and an outcome rule $g : \sum_1 \times ... \times \sum_I \to \mathcal{O}$ such that $g(s)$ is the outcome implemented by the mechanism for strategy profile $s = (s_1, ..., s_I)$.*

Now, because we are mainly interested in finding certain solutions, we use the game-theory to analyze the outcome of a mechanism, which models the problem to be solved. Given a mechanism with an outcome, we may say that it implements a social choice function, if the outcome computed with equilibrium agent strategies is a solution to the social choice function for all possible agent preferences. This intuitive idea is expressed formally by:

Definition 4 (Mechanism implementation) *[Par, p.30] Mechanism* $\mathcal{M} = (\sum_1, ..., \sum_I, g(\cdot))$ *implements social choice function* $f(\theta)$ *if* $g(s_1^*(\theta_1), ..., s_I^*(\theta_I)) = f(\theta)$ *for all* $(\theta_1, ..., \theta_I) \in \Theta_1 \times ... \times \Theta_I$*, where strategy profile* $(s_1^*, ..., s_I^*)$ *is an equilibrium solution to the game induced by* $\mathcal{M}$*.*

These definitions should be clear enough, so let us move on to discuss a solution concept relevant for GVA mechanisms.

Dominant strategy A relevant solution concept here is a dominant strategy equilibrium. In a dominant strategy equilibrium every agent has the same utility-maximizing strategy, for all strategies of other agents. A strategy for agent i, denoted s_i, is a dominant strategy for an agent of type θ_i if it maximizes expected utility u_i, whatever the strategies of other agents. More precisely,

Definition 5 (Dominant-strategy) *[Par, p.28] Strategy* s_i *is a dominant strategy if it (weakly) maximizes the agent's expected utility for all possible strategies of other agents,*

$$u_i(s_i, s_{-i}, \theta_i) \geq u_i(s_i', s_{-i}, \theta_i), \text{ for all } s_i' \neq s_i, s_{-i} \in \Sigma_{-i}$$

Once again, a strategy is a dominant strategy for an agent with some preferences if it simply maximizes expected utility; no matter what the strategies of other agents are.

Quasi-linear preferences A common assumption in auction theory and mechanism design, the one which we make here as well, is that agents have quasi-linear utility functions. To be brief, these types of utility functions can be defined as below.

Definition 6 (Quasi-linear preferences) *[Par, p.32] A quasi-linear utility function for player* i *with type* θ_i *is of the form:*

$$u_i(o, \theta_i) = v_i(x, \theta_i) - p_i$$

where outcome o *defines a choice* $x \in \mathcal{K}$ *from a discrete choice set and a payment* p_i *by the agent.*

Efficiency We can now state some properties of social choice functions implemented by a mechanism.

Definition 7 (Allocative effciency) *[Par, p.33] Social choice function $f(\theta) = (x(\theta), p(\theta))$ is allocatively-efficient if for all types $\theta = (\theta_1, ..., \theta_I)$*

$$\sum_{i=1}^{I} v_i\left(x\left(\theta\right), \theta_i\right) \geq \sum_i v_i\left(x', \theta_i\right), \text{ for all } x' \in \mathcal{K}.$$

In the framework of mechanism design, it is common to state this as allocative efficiency, because the choice sets often define an allocation of items to agents. The key thing to note here is that an efficient allocation maximizes the total value over all agents.

Weak budget-balance A social-choice function is weak budget-balanced, if the following definition holds for it.

Definition 8 (Weak budget-balance) *[Par, p.33] Social choice function $f(\theta) = (x(\theta), p(\theta))$ is weakly budget-balanced if for all preferences $\theta = (\theta_1, ..., \theta_I)$*

$$\sum_{i=1}^{I} p_i\left(\theta\right) \geq 0$$

In this case, there can be a payment made from agents to the mechanism, but no payments from the mechanism to the agents. Each agent makes a non-negative payment to the auctioneer, so the net revenue collected by the auctioneer is non-negative.

Finally, with the above definitions in backpacked, we can easily express the desirable properties of mechanisms. In general, a mechanism has property P if it implements a social choice function with property P.

Efficiency For instance, a mechanism is efficient if it simply selects the choice that maximizes total value:

Definition 9 (Effcient mechanism) *[Par, p.34] Mechanism $\mathcal{M}$ is efficient if it implements an allocatively-efficient social choice function $f(\theta)$.*

In other words, the mechanism implements an efficient allocation, that is an allocation maximizing total value across agents, given rational agent strategies (notice that there may be several such allocations in some cases).

In the similar manner for weak budget-balance: a mechanism is budget balanced if the payment over all agents is nonnegative.

Individual rationality Another property of mechanism relevant for our purposes is individual-rationality. A mechanism is individual-rational if an agent can always achieve as much expected utility from participation as without participation, given prior beliefs about the preferences of other agents.

Definition 10 (Individual-rationality) *[Par, p.35] A mechanism $\mathcal{M}$ is individual-rational if for all preferences θ_i it implements a social choice function $f(\theta)$ with*

$$u_i\left(f\left(\theta_i,\theta_{-i}\right)\right) \geq \bar{u}_i\left(\theta_i\right)$$

where $u_i\left(f\left(\theta_i,\theta_{-i}\right)\right)$ is the expected utility for agent i at the outcome, given distributional information about the preferences θ_{-i} of other agents, and $\bar{u}_i$ (θ_i) is the expected utility for non-participation.

The expected utility from participation is non-negative (with a rational strategy), whatever the strategies of other agents. In other words, a mechanism is individually-rational if no agent can be worse off from participating in the auction than had it declined to participate.

In order to state the final important definition for our discussion, strategy-proofness of mechanisms, we first need to explain two more helpful concepts, namely direct-revelation mechanisms and truth-revealing strategies.

Direct-revelation of mechanisms A so-called direct-revelation mechanism is defined as follows.

Definition 11 (Direct-revelation) *[Par, p. 36] A direct-revelation mechanism $\mathcal{M} = (\Theta_1, ..., \Theta_I, g(\cdot))$ restricts the strategy set $\sum_i = \Theta_i$ for all i, and has outcome rule $g : \Theta_1 \times ... \times \Theta_I \rightarrow \mathcal{O}$ which selects an outcome $g(\widehat{\theta})$ based on reported preferences $\widehat{\theta} = (\widehat{\theta}_1, ..., \widehat{\theta}_I)$.*

Truth-revelation of strategies A truth-revealing strategy for an agent is exactly what one might expect hearing the term: truth-revealing strategy is to report true information about preferences, for all possible preferences. This is defined by:

Definition 12 (Truth-revelation) *[Par, p. 36] A strategy $s_i \in \sum_i$ is truth-revealing if $s_i(\theta_i) = \theta_i$ for all $\theta_i \in \Theta_i$.*

Finally, based on these two notions we can give the last definition.

Strategy-proofness

Definition 13 (Strategy-proof) *[Par, p.37] A direct-revelation mechanism M is strategy-proof if its truth-revelation is a dominant-strategy equilibrium.*

Recall that the in the dominant-strategy equilibrium all agents have the utility-maximizing strategy, for all strategies of other agents.

With the received conceptual machinery and definitions, we can easily see the essential core of Theorem 1. Recall that the only things it states is: GVA is efficient, strategy-proof, individual-rational, and weak budget-balanced for agents with quasi-linear preferences in the combinatorial allocation problem. Theorem 1 shows us the main reason why GVA provides an efficient dominant-strategy solution to the combinatorial allocation problem.

Admittedly, this section has been quite technical and full of definitions. Perhaps a simple example is in order here.

4 Example: Airplane Deicing Scheduling as Generalized Vickrey Auctions

To illustrate the use of GVA in CAP, we design in this section a GVA mechanism to handle a typical CAP, which we call airplane deicing scheduling problem. We will study how the game theoretical concepts of auction theory could be applied to the airplane deicing scheduling problem. It will be pointed out that one merit of the GVA approach is that it will allow us to define the problem formally and find its solutions in extremely elegant way.

4.1 A brief problem description

Consider an airport in northern part of the world, where the airplane flow is constrained in winters by the fact that each plane must be given certain kind of individual deicing treatment before its safe take-off. The overall deicing of planes is controlled by a single scheduler responsible for management of all deicing trucks in the airport. Essentially, the task of the scheduler is to allocate efficiently the available trucks in response to requests from airplane pilots. This must be done in such a way that the overall defeats among airlines are to be minimized, if some planes miss their scheduled take-off times, because all deicing requests cannot be satisfied.

The length of the deicing treatment of a particular airplane depends on (at least)

(i) the size and type of the plane

(ii) the condition of the plane (i.e. temperature of the wings)

(iii) the current weather conditions and meteorological forecast (temperature, snowing etc.)

(iv) the number of trucks working on a particular deicing job

In order to be effective the deicing cannot be started too early before the take-off time; namely, the deicing can be done at most 30 minutes before the take-off.

Planes are typically operated by self-interested competing companies, which prefer their planes to obtain the deicing and to take-off on-schedule even if the other planes must wait on the ground for their turn. While making requests, airline pilots have privative information about possible cargo, repair and fueling schedules as well. In addition, pilots hold privative information about the (possibly tremendous) financial losses, in case their flight will not make the take-off time. The amount of loss for a particular plane depends on e.g. the number of business class travellers on board, whether the plane is reserved for further transfer flights, and whether the plane is so-called slot-flight (i.e. scheduled to arrive at destination exactly at certain time).

Now, the challenges for the deicing manager include:

- for each 30 minute time sequence, decide if all deicing requests from the planes can be satisfied within required times
- select the best feasible subset of requests that can be satisfied so that the overall financial losses among the planes waiting for deicing will be minimal (the reason for this is that the airline companies can charge the airport for possible delays resulting from not obtaining the deicing in time, depending on the possible losses)
- schedule the deicing of the selected planes so that no truck resources are allocated to two planes at the same time.

It is not difficult to see that solving the problem is far from being computationally easy task. Perhaps, this is the reason why the task is currently carried out without computers at Finnish airports, resulting repeatedly in considerable defeats for airline companies.

4.2 Problem analysis

First, it is fairly easy to see that well-known distributed scheduling policies like first-come first-served, shortest-job-first, priority-first do not generally lead to solutions we are looking for. For instance, priority-based scheduling policies raise the problem that if self-interested agents are free to set their own priorities, they may specify the maximum priority for whatever they are interested in, and the desirable results do not necessary follow.

Secondly, we can easily establish that, in fact, the problem shares similar features with many well-known combinatorial problems like e.g.:

- the travelling salesman problem [Kre&Sti, pp.127-128], in which temporal constraints would be added and the goal would be to minimize the travel distance (i.e. the truck resources) but maximize the sum of weights of visited the cities (i.e. the financial losses among planes; assuming, of course, that the city weights decrease as defeats increase)
- the knapsack problem [Kre&Sti, pp.7-8], in which the usual capacity constraints would be replaced by temporal constraints.

So, it appears that, like these problems, the airplane deicing scheduling problem is essentially NP-hard.

The good news for our purposes is that the problem has a general form of Combinatorial Allocation Problem. It may thus be interesting to look at it in terms of Generalized Vickrey Auction. It is not difficult to see how the problem can be handled in terms of GVA, which is the issue of the next subsection.

4.3 GVA for deicing scheduling problem

We design a GVA mechanism for the airplane deicing problem. Here, we only sketch this line of thought to illustrate the applicability of GVA framework, and to exemplify the notions and definitions of previous sections; computational aspects as well as implementation issues fall out of the scope of this essay.

In introducing an auction-based solution, let us first assume that the scheduler is an auctioneer, or principal agent, who sets up a GVA for every 30 minute sequence, starting at the beginning of each hour and 30 past respectively. Since these sequences introduce similar subproblems, it suffices to consider an auction for only one 30 minute sequence. Each airplane is associated with an agent $i \in \mathcal{I}$ that places a bid for obtaining the deicing during the time sequence in question. Let us furthermore suppose that the goods $\mathcal{G}$ represent the deicing trucks available for deicing, and we assume that one truck is capable of doing only 15 minute job during one auction sequence (for their tanks need to be filled etc.). In addition, we suppose that such deicing jobs that cannot be done as a whole during a single period are not to be accepted.

Bidding rules Suppose that each plane agent needing the deicing sends a bid to the auctioneer, before the start of the respective time sequence. Recall that $v_i(S)$ is agent i's valuation function for the bundle of items $S \subseteq \mathcal{G}$. Each agent announces its alleged valuation function v_i, i.e. preferences for combinations of deicing trucks. To simplify the problem, suppose there are only two sizes of airplanes, namely small ones that need 15 minutes of deicing (with one truck), and bigger ones that need 30 minutes of deicing (i.e. 15 min with two trucks). Moreover, let us suppose that plane agents can bid according to the following rules for values:

- a bid value is at least \$5 for each plane that wants a deicing
- a bid for an agent may be more than \$5, if some (or all) of the below conditions are satisfied, each condition increasing the bid for \$5:
  - there are business travellers on board
  - the plane is a slot plane
  - the plane is already late and off-schedule

Then, a valuation announced by an agent is simply a function, that takes needed bundle of deicing trucks as its argument and gives the price, that the agent is willing to pay for the job, as its value.

Allocation rules After receiving the bids, the GVA returns an allocation and payments to be made to the agents. The former, i.e. the allocation of deicing trucks, is obtained by computing S^*:

$$\begin{aligned} S^* &= \arg \max_{S=(S_1,\ldots,S_I)} \sum_i v_i(S_i) \\ \text{s.t. } \forall i,j &\in \mathcal{I} : S_i \cap S_j = \{\} \end{aligned}$$

Then, by Theorem 1, we know that the allocation returned by the GVA is exactly the one we are looking for. And finally, the payments for agents are to be computed with

$$p_{vick}(i) = (V_{-i})^* - \sum_{j \neq i} v_j(S_j^*).$$

as explained in subsection 3.1.

Let us next illustrate this GVA-based deicing scheduling process with a simple example.

Example 14 *Consider a set up of the table below, which gives the information of the planes waiting for deicing in a single 30 minute time sequence:*

Plane type:	**Job lenght:**	**Value:**
$Learjet$	15 min	\$20 (business, slot, off-schedule)
$Boeing747$	30 min	\$15 (business, slot)
$MD-11$	30 min	\$5 (normal plane)

Assume that there are only three deicing trucks available in this round, i.e. $\mathcal{G} = \{1,2,3\}$. Notice that here the agents would be $\mathcal{I} = \{learjet, boeing, md\}$ and their truthful valuations would be:

- $v_{learjet} : v_{learjet}(\{\}) = 0, v_{learjet}(\{1\}) = 20, v_{learjet}(\{2\}) = 20,$ $v_{learjet}(\{3\}) = 20$, and $v_{learjet}(S_j) = 20$ for all other bundles $S_j \subseteq \mathcal{G}$
- $v_{boeing} : v_{boeing}(\{\}) = 0, v_{boeing}(\{1\}) = 0, v_{boeing}(\{2\}) = 0, v_{boeing}(\{3\}) = 0, v_{boeing}(\{1,2\}) = 15, v_{boeing}(\{1,3\}) = 15, v_{boeing}(\{2,3\}) = 15$, and $v_{boeing}(S_j) = 15$ for all other bundles $S_j \subseteq \mathcal{G}$
- $v_{md} : v_{md}(\{\}) = 0, v_{md}(\{1\}) = 0, v_{md}(\{2\}) = 0, v_{md}(\{3\}) = 0,$ $v_{md}(\{1,2\}) = 5$, $v_{md}(\{1,3\}) = 5$, $v_{md}(\{2,3\}) = 5$, and $v_{md}(S_j) = 5$, for all other bundles $S_j \subseteq \mathcal{G}$

If the agents truthfully report their valuation functions, the mechanism finds an optimal solution, e.g. $S^* = (\{1\}, \{2, 3\}, \emptyset)$. Notice that S^* corresponds to the allocation where Learjet gets the deicing from truck 1 and Boeing747 gets its deicing from trucks 2 and 3, but MD-11 still has to wait its turn, and attend the next auction. The important thing to note here is how the mechanism calculates the set of optimal solutions, including $S^* = (\{1\}, \{2, 3\}, \emptyset)$. This is done by confirming that there does not exist a solution with higher value than 35:

$$v_{learjet}(\{1\}) + v_{boeing}(\{2, 3\}) + v_{md}(S_j) = 20 + 15 + 0$$

Finally, the mechanism goes on to compute the payments for all agents. In this example, assuming that the returned allocation is $S^* = (\{1\}, \{2, 3\}, \emptyset)$, the payments for learjet and MD-11would be \$0, and the payment for Boeing747 would be \$5.

Hopefully, this example gives an idea, how the Generalized Vickrey Auction can be used in solving the Combinatorial Allocation Problems.

5 Conclusion and Further Issues

We have discussed the GVA in the general CAP as described in chapter 2 of [Par]. Moreover, we have seen the application of this GVA-based approach to the particular scheduling problem, airplane deicing scheduling. Since the GVA involves essentially NP-complete problems, including the computation of efficient solutions in CAP, an interesting line of thought would be to study the possible use of various NP-solvers in the GVA-based approach to the CAP. This falls out of the scope of this essay, however.

6 Bibliography

Par David C. Parkes, 2001. Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania. See http://www.eecs.harvard.edu/ parkes/diss.html.

Osb&Rub Martin J. Osborne and Ariel Rubinstein, 1996. A Course in Game Theory. The MIT Press, London.

Fud&Tir Drew Fudenberg and Jean Tirole, 1991. Game Theory. The MIT Press, London.

Vic61 William Vickrey, 1961. Counterspeculation, auctions, and competitive sealed tender. Journal of Finance, 16: pp. 8-37.

Kre&Sti Donald Kreher and Douglas Stinson, 1998. Combinatorial Algorithms: Generation, Enumeration and Search. CRC Press, London.