TEL AVIV UNIVERSITY
The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slaner School of Graduate Studies

The Decentralized Field Service Routing Problem

A thesis submitted toward the degree of
Master of Science in Industrial Engineering

By

Edison Avraham

August 2014

The final version of this thesis appeared in Transportation Research Part B, 104 (2017) 290–316

TEL AVIV UNIVERSITY
The Iby and Aladar Fleischman Faculty of Engineering
The Zandman-Slaner School of Graduate Studies

The Decentralized Field Service Routing Problem

A thesis submitted toward the degree of
Master of Science in Industrial Engineering

By

Edison Avraham

This research was carried out in the Department of Industrial Engineering
under the supervision of Dr. Tal Raviv and Prof. Eugene Khmelnitsky

August 2014

The final version of this thesis appeared in Transportation Research Part B, 104 (2017) 290–316

Acknowledgements

To my advisors Dr. Tal Raviv and Prof. Eugene Khmelnitsky I wish to express my deepest gratitude. It has been a long (and fruitful) journey that could not have ended this way without your support, guidance and insights. I have certainly learned a great deal about the nature of academic research during our mutual work and for that – I will be eternally grateful.

I wish to thank the employees of ClickSoftware for their willingness to talk about this research, its methodology and findings. In particular, I wish to express a special gratitude to Israel Beniaminy, Senior Vice President of Product Strategy in ClickSoftware for his insights and advices. These were especially valuable.

I wish to thank my Mother, Malka, and my siblings, Vered and Gil, for their endless support and encouragement throughout this long process.

Last, but definitely not least, I wish to express my deepest gratitude to my father, Sahilu Avraham. As long as I can remember, my education has been his prime focus and this achievement is his no less than it is mine.

Abstract

Large companies often outsource their field service tasks to smaller contractors. Since sharing of private information between the various parties is not always possible, the common practice is allocating the tasks to the contractors heuristically. With this, the tasks for which the contractors have committed to other companies are not considered at all. As a result, the allocation can be inefficient. This study develops 2-stage collaborative mechanisms that cope with the problem and result in nearly optimal solutions.

In the first stage, a feasible, not necessarily optimal, allocation of tasks to contractors is generated. We consider several possible allocation procedures such as sequential Vickrey auctions, sequential combinatorial auctions and sequential negotiation. The sequential combinatorial auctions procedure implements the Generalized Vickrey auction mechanism, which is a strategy-proof mechanism for the allocation problem of multiple goods among several competing agents.

In the second stage, the contractors are allowed to exchange the tasks among themselves so as to decrease their operational costs. The exchanges may or may not include money transfers.

The quality of the generated allocation is evaluated according to various performance measures. It was found that the first stage procedures yield fairly efficient allocations and the second further improves it. The resulted allocations are considerably more efficient compared with the solutions generated by a reasonable benchmark heuristic. The allocations' costs are close to a lower bound established by the optimal allocation of a central planner. That is, the price of anarchy is shown to be small.

Table of contents

List of Figures

List of Tables

List of Notations

Symbol	Meaning
N	Number of company's tasks
K	Number of contractors
I^C	The set of tasks owned by the company
k	Index of contractors and their depots.
I_k	Set of pre-committed tasks of contractor k
i, j, m	Index of tasks
s_i	Service time of task i
c_{ij}	Traveling cost from task i to task j
t_{ij}	Traveling time from task i to task j
f_1	Regular time hourly salary of the field service team
f_2	Increase in salary of the field service team in overtime
L	Number of regular working hours
M	Maximal number of working hours. That suggest $\mathrm{M}-\mathrm{L}$ overtime hours are allowed.
I	Set of all locations of a problem.
x_{ijk}	$x_{ijk} = \begin{cases} 1 & if\ contractor\ k\ travels\ from\ i\ to\ j \\ 0 & otherwise \end{cases}$
u_{jk}	Arrival time of contractor k at location j
T_k	Total working time of contractor k
E_k	Overtime duration of contractor k
C	Set of nodes that constitute a sub-tour in the solution of the problem of the central planner
0	Depot city of the company
n	Number of tasks in one cluster
n_{max}	Maximal size allowed of cluster
$\mathcal{C}$	The set of tasks in the cluster
$P(\mathcal{C})$	The power set of $\mathcal{C}$.
$p_{k,S}$	The cost of the offer of contractor k for subset S
$x_{k,S}$	$x_{k,S} = \begin{cases} 1 & the\ contractor\ k\ serves\ subset\ S \\ 0 & otherwise \end{cases}$
$Z^*_{\{1,..,K\}}$	Optimal total cost for contractors in a single combinatorial auction
$x^*_{k,S}$	Optimal solution of the winner determination problem with all contractors.
$Z^*_{\{1,..,K\}\setminus\{k\}}$	Total cost for contractors when contractor k is excluded from the auction
P_k	Payment to contractor k
r_i	Prize offered to contractor for serving task i
$g(s_i)$	Initial prize for serving task i
a	Increase rate of r_i
$\widehat{I_k}$	Set of tasks contractor k is committed to
I^s	Set of offered tasks the contractor commits on
x_{ij}	$x_{ij} = \begin{cases} 1 & if\ contractor\ k\ travels\ from\ i\ to\ j \\ 0 & otherwise \end{cases}$
u_j	Arrival time of contractor at location j
T	Total working time of contractor
E	Duration of overtime
a_{ij}	Change of cost inflicted to the contractor that serves i from exchanging task i with task j
$\widetilde{I_k}$	Set of company's tasks allocated to contractor k
y_{ij}	$y_{ij} = \begin{cases} 1 & if\ task\ i\ is\ exchanged\ with\ task\ j \\ 0 & otherwise \end{cases}$
$\tilde{C}$	Set of tasks that constitute a circle in the problem of maximizing the total weight of disjoint circles in a directed graph
$\tilde{S}$	Set of contractors that own the tasks in $\tilde{\mathrm{C}}$
π_k	Profit of contractor k $\in \tilde{\mathrm{S}}$
i_1, j_1	Tasks in $\tilde{\mathrm{C}}$
$\bar{a}_{i_1 j_1}$	True profit of contractor from trading i_1 for j_1

1 Introduction

Field service organizations operate in a dynamic, ever-changing world which combines long-range planning with emergency responses. While this could be said to apply to most types of businesses, there are two characteristics of field service that make it particularly challenging. First, the main “raw material” used in “producing” field service are work hours which must be applied at the right time – when the service is provided. Unlike tangible raw materials, and working hours used to produce tangibles that can be stored, a supply of work hours cannot be stored until a time when there is a demand for it. Field service managers describe this fact as “work hours have zero shelf life”, or even more briefly as “use it or lose it”. The second characteristic of field service is that resources and demands all have physical locations (e.g. location of the service engineer, location where the task needs to be performed). This brings travel times to the forefront of any attempt at optimization.

Most commercial packages assume that the optimization is done by a *central planner* that is trying to optimize a single objective function and has access to all relevant data. The optimization process often includes employing algorithms that are *distributed* – that is, written in such a way that several computing resources can share the required computational work. However, the algorithms are typically *centralized* – that is, they allow each computing resource to access any item of problem information, so that the computing process may be considered to be performed by a central aggregated computing system. However, in many cases this assumption is not valid: In such cases, there are many interacting agents, each of which has its own resources, where all the resources (e.g. technician, trainer, heavy-equipment operator) need to agree on accepting a certain task and on the task's timing. Typically, each agent has private information not shared with the other agents.

In this study we formulate the Decentralized Field Service Routing Problem (DFSRP), which requires a company to allocate a set of tasks to several contractors. Each task is characterized by its location and service time. In addition, each contractor is pre-committed to its own set of tasks which are not revealed to the company. Each contractor is interested in maximizing his revenue net of the labor and routing costs. The company, on the other hand, is interested in minimizing its payments to the contractors.

Such a situation occurs in many real-world processes where providing the service involves several business entities, some of which have tasks that they need to perform, and some others have resources which may perform these tasks. For example, service firms (that have tasks to perform) that provide the service by outsourcing the tasks to contractors, where each of the contractors controls one or more vehicles. Contractors do not share their business information, such as the number and the availability of their vehicles, with the service firms. This lack of information sharing leads to inefficient (and therefore expensive) schedule and routes of the agents. Even worse, it leads to failures to deliver the work on time.

In this study we suggest a decentralized 2-stage mechanism for the allocation of the service tasks to the contractors. This mechanism does not require the contractors to reveal their private business information. The quality of the generated routing and allocation are measured both from the point of view of the society (efficiency) and from the point of view of all the business agents (profits). The business relations between the agents are assumed to be of a long term nature.

The thesis is organized as follows: In Chapter 2, we review the relevant literature regarding the solution of routing problems and decentralized problems. In Chapter 3, the DFSRP is defined. In Chapter 4, we present several variations of a 2-stage task allocation mechanism. In Chapter 5, we benchmark these mechanisms and study their properties. Chapter 6 concludes the thesis and suggests directions for future research.

2 Literature review

In this chapter we review the Field Service Scheduling (FSS) optimization problem, compare it with the VRP and present several solution algorithms for centralized settings of the problem. Thereafter, we discuss the methodology of solving the decentralized VRP and introduce several concepts from auction theory and mechanism design that assist in developing a decentralized solution method. Finally, a gap to be addressed by this study is identified.

2.1. The Field Service Scheduling (FSS) Problem

The Field Service Scheduling problem has been the prime focus of many studies carried out over the last two decades. Its main attributes are described quite comprehensively in Zerdin et al. (2011) that discuss the problem as a combinatorial NP-hard optimization one. The problem schedules technicians (**resources**) to serve a set of tasks (**demands**), scattered at different locations. When a technician serves a task, he is awarded with a prize. Each technician has a different set of skills and his capacity is limited. Serving a task requires certain skills, and service time of each task is known. The time and cost of traveling from one task to another are known, as well.

The objective function is maximizing the total sum of awarded prizes minus the total traveling cost. Typically, not all tasks are able to be served due to limited capacities of the resources. A common variant of the FSS is The Field Service Scheduling problem with Time Windows (FSS-TW) where the time a certain task started and completed is restricted.

The authors of this paper claim that if the objective function of the FSS problem is modified and some of the constraints are relaxed, FSS instances become similar to instances of VRP. Yet, FSS differs from the VRP in the following:

1. The resources in FSS are different not only in capacity, but also in quality. Each resource can serve certain tasks and there is no assumed relation between the set of tasks one resource is capable of serving and the set of tasks another resource is capable of serving.
2. In FSS, not all demands are served due to limited availability of resources. The possibility of having the demands which are not served increases the complexity of the problem. In VRP, all demands must be served[1].
3. The objective function in FSS is different from the objective function in VRP. VRP is usually solved with the goal of minimizing either the total traveling time or the number of routes, or some weighted average of the two. On the other hand, FSS is usually solved to maximize total profit (prizes minus traveling costs) and, if possible, minimize traveling time.
4. Other characteristics:

[1] This is true whether the capacity of the vehicle is large enough to serve all demands or whether the capacity of a single vehicle is small but the number of available vehicles is enough.

a. Service times of the tasks (in VRP the times are usually 0, whereas in FSS the times are positive parameters).
b. Limits on the total duration of a route
c. Multiple depots (possible in FSS, not standard for VRP)

Following that, the authors suggest that certain instances of FSS can be transformed to instances of VRP. They note several examples. The first is FSS instances where capacity of resources is enough to satisfy all demands, so that the problem can be transformed to VRP with given time windows and vehicle parameters[2]. The second is instances where all resources are able to serve all tasks, and the problem reduces to VRP with time windows.

Therefore, assuming that all resources are capable of serving all tasks and that their capacity is large enough to serve all tasks, our FSS instances can be modeled as a generalization of the VRP that includes the following features:

1. Service times of the tasks are greater than 0.
2. Multiple depots (one for each contractor) exist.
3. Maximal duration of working day for all contractors.

Indeed, this is the way we follow when modeling and solving FSS settings. Although the properties of the settings enabled us to cope with them using a commercial optimization solver, we find it appropriate to include various solution algorithms for the FSS problem in this survey. These are now presented.

2.2 Selected solution algorithms for the FSS

Various solution algorithms for the problem are presented in numerous studies. We discuss three of them that highlight important components of the state of the art optimization methods. The first study suggests a method based on multiple processors. The second study solves a stochastic problem using a Branch and Bound (B&B) algorithm and the third one demonstrates the use of state of the art heuristic optimization methods (Genetic algorithms and Ant colony).

Xu and Chiu (2001) suggested a heuristic solution for a scheduling problem with K service technicians and a planning horizon of one working day. A solution to the problem is an allocation of tasks, scattered in different locations, to a set of service technicians, while satisfying time window constraints and considering predefined priorities. The technicians own diverse skills required to serve the tasks[3].

[2] Traveling time in this case is the sum of actual time spent in traveling and the service time.
[3] The authors represent the level of capability of a certain technician to perform a certain task by a fraction which varies from 0 to 1.

The authors solve the problem by transforming it to a generalized version of the VRP with Time Windows (VRPTW), tailored to consider unidentical resources. The generalized version of VRPTW differs from the classical VRPTW in its objective function. While VRPTW is solved in order to minimize either the number of routes or traveling distance, the generalized problem is solved mainly in order to maximize the number of served tasks[4] in a given period of time. A secondary goal is minimizing the total working time (sum of traveling time and waiting time). An additional goal is minimizing the total traveling time. Furthermore, the generalized problem differs from the standard VRPTW in its constraints. While VRPTW's solution is subject to limited capacity, the generalized problem's solution is constrained due to the differences in skills of the service technicians.

The authors present a mathematical model for the problem and solve it heuristically by applying a 2-stage algorithm. The output of the first stage is an initial feasible solution of the problem constructed using a *Greedy* algorithm previously developed by the authors. This solution is an input for the second stage, in which a local iterative search is operated to improve an initial solution. A solution's neighborhood is defined rather standardly and includes solutions generated by exchanging tasks between technicians, moving tasks from one technician to another, etc. The objective function is to maximize the total free time of the technicians[5].

Next, the authors modify the construction process of the greedy solution by including randomness based on the GRASP (Greedy-Randomized Adaptive Search Procedure) approach. As a result, several greedy solutions are constructed. Each of them is processed by a different processor, with a local search being applied to the solutions. When some processor reaches a local optimum, the search ends and the local optimum is kept. Then, a new random initial greedy solution is generated and sent to the idle processor that operates a local search on the new solution. This algorithm can be stopped by setting a time limit or a limit on the amount of total operations performed. Numerical experiments show that the modified algorithm yields the best solutions to the problem with respect to the predefined measures. However, the difference in performance between the algorithm's versions is not significant and the running time for the modified algorithm is considerably larger.

The authors also present upper and lower bounds for the optimal solution. Additionally, they develop a more general version of the problem, where the number of possible depots is larger than 1 and overtime is allowed. The authors do not address randomness in service times or traveling times. Instead, they note that minimizing the total working time increases the robustness of the generated solutions. This, according to the authors, is beneficial if randomness exists.

[4] This number is weighted with the predefined priority of the task.

[5] Xu and Chiu present a procedure which receives a specific technician and the list of tasks allocated to him and returns a feasible scheduling that maximizes the free time.

Unlike Xu and Chiu, Hadjiconstantinou and Roberts (2002) address randomness directly. They solve a VRP with stochastic service times and constant traveling times. The authors state the problem and claim that identifying a solution for each specific scenario of service times realization is not practical and suggest instead a policy that maximizes the expected performance. The problem deals with several tasks to be carried out by a single service technician. The planning horizon is K working days. The length of a single working day is known. Overtime is allowed and penalized. The primary objective is minimizing the total transportation costs. A secondary objective is minimizing the total expected overtime.

The authors claim that the problem can, in principle, be solved straightforwardly by formulating and solving an extremely complicated 2-stage stochastic programming model. The first stage is identifying possible schedules by solving an Integer Linear Programming model. This stage is followed by formulating a recursive program designed to find the total overtime of each schedule. Due to the complexity of this procedure, the authors choose to solve the problem using a Branch and Bound algorithm.

The primary data structure of the B&B algorithm is a paired tree comprised of 2 trees. The first one is a decision tree (travel/not travel from task i to task j). The second tree maps each possible branch in the first tree to the corresponding set of schedules that can result from choosing this branch. The set of possible schedules is constructed according to the distributions of service times of the tasks served in a specific branch. The authors also calculate lower and upper bounds of the total overtime in each branch.

The authors comprehensively describe the real life scheduling problem they solve using the B&B algorithm. It consists of a set of tasks scheduled to service during a 5-day working week of a single service technician[6]. The length of a working day is set at 8 hours and 15 minutes. The service times as well as the set of tasks to be scheduled are random, since this set consists of both planned tasks (weekly/monthly maintenance) and unexpected tasks (damages needed to be addressed immediately). The latter are unknown in advance.

This problem is solved in two ways. In the first one, an optimal schedule is generated at the beginning of the week by applying the algorithm with the planned tasks as its input. The schedule is modified when information regarding an unexpected task arrives according to a set of rules defined in the paper. This generates the *actual-optimal* scheduling. The algorithm does not imply re-optimization. According to the second method, the schedule is re-optimized at the end of each working day. The method considers the unexpected tasks that arrived during that day. Both the "actual-

[6] The problem was solved retrospectively i.e. according to data obtained in a certain week.

optimal" and re-optimized schedules are compared to the actual one followed by the service engineer[7]. Both schedules proved to perform better than the actual one.

Beniaminy et al. (2009) present several solution algorithms for a FSS problem which is a generalized version of the VRPTW. First, a genetic algorithm (GA) is presented. According to the GA, a *chromosome* represents an assignment of demands (service tasks) to resources (service technicians). Each chromosome consists of genes. A gene is a pair (R_i, D_j) implying that demand D_j is assigned to technician R_i. A valid chromosome is a chromosome where each demand is assigned at most once. The actual scheduling that fits a chromosome is constructed by starting the service of demands as early as possible considering traveling times and the other constraints. A chromosome can represent an infeasible schedule, since resources are limited. When an initial population of schedules has been randomly built, the next generation is created in the following manner. The current population is segmented and the first segment "elite" moves the best solutions on to the next generation. All the other segments undergo some variation of crossover after parents have been chosen randomly using a uniform distribution, or mutation. The GA was found to perform well when applied to a real case study with 620 demands and 88 different resources with a variety of required skills.

The second algorithm is an Ant Colony Optimization (ACO) algorithm. According to this algorithm, an arc (A, B) in a *pheromone table* describes the worthwhileness of assigning demand B immediately after demand A. Since the technicians differ in their skills and in the set of assigned demands, a separate pheromone table for each resource is created. The algorithm has 2 levels of operation – a higher level (the *controller*) and a lower level (the *builder*). The builder assigns demands to resources based on the current pheromone tables. In order to avoid convergence to a local optimum, randomness is added to the creation of assignments. The controller uses the builder to create full-range solutions. Then, the pheromone tables are updated based on the solutions where either the objective function's value is high, or the number of supplied demands is high. The process repeats until a stopping criterion is reached. Since more than one colony can be established, the process can be done using multiple processors. From applying the algorithm to two case studies it was evident that considering separate pheromone tables had a significant effect only when resources were non-identical. For identical resources no considerable effect is reported.

The third algorithm combines ideas from the ACO and the GRASP algorithm. The first stage of the algorithm constructs an initial solution with randomness taken in the choice of the next step of the solution construction. The second stage improves the first one by applying a local search. The suggested algorithm uses a pheromone table in both stages of GRASP with the intent to store the desired components of the solution that are learned through various iterations. Preliminary results indicate that applying this method improves GRASP considerably.

[7] This schedule was probably built by the company, heuristically.

As we have shown, most researchers solve a centralized problem either by developing and implementing heuristic algorithms, or by using mathematical programming methods. Data required to solve the problem is known and available.

Solving the problem centrally bares a hidden assumption that all components of the system (vehicles/service technicians, clients/service tasks and the central planner) share the same interests. This assumption is accurate when both the central planner and the service technicians belong to the same business entity. However, when the service technicians are independent contractors serving the tasks of many companies, their interests are clearly different from those of the companies. In fact – they are even opposite. The contractors aim at maximizing income, while the companies aim at minimizing their costs associated with the payments to the contractors.

Therefore, much of the data required to solve the problem centrally is usually unavailable and a centralized solution to the problem can't be obtained. Furthermore, a solution of this sort has little value, since different business entities involved in the decision making do not have to follow it. A decentralized solution method, which considers different interests of the business entities, must be applied. This is the topic of the following section.

2.3 Solving Decentralized problems

We now review the literature that deals with the solution of Decentralized problems. This area has been the primary concern of many studies done under the discipline of computer science in relation to artificial intelligence.

Smith (1980) developed the CNP (Contract Net Protocol) with the intent to efficiently distribute tasks, which are to be processed by a set of decentralized loosely coupled Knowledge-Sources located at different processors. An efficient distribution of tasks allows a certain node (Knowledge-Source), which is requested to process a task and is busy in processing a previous task to transfer the new task to another node that is idle. The CNP was developed in order to define the communication between the nodes before and during the transfer.

The mechanism that regulates the transfer of tasks from a busy node to an idle one is similar to a negotiation towards signing a contract. It has four characteristics: (1) local process with no central control (2) bilateral exchange of information (3) each side analyzes the information from its perspective (4) a final contract is signed by a mutual agreement.

According to Smith (1980), the set of nodes is a contract net. Deciding what node processes a task is a contract between two nodes – the *manager* and the *contractor*. The contractor actually executes the task. A node can be the manager of a certain task and a contractor of another task. The contract is signed on the basis of mutual local selection based on bilateral exchange of information.

This exchange begins when the manager sends out a message (in a format defined by the CNP) regarding a task to be processed by a contractor. Each contractor examines the list of messages and chooses the messages which he would like to bid on. After all bids have been received, the manager examines the bids, chooses the most suitable contractor and transfers the task to him for processing. That contractor can become a manager too, if he chooses to send out a message regarding the task he had received offering it to other contractors.

Smith's work suggested the task distribution mechanism that became the basis to other mechanisms developed later for distributing clients to be served by vehicles[8]. These mechanisms can be classified into two types; the first – mechanisms designed to solve a complex centralized problem using distributed algorithms and the second – mechanisms designed to solve a decentralized problem. Examples of mechanisms from both types are presented in the following sections.

2.3.1. Solving Centralized problems using distributed algorithms

Kohout and Erol (1999) present an agent-based method developed to cope with a dynamic Pickup and Delivery problem with Time Windows (PDPTW). They developed a computationally efficient algorithm that identifies a solution to the problem and then improves it by a stochastic mechanism. The use of agents allows decomposing the problem into several sub-problems. Each sub-problem can then be solved independently and simultaneously using parallel computers.

The authors suggest a configuration that consists of two types of agents – *client* agents and *vehicle* agents. The solution process is as follows: a client agent announces his need for a vehicle. Then, each vehicle agent estimates added costs associated with the client. This is done by implementing an adapted version of Solomon's (1987) algorithm. Then, payment requests are sent to the client. The client chooses the vehicle with the smallest payment. Once all clients are assigned to vehicles, the stochastic improvement stage begins. Clients, chosen randomly, can re-announce a need for service. Then, the bidding process is repeated and the client can be assigned to another vehicle. The solution process is quite quick so new clients, that are not known in advance, can be assigned to vehicles dynamically. Numerical experiments

[8] Smith and others exploring the CNP thoroughly discuss the actual transportation of data, its bottle necks, computational load etc. in their work. Our focus is on the contractual mechanisms developed in their work.

with static instances indicate that this algorithm yields fairly good results compared to the Solomon's adapted algorithm.

Thangiah et al. (2001) suggested a decentralized solution to a VRP in order to distribute computations among several computers. Their objective is minimizing the number of vehicles and the traveling distance. They discuss the concept of an intelligent agent, who is independent, communicative and goal-driven, and define two types of agents in the solution process – a *scheduling* agent and a *vehicle* agent. Their solution method consists of two stages. At the first stage, the scheduling agent announces the client's data to all vehicle agents. Each of them calculates his bid, based on the added cost of serving that client. That added cost is estimated by implementing the Clarke-Wright heuristic. After all bids have been calculated and sent to the scheduling agent, the latter allocates the client to the lowest bidder. This process repeats until all clients are allocated. At the second stage, vehicle agents are allowed to exchange clients between them to decrease their costs. The obtained solution proved to be considerably worse than the best one generated when solving the problem centrally. The authors claim that it is the result of the use of a very simple Clarke-Wright heuristic.

Leong and Liu (2006) solve a Capacitated VRP with time windows. Their primary objective is minimizing the number of routes. Their secondary objective is minimizing the total traveling distance. The solution process is comprised of two stages. The first stage builds initial feasible routing of vehicles by applying a heuristic method presented by Solomon (1987). The second stage improves the initial routing by applying a decentralized multi-agent algorithm.

The authors model clients and vehicles by agents. Each *client* agent aims at minimizing his waiting time. Each *vehicle* agent aims at maximizing the utilization while minimizing its traveling distance. A client agent is allowed to share information with the other client agents. A vehicle agent is allowed to share the information regarding the clients it serves with the other vehicle agents. The authors also define a *global planning* agent who is responsible for all vehicles and is aware of the clients' data and the vehicles' data.[9] He aims at minimizing the number of routes and the total traveling distance. [10]

According to the developed algorithm, the construction of the initial route is followed by a distribution of relevant information among the agents.[11] Then, the negotiations between clients and vehicles offering possible *transitions* begin. A *transition* is a

[9] Examples of data characterizing a client: it's time windows, the vehicle serving it, exact time the vehicle arrives at the client etc. Typical information regarding a vehicle is its route and its capacity.

[10] Leong and Liu define a cost function weighting the number of routes and the total traveling distance.

[11] Relevant information of a client is what vehicle serves him and when that vehicle arrives. Relevant information for a vehicle agent is the list of its clients, and data regarding them.

valid operation on the current routing such as exchanging clients between vehicles, transferring clients from one vehicle to another, etc. A client is allowed to suggest a transition in which it is transferred to be served by a different vehicle. He can also suggest a mutual exchange of vehicles that includes another client. A vehicle is allowed to suggest the removal of a client from another vehicle's route followed by the insertion of the client into his route. He can also suggest an exchange of two sets of clients (the size of the sets can be greater than 1) between two vehicles.

The global planning agent is responsible for selecting the transitions to be carried out from the list of transitions and for executing transitions in order to improve the performance of the system as a whole. The execution of the chosen transition is followed by optimizing all routes. This is done by the global planner that implements a 2-opt heuristic. Then, the global planner examines the new routing and finds routes in which the capacity of a vehicle is not fully utilized. The global planner wishes to eliminate these routes by transferring their clients to other vehicles. The process repeats until no improvement can be achieved.

The authors apply their algorithm to solve several benchmark problems presented by Solomon and report that the algorithm produced good solutions, compared with other heuristics (SA, TS) as well as with the best known.

Zhenggang et al. (2009) also solve a Capacitated VRP with time windows by implementing a modified version of Smith's CNP. They aim at minimizing the sum of traveling distances and present a system of agents comprised of multiple *vehicle* agents and a single *scheduling* agent, responsible for planning and controlling the vehicle agents.

This method allocates an order (defined as a request to supply a certain amount of products at specific time window and location) to a vehicle in the following manner:

- an order is announced by the scheduling agent that sends its details to the relevant vehicle agents;
- each vehicle agent estimates the feasibility of serving this order (considering previous orders and capacity constraints);
- if serving this order is feasible, the vehicle agent sends a bid for the order. This bid is formed based on an approximation of the marginal cost added to the agent if he serves the order. The scheduling agent then allocates the order to the lowest bidder.

As implied, not all agents are relevant. A relevant agent of a newly announced order is defined as an agent that at least one of the orders he serves is relatively close to the announced order (the distance between them is less or equal to a predetermined value). The authors claim that this definition improves the CNP.

The numerical experiments with the Solomon's benchmark problems show that the amount of negotiations held when implementing their protocol is only 30% of the amount of negotiations held implementing the original CNP. The total negotiation duration decreases by 30%.

2.3.2 Solving decentralized problems

Rassenti et al. (1982) present a version of a combinatorial auction (more on this – to follow) applied in order to find the allocation of various airports' time slots to competing airlines so as to maximize the sum of values of slots allocated among all airlines. A time slot can be a slot for departure or a slot for arrival. Since values for these slots are generally correlated, grouping of slots into packages allows bidding on them by a combinatorial auction and allocating them to the airlines.

A linear integer programming model is presented. It aims at maximizing the sum of bids of packages among all airlines while satisfying the airlines' constraints. The combinatorial nature of the auction, which resulted from the correlation between values of slots, is expressed in the model as a set of constraints. Mathematically, the problem is a version of the set-packing problem and is solved using a previously-known algorithm.

For the allocated packages, the airlines pay prices. A price of a package is no greater than its bid and is determined in the following manner: first, the shadow price of each time slot is found; then, the price of a package is set at the sum of shadow prices of the slots it contains. Each airport receives the sum of shadow prices of the slots it owns.

The proposed auction does not ensure that truthful bidding is a dominant strategy (more on this – to follow). However, the authors claim that speculations can be risky when an agent does not know the bids and preferences of the other agents, especially if the auction is combinatorial. Multi-period lab experiments seem to validate this claim. When the users tended to bid true preferences, the mechanism gave results close to the global optimum.

Sandholm (1993) expands the CNP by defining the negotiation between the nodes. He notes that Smith hadn't defined how contractors form bids for tasks[12], and how managers find winning contractors. Sandholm addresses these issues by using the marginal cost criterion.

[12] A task is defined as a client or a set of clients to be served.

Sandholm solves a transportation problem comprised of several distribution centers that operate in overlapping geographical areas. Each center is owned by a different company and is responsible to serve some amount of clients in different locations. Each center owns a fleet of un-identical vehicles and aims at minimizing its transportation cost. Each vehicle is characterized by its own traveling cost per km, maximal length of route, maximal traveling duration etc. All clients must be served.

Sandholm models each distribution center with a single agent and constructs the solution to the problem in the following manner: first, each agent, seeking to minimize his total transportation cost, solves his own routing problem and finds an allocation of its customers to the vehicles. Thereafter, each agent can negotiate with other agents in order to receive or transfer clients in exchange for a payment. A negotiation ends successfully if transferring a client from one agent to another is profitable to both agents. By repeating this process, the initial allocation is improved, without solving the problem centrally.

Each agent has two components – a *bargaining system* and a *local optimizer*. The bargaining system has four functions that are periodically run by the agents. The second, third and fourth function run first. The first function runs last. All functions use the local optimizer that is able to find the marginal cost of tasks and to optimize the routing program of the agent.

The first function finds the tasks that can be transferred to other agents and announces these tasks in an auction.[13] An Announcement of a task includes its clients' data and a maximal payment for the task, based on the marginal saving the agent gains when the task is transferred to another agent.

The second function is responsible for finding tasks announced by other agents and for building and sending corresponding bids to these agents. Constructing a bid is done using the marginal cost criterion – the cost added to the agent if he serves the task's clients[14]. No agent has information regarding the bids of the other agents.

The third function examines the bids received in the auction and allocates each task to its lowest bidder when the marginal saving gained by the announcing agent is larger than the lowest bid. The forth function integrates newly won tasks into the winning agent's routing program.

[13] Sandholm notes that usually, the tasks announced by an agent are the ones close to the operation area of other agents.

[14] Estimating the added cost is quite hard due to the fact that the set of tasks the agent serves is not known. Its computational complexity is exponential.

Sandholm also solves an actual routing problem with five distribution centers located in Finland. Three centers are owned by one company and the other two centers are owned by another company. Sandholm generated two solutions to the problem. The first solution was generated using a time limit of 15 minutes and the second was generated using a time limit of 30 minutes. Both solutions yielded lower costs compared to the routing the companies actually followed.

As shown, some researches develop and implement a two stage solution algorithm that uses agents to cope with optimization problems, typically versions of the VRP. The first stage ends when a feasible, not necessarily optimal, solution to the problem is generated. Thereafter, the second stage, during which the solution can be improved by negotiations between the agents, is operated. A successful negotiation results in the exchange of tasks between vehicles. That exchange generally improves the status of the system as a whole.

This procedure seems adequate for our needs, particularly the improvement stage done by an exchange of tasks[15]. However, most researchers do not deal with a configuration where various agents present in the problem may have different interests, and therefore can't be expected, a priori, to share private information, for example – information on their costs.

When the problem is decentralized, the use of agents simulates a real life configuration where various independent entities are at competition. Different interests obviously exist. Ignoring these interests can result in a poor solution. This is especially true when the agents themselves are contractors that serve tasks that originally belong to other companies and the payments are determined by mutual interaction. A priori, there is no reason to anticipate the sharing of information between the parties. Therefore, the need for a mechanism that encourages communication between them is evident. Mechanisms of this sort were developed under the disciplines of mechanism design and auction theory.

2.4 Auctions

An *auction* is a method of buying or selling objects. This is done by receiving bids for an object from potential buyers and selling that object to the highest bidder. An auction can be considered as a game with incomplete information, and classified according to several parameters.

An auction can be *open* or *sealed*. In an open auction, all participants hear or see each other during the bidding process and, therefore, can offer counter bids. In a sealed bid auction all participants offer their bids simultaneously and no participant has any

[15] An interesting mechanism for the exchange of tasks can be found at Bachem et al. (1996)

information on other participants' bids. Additionally, the announced object can have a *private value* or a *common value*. An object is said to have a common value, when the value of the object is identical among all participants and unknown. In the case of private value, its value is known, and differs among the participants. Lastly, auctions can announce a single item, several identical items or several distinguishable items.

Zamir et al. (2008) present several common auctions of a single object with private value. Among them are the *sealed-bid first price auction* and the *sealed-bid second price auction*, also known as the Vickrey auction. In these bids, each participant inserts into a box a sealed envelope containing his bid for the object. Once all participants have inserted their envelopes, the auctioneer opens the envelopes and awards the highest bidder with the object. In a first price auction, the highest bidder pays his bid for the object. In the second price auction, the highest bidder pays the second highest bid.

The authors analyze the first price auction and present a suitable symmetric equilibrium strategy and the expected profit of the participants. This requires making some assumptions regarding the distributions of private values of all participants and regarding the knowledge each participant has on other participants' distributions of private values. Similar analysis for the second price auction is much simpler, since bidding truthfully in the second price auction is a dominant strategy for all participants[16]. This result holds without the need for any of the assumptions made for the first price auction analysis. When applying the second price auction, the participant with the highest private value wins the bid. This result is typically desired.

However, the second price auction has shortcomings as discussed in Sandholm (2000): the possibility to create coalitions in order to decrease the selling price; an untruthful auctioneer; lower revenues for the seller compared with the English auction and the loss of dominance of the truthful bidding strategy. The latter two shortcomings are relevant when the value of an item is not private (i.e. common). Elaborating the common value issue, Sandholm addresses his 1993 study (presented in this survey earlier) and argues that the value of an auctioned transportation task that may be, at least in part, non-private if sub-contracting (transferring the execution of a task from one vehicle to another) is allowed. This is true, since the profitability of winning the task for a certain vehicle may be affected from the ability of other vehicles to execute the task as sub-contractors.

Thereafter, Sandholm (2000) discusses additional shortcomings of the second price auction associated with allocating several distinguishable items with correlated values by a sequential Vickrey auction. In this case truthful bidding (following myopic strategies) can lead to an inefficient allocation of items among the buyers, while the looking ahead strategy can result in a more efficient allocation. This is demonstrated

[16] When the dominant strategy for all participants in an auction is bidding truthfully, the auction is said to be strategy proof.

by presenting and analyzing an example of 2 transportation tasks where transportation costs are common knowledge. Sandholm suggest that for the common knowledge case, an optimal strategy for each bidder (buyer) exists and should be found by constructing a search tree[17].

Sandholm argues that a simultaneous auction, when the buyers bid for several single items at the same time is not an appropriate solution for the correlated values case, since the value of an item for a certain agent depends upon the other items that are in the agent's possession (see further discussion of simultaneous auctions in Milgrom (2000), Sandholm (2002)).

Sandholm proves also that for risk-averse agents, truthful bidding is not necessarily a dominant strategy in the second price auction if the value of the item is uncertain. Additionally, when an agent is allowed to transfer money as to eliminate that uncertainty, speculation over other agents' preferences is beneficial.

Despite possible weaknesses we implemented a sequential Vickrey auction as one of the options available for the company to allocate its service tasks to contractors (see section 4.1.1). When the set of offered tasks is large enough and there is little or no common knowledge between contractors, myopic contractors' strategy in the sequential auction seems reasonable, since looking ahead is extremely difficult. The performance of this strategy is to be analyzed (see section 3.3).

A different, more reasonable way to allocate objects to participants, who have private values for objects as well as for sets of objects, is the topic of the next section.

2.4.1 The combinatorial allocation problem

Parkes (2001) extensively discusses the *combinatorial allocation problem*. The problem is to allocate a set of items to a set of self-interested agents that have different private values for the items. The private values for bundles of items are non-linear. The goal is to allocate the items among the agents in a way that maximizes the total value over all agents.

Formally, let $v_k: 2^n \to \mathbb{R}_+$ be the private valuation function of agent $k \in \{1..K\}$ with respect to the subsets of the set of n items, where K is the number of agents. The function defines preferences of an agent over different outcomes. It is assumed that the value function is non-negative, $v_k(S) \geq 0$ for all bundles of items, $S \subseteq$

[17] An interesting application of finding an optimal strategy for an agent in a sequential auction for resources can be found at Boutilier et al. (1999)

$\{1,2,\dots,n\}$; an empty bundle has a value of 0, $v_k(\emptyset) = 0$; and monotone, that is, $v_k(S) \leq v_k(S')$ for all $S \subset S'$ meaning that items have non negative marginal values.

A feasible solution to the problem defines an allocation $(S_1, S_{2,} \dots \dots \dots S_K)$ of items to agents so that agent k receives one bundle, S_k. Each item is allocated to no more than one agent. The goal is to maximize the total value of bundles among all agents.

Parkes mentions various optimization problems that can be modeled as combinatorial allocation problems, among them are the time-slot allocation problem (Rassenti et al. (1982)) and the distributed vehicle routing problem (Sandholm (1993)). We believe a combinatorial auction can be used to solve the problem stated in this work. An implementation of this auction is developed in Chapter 4.

Let us now review the way the combinatorial allocation problem can be solved. The classical mechanism by which the problem is solved is named Vickrey-Clarke-Groves (VCG). This mechanism makes all agents to have the same dominant strategy – truthfully reveling their preferences over the different outcomes. Importantly, applying the VCG mechanism does not involve game theoretic considerations.

The VCG mechanism is a sealed-bid, single shot, combinatorial auction. Each agent is required to provide the auctioneer (a central allocator) with his private value function $v_k(\cdot)$, i.e., a vector of length $2^n - 1$ values, a value for each possible bundle of items. The latter solves the combinatorial allocation problem, and allocates bundles to agents while maximizing the total value of the bids. Then, each agent is charged with a payment. The way the payments are defined ensures that truthful revelation of preferences is a dominant strategy.

The GVA (Generalized Vickrey Auction) is a variation of the VCG where the payment of agent k is equal to the change of the total value among all other agents $\{1..K\}\backslash k$ caused by his participant in the auction. The utility that agent k gains from participating in the auction is assumed to be quasi-linear, and is equal to the total value gained by him minus the price he paid.

We denote the optimal solution when all K agents participate in the auction by $\left(S_1, S_{2,} \dots \dots \dots S_K\right)$ and the optimal solution when agent k does not participate in the auction by $(\overline{S_1}, \overline{S_2}, \dots \overline{S_{k-1}}, \overline{S_{k+1}}, \dots \overline{S_K})$. Then, agent k's payment is:

$$(v_1(\overline{S_1}) + v_2(\overline{S_2}) +.. v_{k-1}(\overline{S_{k-1}}) + v_{k+1}(\overline{S_{k+1}}) +.. v_K(\overline{S_K})) -$$

$$(v_1(S_1) + v_2(S_2) +.. v_{k-1}(S_{k-1}) + v_{k+1}(S_{k+1}) +.. v_K(S_K))$$

Parkes believes that applying the VCG mechanism as part of the solution method for decentralized problems can yield efficient solutions, in spite of the fact that its computational complexity is quite considerable. Each agent is required to calculate an exponential number of private values of bundles. Additionally, the winner determination problem, which determines which agents are awarded with which bundle, is NP-Hard (see Sandholm (2002)). This method is even more intricate when calculating the private values themselves is NP-hard, as with the case for routing problems.

Ausubel and Milgrom (2006) note several other weaknesses of the VCG mechanism. Among them are: low revenues for the seller; creation of coalitions of losing bidders and the possibility for a single bidder to falsely present himself as several competing bidders and offer multiple bids with the intent to decrease his payment. The authors demonstrate these weaknesses by analyzing a simple combinatorial auction in which two items are to be allocated among three agents when some items are worthless (have the value 0) for some agents. The authors prove that only in this case, these weaknesses are relevant. The authors also mention privacy problems as a shortcoming of the VCG mechanism. These problems can occur when public resources are sold in exchange for a price that is considered by the public low. The authors also note that when the value of items is common, i.e., not private, truthful revelation of preferences is not necessarily an optimal strategy for the agents.

As noted, the combinatorial auction has been implemented and solved by using the GVA mechanism in Chapter 4 while assuming that no coalitions can be formed. The other weaknesses seem to be irrelevant in our case. However, the computational load and therefore running times of the auction are relatively large. Therefore, there is a room for alternative mechanisms that can cope more efficiently with the combinatorial allocation problem. Parkes (2001) suggests an iterative combinatorial auction named i-bundle.

i-bundle is an iterative combinatorial auction that obtains an optimal solution to the combinatorial allocation problem when agents are assumed to follow myopic strategies. In each iteration of i-bundle, the agents are allowed to bid on bundles of items, while considering other agents' bids, as a response to the auctioneer who raised the minimum prices. Then, a tentative allocation of items is calculated by solving a winner determination problem. The auction ends when all agents receive a bundle or when all agents do not change their bids in 2 consecutive rounds. i-bundle solves the combinatorial allocation problem to optimum. That is proved by applying a duality theorem.

i-bundle may be superior to the GVA for two reasons. First, the total amount of bids that are calculated is significantly smaller compared with the GVA, in which bids for all possible bundles are calculated in advance. Second, the winner determination problem is typically smaller. As a result, the running time of i-bundle is significantly smaller than the running time of the GVA. However, i-bundle does not ensure truthful bidding by agents.

We now move forward to presenting the implementation of single shot combinatorial auctions in the area of freight transportation procurement.

2.4.2 Combinatorial auctions in freight transportation procurement

A combinatorial auction seems to be an effective method of solving decentralized problems. Auctions of this sort are typically applied in the area of freight transportation, where the auction is reversed.

Caplice and Sheffi (2006) discuss such auctions. The auctioneer or *shipper* (typically a large retailer) has to buy items (freight transportation services) from a large number of potential suppliers or *carriers* (transportation providers). Generally, the auction is a single shot, first price auction that is conducted by a software company or a third party consultant that carries out the auction on the auctioneer's behalf.

A shipper conducts an auction every two years (in average) and the bidding process usually takes several months. The main focus in the auction is in determining winning carriers (the auctioneer's problem) rather than in the optimization problems of each carrier.

Indeed, several (large) commercial companies have held complex combinatorial auctions over the recent decades and reported the saving of millions of dollars. See for example Porter et al. (2002). However, these auctions are generally first-price auctions, and as such do not ensure truthful bidding. Although recent studies suggest the implementation of the GVA mechanism to the procurement of transportation services (see Haung and Xu (2013) and Haung and Xu (2014)), this mechanism has never been applied in real life to the procurement of transportation services, due to its complexity and the desire of carriers not to reveal private information.

However, we do find the GVA mechanism appropriate for the solution of the problem studied in this thesis.

2.5 Literature Gap addressed in this study

We apply the GVA mechanism for the solution of the decentralized allocation problem of service tasks while using clustering as a means for decomposition of the problem as to reduce the GVA's computational complexity. To the best of our knowledge, this is the first study to suggest the use of the GVA mechanism for allocation problems similar to the one stated in this thesis. That is, the first study to suggest the use of a strategy-proof mechanism for these allocation problems and by doing so cancel out game-theory considerations of the agents.

Indeed, there is some risk at a decomposition of the problem since bidders may be encouraged to bid untruthfully or speculate (see Rothkopf (2007)). However, we assume this risk is negligible, due to the exponential computational complexity and little common knowledge.

Additionally, less computationally demanding allocation methods, such as a sequential Vickrey auction and sequential negotiations are implemented and examined.

The framework of a 2-stage allocation mechanism, which is commonly used to solve distributed transportation problems, is accepted in this work. Therefore, after the initial allocation process ends, contractors are allowed to exchange tasks between them as to decrease their total cost.

3 Problem Definition

In this chapter, the decentralized field service routing problem is defined. We describe elements of the problem (independent acting agents), define the research goals and develop features of a desired solution. Performance measures are then presented and a default status is characterized.

3.1 Problem settings

The Decentralized Field Service Routing Problem (DFSRP) can be stated as follows: A set of N service tasks should be carried out by a company. Each task is characterized by its location and service time. The company provides this service by outsourcing the tasks to K contractors. In addition, each of the contractors is pre-committed to additional service tasks, originated from different companies. The details of the contractors' tasks are private information of each one of the contractors and cannot be revealed to the other agents. Similarly, before the tasks of the company are allocated to contractors their details is private information of the company.

We denote the set of tasks owned by the company by I^C and the sets of the pre-committed tasks by I_k for $k = 1, \dots, K$. Each contractor provides a single field service team which is initially located at a given point. That is, the contractors are characterized by the location of their depots and by the location and duration of their pre-committed tasks.

The goal of the company is to outsource all its tasks at the minimal possible total payment for her, while the goal of each contractor is to maximize the payments obtained from the company net of the variable costs that he incurs. These costs consist of the regular time and overtime fee to his field service team and of the traveling cost of the vehicle.

The traveling cost (of a service team's vehicle) between a pair of locations i and j, is denoted by c_{ij} and the traveling time by t_{ij}. The regular time hourly salary to the field service team is denoted by f_1. In overtime, the hourly salary increases by f_2. The planning horizon is a single working day that consists of up to L regular working hours and possibly up to $M - L$ overtime hours.

3.2 Research Goals

While there is a clear conflict of interests between the company and the contractors, it is not a zero sum game. Efficient allocation of tasks to contractors may benefit all the parties. Moreover, since the business relations between the company and the contractors are of a long term nature, an allocation mechanism can be agreed upon in advance and the parties are not likely to abandon the agreement as long as their expected long-run benefits are greater than the conceivable alternatives.

The goal of this study is to devise an automatic allocation mechanism that satisfies the following requirements

1. The mechanism results in an allocation of the tasks to contractors such that the total operational cost for them will be as close as possible to that of the allocation that could be constructed by a hypothetic omniscient central planner whose objective is to minimize this cost while serving all the required tasks.
2. The mechanism does not require the contractors to reveal their private information regarding the pre-committed tasks.
3. The total sum of payments paid by the company to the contractors is small enough to motivate the outsourcing of the tasks. In other words, the company is not able to perform the tasks by its own fleet at a lower expense.
4. The reward obtained by each individual contractor is large enough to motivate him to accept the agreement. That is, the net profit of the contractor from executing the tasks allocated to him is large enough.
5. Implementation of the mechanism is computationally tractable.

3.3 Performance measures

In order to evaluate the proposed allocation mechanism, we define several performance measures that reflect the attractiveness of the method from different points of view.

From the company's point of view, a good solution would allow the sum of the payments the contractors are paid to be lower than the cost of serving all tasks by itself. Therefore, the first performance measure is the difference between the total payments paid by the company to the contractors and the cost of serving its tasks by its own employees and vehicles. We refer to this measure as *profitability of outsourcing.*

When calculating the cost of providing the services by the company without outsourcing we assume that the company employs appropriate personnel and maintains its own fleet of vehicles. In addition, we assume that the cost structure of the company is similar to that of the contractors except for the fact that the service teams of the company depart from a single location (the company depot) and hence need regularly to spend more time and cover longer distances in order to arrive at the location of the tasks.

From the contractor's point of view, a good solution would allow the additional costs incurred by him to be lower than the rewards received from the company. That is, his marginal profit from the cooperation with the company is positive.

As noted, a solution is defined by the payments paid by the company and by the allocation of the tasks to the contractors. Therefore, the next step would be evaluating the quality of the allocation.

From the society's point of view, a good solution is one that allocates the tasks among the contractors such that their total operational cost is minimized. The quality of the allocation from this perspective can be evaluated by comparing it to an optimal allocation identified by a hypothetic central planner who considers the pre-committed tasks and the other operational constraints. The gap between these two solutions is referred to as *the price of anarchy*[18]. Clearly, the lower price of anarchy, the better the decentralized solution is.

In real life, the companies may outsource their service tasks heuristically, without taking the contractors' pre-committed tasks into account. As a result, the generated solutions may be inefficient, and the price of anarchy may be very high.

Another way to measure the social benefit of an allocation mechanism is to compare its outcome to the outcome achieved in a default situation where the agents do not cooperate. We define the default as such a situation where the company serves all its tasks by its own and the contractors are engaged with their pre-committed tasks exclusively, again assuming that all parties admit a similar cost structure. The *value of cooperation* is the net increase in the contractors' profits as a result of the collaboration plus the net decrease in the company's costs, both with respect to the default situation. Consider the example illustrated in Table 1 where a 900$ value of cooperation is derived from a 400$ decrease in the company net costs and 500$ increase in the contractors profits.

[18] This term was first used by Koutsoupias and Papadimitriou (1999) and was defined as the ratio of the worst-case objective function value of a Nash equilibrium of a game and that of an optimal outcome. The term quantifies the inefficiency of selfish behavior in an n-agent game, and seems appropriate to describe the loss of social value caused by a multi-agent with self-interests.

Table 1: The value of Cooperation

Default Situation		
Company	**Contractors**	**System**
Total cost for the company without outsourcing	Total cost for the contractors without outsourcing	Total cost for the system
4400	4000	4400+4000=8400
After applying our mechanism		
Total cost for the company - outsourcing (sum of prizes/payments)	Total cost for the contractors - outsourcing	Total cost for the system
3900	7500	7500
Value of Cooperation		
Decrease in costs for the company	Profit for the contractors	Value of cooperation
4400-3900=500	3900-(7500-4000)=400	500+400=900 or 8400-7500=900

The following figure demonstrates the relation between the price of anarchy and the value of cooperation

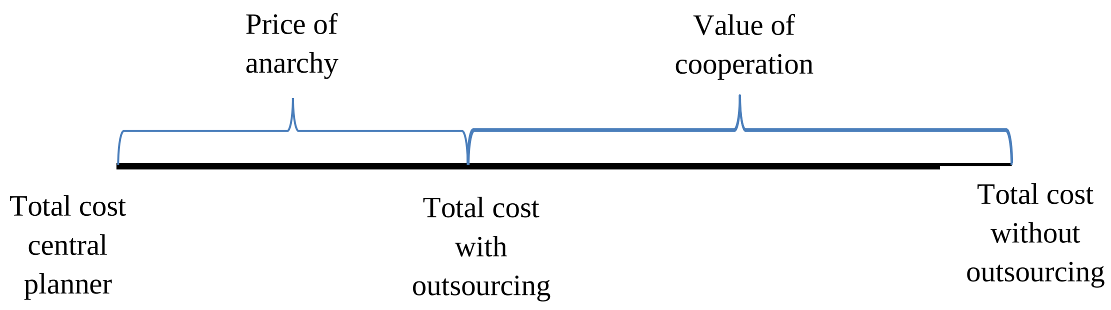
Price of anarchy
Value of cooperation
Total cost central planner
Total cost with outsourcing
Total cost without outsourcing

Figure 1: Price of Anarchy and value of cooperation

A summary of the performance measures is given in Table 2.

Table 2: Summary of performance measures

Measure	Performance measure	Compared with
Profitability of outsourcing (Company Perspective)	Sum of rewards	Total cost for the company without outsourcing
Profitability of outsourcing (contractors' perspective)		Additional costs
Value of cooperation (Social Perspective)	Total cost for the contractors	Total cost for the system without outsourcing
Cost of Anarchy (Social Perspective)		Optimal total cost (under complete information and centralized decision maker

In order to calculate the above measures, one needs to solve the optimization problems faced by the agents when no outsourcing is practiced, as well as the problem of the central planner. While these two problems are not in the focus of this study, we formulate and solve them for the sake of evaluating and benchmarking the proposed allocation methods. The problems are closely related but not identical to the field service and routing problems discussed in the literature. In the next two sections we formulate these problems as mixed integer programs. In the numerical experiments reported in Chapter 5 we solve these programs using a state of the art solver.

3.3.1 The problem of the central planner

As noted, an optimal allocation of company's tasks to contractors is the allocation that minimizes the total cost of all contractors while meeting certain constraints. In particular, in a feasible solution each pre-committed task should be served by the contractor that owns it and each of the company's tasks should be served by one of the contractors. This allocation is found by solving the following model.

Notations

$k \in \{1..K\}$ – depot of contractor k

I_k – set of pre-committed tasks of contractor k

I – set of all locations in the centralized problem $I = \{1..K\} \cup I^C \cup \bigcup_{k \in \{1..,K\}} I_k$

Decision Variables

$$x_{ijk} = \begin{cases} 1 & if\ contractor\ k\ travels\ from\ i\ to\ j \\ 0 & otherwise \end{cases}$$

u_{jk} – arrival time of contractor k at location j

T_k – total working time of contractor k

E_k – overtime of contractor k

Model

$$Min\left\{\sum_{\substack{i,j\in I\\k\in\{1..K\}}} c_{ij}x_{ijk} + f_1\sum_{k\in\{1..K\}} T_k + f_2\sum_{k\in\{1..K\}} E_k\right\} \qquad (1)$$

$s.t.$

$$\sum_{i\in I} x_{kik} = 1 \qquad \forall k\in\{1..K\} \qquad (2)$$

$$\sum_{i\in I} x_{ikk} = 1 \qquad \forall k\in\{1..K\} \qquad (3)$$

$$\sum_{\substack{j\in I\\k\in\{1..K\}}} x_{ijk} = 1 \qquad \forall i\in I\backslash\{1..K\} \qquad (4)$$

$$\sum_{j\in I} x_{ijk} = 1 \qquad \forall i\in I_k, \forall k\in\{1..K\} \qquad (5)$$

$$\sum_{j\in I} x_{ijk} = \sum_{j\in I} x_{jik} \qquad \forall i\in I, \forall k\in\{1..K\} \qquad (6)$$

$$T_k = \sum_{i,j\in I} x_{ijk}s_i + \sum_{i,j\in I} x_{ijk}t_{ij} \qquad \forall i,j\in I, \forall k\in\{1..K\} \qquad (7)$$

$$u_{jk} \geq u_{ik} + t_{ij} + s_i - M(1 - x_{ijk}) \qquad \forall i\in I\ , \forall j\in I\backslash\{0\}, \qquad \forall k\in\{1..K\} \qquad (8)$$

$$E_k \geq u_{ik} + s_i + t_{ik} - L \qquad \forall i\in I\backslash\{0\},\ \forall k\in\{1..K\} \qquad (9)$$

$$E_k \leq M - L \qquad \forall k\in\{1..K\} \qquad (10)$$

$$\sum_{\substack{h\in\{1..K\}\\h\neq k\\i\in I}} x_{ikh} = 0 \qquad \forall k\in\{1..K\} \qquad (11)$$

$$x_{ijk}\in\{0,1\} \qquad \forall i,j\in I \quad , \qquad \forall k\in\{1..K\} \qquad (12)$$

$$E_k \geq 0 \qquad \forall k\in\{1..K\} \qquad (13)$$

$$u_{jk} \geq 0 \qquad \forall j\in I\backslash\{0\}, \qquad \forall k\in\{1..K\} \qquad (14)$$

This model aims at minimizing the total cost (traveling, overtime and regular time) of all contractors, while making sure that contractor k begins and ends his route at depot (k) (see constraints (2) and (3)). Moreover, the model assures that all tasks are served by the contractors (constraint (4)) and that the set of pre-commited tasks of contractor k, I_k, is served solely by him (constraint (5)). Constraint (6) is a flow conservation equality. Constraint (7) calculates the total working time for the k'th contractor, by taking into account both the traveling time and the service time. Constraint (8) relates the variable u_{jk} that keeps track on the arrival time of the contractor at each task to the routing variable x_{ijk} and is used to eliminate sub-tours, i.e., cycles that do not pass through depots. Constraint (9) assigns correct value of overtime to each contractor and (10) limits the total overtime of contractor k. Constraint (11) prevents a contractor from traveling to another contractor's depot. Lastly, Constraints (12), (13) and (14) define the decision variables.

Constraint (8) is based on a well-known sub-tours elimination technique in a variety of vehicle routing problems originally introduced by Miller, Tucker and Zemlin (1960) in the context of the traveling salesman problem. Its advantage over other alternatives is that it eliminates the exponential number of sub-tours using a small set of constraints. Hence, it can be easily implemented in a commercial solver and modeling language. However, the LP relaxation of the formulation based on this technique is known to yield weak lower bounds and hence these formulations are typically hard to solve to optimality. In fact, state of the art solvers typically fail to deliver optimal solutions for these programs even for instances with several dozens of nodes. Near optimal solutions with optimality gaps of a few percentages can be achieved for larger instances and such solutions are often adequate for practical purposes. Note that we are interested in the optimal solution of an omniscient central planner as a reference point to what is achieved by the proposed allocation mechanism. For this reference to be valid, it makes no sense to use a sub-optimal solution.

However, a super-optimal solution may be beneficial. A solution of this sort can be obtained by solving the model without considering constraints (8) and (14), i.e., without arrival time variables u_{jk}.

This requires modifying constraint (9) to

$$E_k \geq T_k - L \qquad \forall k \in \{1..K\} \qquad (9)$$

The generated solution is obviously a lower bound for the optimal solution, and can be considered to be equal to the optimal solution if close enough. Iterative sub-tour elimination enables us to approach this kind of solutions.

The pseudo-code of the iterative sub-tour elimination process (limited by maximal time allowed) is given below:

```
1. Solve (1-13) without constraint (8)
2. Find the inner circles that are part of the solution
3. While inner circles are part of the solution and time limit hasn't been exceeded
   3.1. Insert constraints forbidding the inner circles to the model
   3.2. Solve the new model
   3.3. Find new inner circles in the solution
4. Return the total cost for the final solution
```

An inner circle is defined as a path that begins and ends at location $i \in I^C$ and does not contain any contractors' depots.

Let C be a set of nodes that constitute a sub-tour in the obtained solution. The corresponding forbidding constraint is:

$$\sum_{k \in \{1..K\}} \sum_{\substack{i,j \in C \\ i \neq j}} x_{ijk} \leq |C| - 1 \qquad (15)$$

The solution is optimal when it does not contain inner circles.

3.3.2 Total cost for the company without outsourcing

Recall that in the default situation the company serves the tasks by itself. The service is done using a fleet of vehicles and service teams that exit from and arrive to the company's depot. The company is then faced with the problem of determining the routes of its vehicles covering the tasks so as to minimize the total operational cost of the vehicles and the total working cost of the service teams. A formulation of this problem as a mixed integer programming model can be derived from the formulation of the central planner's problem applying the following adjustments.

1. All vehicles exit from and arrive to the company's depot, denoted by node 0.
2. No pre-committed tasks exist in the company's problem. Therefore, constraint (5) is unnecessary. Additionally, I, the set of all locations in the problem is just $I = I^C \cup \{0\}$

An optimal solution to this problem cannot be found using a state of the art solver. Therefore, we apply the sub-tour elimination method to obtain a near super-optimal solution for this problem as well.

3.3.3 Total cost for a single contractor

The formulation of the contractor's problem is a particular case of the central planner's problem, where only a single vehicle exists and all tasks to be considered are served by him. Therefore, constraints (5) and (11) are unnecessary. All the other constraints are adjusted to consider a single vehicle.

Note that this problem is in fact an instance of the TSP. Hence, the problem is much easier than the central planner's problem and therefore, its optimal solution can be found quite quickly using a state of the art solver.

4 The Proposed Mechanism

A two-stage solution mechanism is developed and implemented in order to cope with the decentralized problem formulated in the previous chapter. At the first stage, a feasible allocation of all the company's tasks to the contractors is created. At the second stage, the contractors are allowed to exchange tasks among themselves. The exchange procedure may reduce the total operational cost of the contractors, and improves the solution obtained at the first stage.

We consider two variations of the first stage mechanism that allocates the tasks to the contractors:

a. Sequential Sealed-bid auctions: second price auctions and combinatorial auctions.
b. Sequential negotiation protocol between the company and the contractors.

The second stage procedure is studied in two different settings, either with or without cash transfers between the contractors.

In section 4.1 the variations of the Stage A allocation mechanism are presented and Section 4.2 is dedicated to the exchange protocol of Stage B.

4.1 Stage A – Allocation of tasks to contractors

The variations of the allocation mechanism are discussed here. The sealed-bid auction mechanism is presented in 4.1.1 and the Sequential negotiation protocol in 4.1.2

The scheme of the sealed-bid auctions is as follows: a task (or a cluster of tasks) is communicated to all contractors at once by the company. Each contractor forms his bid (or bids). Then, the company decides which contractor should serve which task and rewards the contractors accordingly. The company's decision is based on minimizing the total added cost of the contractors required to serve all the offered tasks. Payments to contractors are paid as to ensure that bidding truthfully (bidding the real added cost) is a **dominant strategy** of the contractors. Here, we consider two types of sealed-bid auctions – the second price Vickrey auction and the combinatorial auction.

The Sequential negotiation procedure is designed in the following manner: First the company sets a payment (a "prize") for each task. The tasks and their attached payments are than offered to the contractors one by one according to some arbitrary order (of the contractors). Each contractor, on his turn, may either accept each of the tasks with its offered payment or reject it. Once all the contractors are offered, the payments for the remaining unserved tasks are increased and the process is repeated until all the tasks are allocated to contractors.

The sequential sealed-bid auctions and the Sequential negotiation differ in several ways. The first one is the number of times a task is communicated to contractors. While during a sequential negotiation, a task can be announced more than once (not only to different contractors but also in different iterations of the procedure), in a sealed-bid auction a task is announced once (one-shot auction). The second major difference is the way contractors interact with the company. While in sealed-bid auctions, all contractors bid simultaneously, only one contractor commits on a task in a single iteration of the negotiation procedure. The third difference is the amount of information communicated. While in the sequential negotiation the contractors communicate only their willingness to commit to a subset of the tasks, the sealed-bid auctions require providing an exact bid.

4.1.1 Sealed-bid auctions

A full combinatorial auction appears to be an ideal implementation of the sealed bid auctions to allocating the company's tasks to the contractors. In this scenario, all company's tasks are announced to the contractors at once and each contractor estimates the cost of serving each sub-set of the tasks. These estimations are communicated to the company which applies the GVA mechanism (Vickrey (1961), Clarke (1971) and Groves (1973)) and determines the winner of each task and payments of the winners. The payments are calculated in a way which ensures truthful bidding. Using this setting, all tasks are optimally allocated at the same time, and the total cost for the system is minimal.

Unfortunately, the computational complexity of this mechanism is exponential. Indeed, in order to calculate the cost of serving all the subsets of N offered tasks each contractor has to solve $2^N - 1$ NP-hard optimization problems. Clearly, this is impossible for any reasonable large N. Therefore, applying the combinatorial auction with a large number of tasks is not practical. How can the complexity of the combinatorial auction be reduced to make this mechanism applicable for our problem?

A reduction of the complexity can be achieved by clustering the company's tasks into subsets to be offered to contractors sequentially. The auction determines the winners of each task of the subset and the payments. The size of each subset should be small enough to allow solving the hard optimization problems required to calculate the additional cost that each contractor incurs by serving the subset and all of its subsets in reasonable time.

We consider several possible sizes of task's subsets:

a. $n = 1$. In this case, the combinatorial auction reduces to a simple second-price (Vickrey) auction.
b. $1 \leq n \leq n_{max}$, where n_{max} is the maximal subset size allowed.

For a contractor already committed for a set of tasks $\hat{I}$ the marginal cost of serving a set of additional tasks I' is calculated as the difference between the contractor's costs for serving $\hat{I} \cup I'$ and serving $\hat{I}$ alone. This calculation can be carried out by solving the MILP model (1)-(14). Since in the proposed bidding mechanism bidding truthfully is a dominant strategy this cost represents the bid for I'. When no feasible solution for $\hat{I} \cup I'$ is found, the bid is equal to ∞.

Sequential vickrey auctions
For the Vickrey auction ($n = 1$) the process is as follows: The company announces its tasks one at a time to the contractors according to some arbitrary order. For each task, the contractors calculate their bid as described above and submit it to the company. Thereafter, the company allocates the task to the lowest bidder. His payment is the second lowest bid. The process ends once all tasks have been allocated to contractors.

The order in which tasks are announced may affect the quality of the obtained allocation. In order to increase the social welfare we propose to announce the tasks starting with the geographically most isolated tasks and proceeding, at each iteration, to the nearest neighbor of the previously announced one. A pseudo-code of this process is given below:

```
1. Create an empty list called ORDER
2. Add the task for which the distance to its nearest neighbor is largest to the list ORDER.
3. While not all tasks are in ORDER
    3.1. Add to ORDER the nearest neighbor of the last member in ORDER
```

The winning contractor's net profit (payment paid by the company net of added cost) is non-negative. Note that for the analysis of the outcome of this mechanism we assume that the contractors have no knowledge regarding the tasks before they are announced and apply no speculative consideration regarding them.

Sequential Vickrey auctions with small clusters
The proposed procedure is as follows: company's tasks are grouped in several clusters. Each cluster is announced, in turn, to the contractors. Each contractor constructs and communicates one bid that is his added cost when serving **all** tasks in the cluster. Once all bids are received, the company allocates all cluster's tasks to the lowest bidder and pays him the second lowest bid.

According to this procedure, winning contractors are allocated with all tasks in a cluster. Therefore, and in order to allow them to win more than one cluster, n_{max}, the maximal number of tasks in cluster, should be small.

A protocol for this process is listed below

```
1. Group the company's tasks in small clusters
2. Form the order by which the clusters are announced
3. While there are still clusters that were not announced to contractors
    3.1. Company: announce the next cluster to contractors
    3.2. Contractors: calculate one bid for all cluster's tasks and communicate it to company
    3.3. Company: award the lowest bidder with all cluster's tasks
    3.4. Company: pay the lowest bidder the second lowest bid
```

Bids are constructed based upon the additional cost that each contractor incurs if serving all cluster's tasks. Methods for cluster's formation and determining the order of announcement are presented in the next section on sequential combinatorial auctions.

Sequential combinatorial auctions

Next, we present the sequential combinatorial auctions. The auction is administrated as follows: at first, tasks are grouped together in several **clusters.** Each cluster, in turn, is announced to the contractors. The contractors compute the appropriate bid for all subsets of the cluster, meaning that for a cluster containing n tasks and an auction containing m participating contractors, $m(2^n - 1)$ bids are calculated. A bid of a contractor for some subset is equal to the added cost incurred by him should he serve this subset. The bids are communicated to the company. Then, the company implements the GVA mechanism and determines the winner of each task in the cluster. The company also rewards each contractor with his corresponding payment. A protocol of the process is listed below.

```
1. Group the company's tasks into clusters
2. Form the order by which the clusters are announced
3. While there are still clusters that were not announced to contractors
   3.1. Company: announce the next cluster to contractors
   3.2. Contractors: for all subsets of the cluster: calculate the bid and communicate it to company
   3.3. Company: determine the contractors who will serve the cluster's tasks by applying the GVA
        mechanism and notify the contractors
   3.4. Company: Calculate the payment each contractor is entitled to and award him with that
        payment
```

Order of announcement is constructed by applying the following method: first, the center of mass for each cluster is found, assuming that all tasks within a certain cluster are weighted equally. Then, we apply the procedure presented for the Vickrey auction when these centers are considered as tasks. The underlying assumption is that offering nearby clusters one after another results in a better allocation of tasks and hence increases the social welfare.

The main key stones of this process are: Clustering (done by the company once); forming bids (done, for all clusters, by contractors); Winner-determination of each task (done, for all clusters, by the company); Payment Calculation (done by the company, for all clusters). We now describe these key stones in details.

Clustering

The clustering procedure initially creates N clusters. Each cluster contains one of the company's tasks. As this procedure proceeds, tasks located closely to one another are grouped to the same cluster, and that causes some clusters to be eliminated. The number of clusters should be selected such that the size of the largest one (n_{max}) is sufficiently small.

A pseudo-code of the process is given below. This procedure was inspired by the work of Kruskal (1956):

```
1. Create $N$ different clusters, each containing one company task. Mark them by $(1..N)$.
2. While the number of clusters is greater than $K$
   2.1. Find the minimal distance in the distance matrix. Save the row $(i)$ and the column $(j)$
   2.2. If $i$ and $j$are not in the same cluster then
        2.2.1. Count the number of tasks that the cluster containing $i$ contains. Mark it by $n_i$
        2.2.2. Count the number of tasks that the cluster containing $j$ contains. Mark it by $n_j$
        2.2.3. If $n_i + n_j \leq n_{max}$ then
               2.2.3.1. group all tasks that the cluster containing $i$ contains with the tasks contained by
                        The cluster containing $j$.
   2.3. Set the distance between $(i)$ and $(j)$ to $\infty$
```

Once all the company's tasks are grouped in K clusters, the sequential auctions can begin.

Forming bids – sequential auctions

Given a cluster offered by the company each of the contractors calculates the additional cost incurred by serving each possible non-empty subset of tasks in the cluster. This calculation can be done by solving MILP (1)-(14) with the proper input.

Winner Determination

Once all bids for a cluster are received, the company is able to apply the GVA mechanism and determine the contractor who serves each cluster's tasks. This is done with the intent of minimizing the overall additional cost for the contractors by solving an optimization problem.

The formulation of the optimization problem is as follows:

Notations

$\mathcal{C}$ - the set of tasks in the cluster. $P(\mathcal{C})$ is the collection of all subsets of this set (the power set).

$p_{k,S}$ - the cost of the offer of contractor k for subset S.

Recall that this value is obtained from each contractor for each subset of the cluster $\mathcal{C}$

Decision variables

$$x_{k,S} = \begin{cases} 1 & \textit{the contractor } k \textit{ serves subset } S \\ 0 & \textit{otherwise} \end{cases}$$

Model

$$Z^*_{\{1,..,K\}} = \sum_{\substack{S\in P(\mathcal{C}) \\ k\in\{1..k\}}} p_{k,S}x_{k,S} \tag{16}$$

$s.t.$

$$\sum_{S\in P(\mathcal{C})} x_{k,S} \leq 1 \qquad \forall k \in \{1..K\} \tag{17}$$

$$\sum_{\substack{S\in P(\mathcal{C}):i\in C \\ k\in\{1..K\}}} x_{k,S} = 1 \qquad \forall i \in \mathcal{C} \tag{18}$$

$$x_{k,S} \in \{0,1\} \qquad \forall k \in \{1..K\}, \quad S \in P(\mathcal{C}) \tag{19}$$

This model aims at minimizing the total additional total cost to contractors (16) while all tasks are to be served once (constraint (18)) and no contractor is allowed to serve more than one sub-set (constraint (17)) since the auction is combinatorial. A contractor who serves subset S serves all tasks grouped in it.

Payment calculation

Once all cluster's tasks are allocated to contractors, the company is required to reward the contractors with their corresponding payment. According to the GVA mechanism, a contractor should be rewarded with the amount of money he saves for others when participating in the auction.

For contractor k, this amount is equal to the difference between the total cost for all other contractors ($\{1..K\}\backslash k$) when contractor k does not participate in the auction and the total cost for all other contractors when contractor k participates in the auction. The latter cost is the value of the solution of the original Winner-Determination problem (16)-(19) net of the added cost for contractor k. The former cost is found by solving another instance of the Winner-Determination model for all the contractors excluding k. We denote the values of the optimal solutions of these problems by $Z^*_{\{1,..,K\}\backslash\{k\}}$.

Therefore, the winner-determination problem is solved $K + 1$ times. It is solved once with all contractors and K additional times without each of the contractors.

Formally, the payment that contractor k obtains from the company denoted by P_k, is given by

$$P_k = Z^*_{\{1..K\}\setminus k} - \left(Z^*_{\{1,\dots,K\}} - \sum_{S \in P(C)} p_{k,S} x^*_{k,S} \right) \quad (20)$$

where $x^*_{k,S}$ is the optimal solution of the original winner determination problem for all the contractors. Note that $Z^*_{\{1,\dots,K\}} - \sum_{S \in P(C)} p_{k,S} x^*_{k,S}$ is the total cost of all the contractors excluding k when all the contractors are included.

Let us show that the net profit of contractor k is non-negative.

$$P_k = Z^*_{\{1..K\}\setminus k} - \left(Z^*_{\{1,\dots,K\}} - \sum_{S \in P(C)} p_{k,S} x^*_{k,S} \right) = Z^*_{\{1..K\}\setminus k} - Z^*_{\{1,\dots,K\}} + \sum_{S \in P(C)} p_{k,S} x^*_{k,S}$$

Since $Z^*_{\{1,\dots,K\}}$ is the optimal cost when all contractors participate in the auction, $Z^*_{\{1,\dots,K\}} \le Z^*_{\{1..K\}\setminus k}$ and therefore $Z^*_{\{1..K\}\setminus k} - Z^*_{\{1,\dots,K\}} \ge 0$. Thus, $Z^*_{\{1..K\}\setminus k} - Z^*_{\{1,\dots,K\}} + \sum_{S \in P(C)} p_{k,S} x^*_{k,S} \ge \sum_{S \in P(C)} p_{k,S} x^*_{k,S}$. The latter phrase is exactly the additional cost to contractor k. Hence, the payment is larger than the additional cost and the net profit is positive.

Note that payment of contractor k is given in return to serving all tasks in subset S.

4.1.2 Sequential negotiation

In the Sequential negotiation scheme, a prize for serving each task is initially set by the company. Then, all tasks are offered, simultaneously, to the contractors (one contractor at a time) according to some arbitrary order. That is, each negotiation session is held (automatically) between the company and a single contractor. Each contractor commits for a subset (possibly empty) of the tasks offered as to maximize his net profit (prizes net of operational costs). If the negotiations with all contractors end and some tasks are yet to be allocated, new, higher, prizes are set and the process is repeated until all tasks are allocated. We assume that no speculative considerations are applied by contractors. That is, contractors never reject a profitable offer in anticipation for a more profitable one. In practice, while this assumption may not hold, the speculative power of each contractor with respect to a task is limited by his beliefs about the cost of serving the task by the other contractors.

We identify three factors that influence the dynamics of the negotiation: the order by which the contractors are negotiated; the initial prizes; the prize incrementing method.

Contractors can be negotiated according to a fixed order (round robin). However, this is unfair since the contractors that are negotiated first are favored over those who are negotiated last.

Another option would be fixing a certain negotiation order for the first iteration, and then reverse this order repeatedly until all tasks are allocated. However, such ordering implies that the last contractor in each iteration is negotiated immediately at the beginning of the next iteration. Thus, assuming that he does not speculatively postponed his offer to the later iteration, when the prizes are higher, does not make sense.

Therefore, it seems that random order generated in each iteration is the preferred option – applying it does not favor one contractor over the other (assuming the process is done on a regular basis) and no contractor can be sure that applying speculative considerations will result in higher prizes for him.

The initial prize for a task is determined by an increasing function of the service time of the task, $r_i = g(s_i)$. A policy that dictate lower initial prizes are likely to result in lower profits for the contractors, more efficient allocation and more iterations of negotiations. In practice, the function g, and hence the prizes, are determined largely by the balance of power between the company and the contractors. In our numeric experiments we consider two levels of initial prizes in order to demonstrate these effects.

In order to maintain the correlation between the service time of a task and the prize of that task, the prize is multiplied by a constant after each iteration. Thus, the prizes grow exponentially with the number of iterations.

A protocol of the negotiation process is listed below

```
1. Set $I^o$ to be the group of all company's tasks: $I^o = I^C$.
2. Company: sets initial prizes for the company's tasks $r_i = g(s_i)$    $\forall i \in I^o$
4. While $I^o \neq \emptyset$ do
  4.1 Company: Creates a random $K$-size permutation. This is the order by which the contractors are
      negotiated in the first iteration.
  4.2. For each contractor $k$ (in the permutation)
      4.2.1. Company: offers $I^o$ to the next contractor
      4.2.2. Contractor $k$: solves an optimization problem to find $I^s$
      4.2.2. Contractor $k$: commits for a sub-set $I^s$ of $I^o$. $I^s \subseteq I^o$
      4.2.3. Contractor $k$: updates $\widehat{I_k} = \widehat{I_k} \cup I^s$
      4.2.4. Company: awards contractor with prizes $\{r_i | i \in I^s\}$
      4.2.5. Company: updates $I^o = I^o \backslash I^s$
      4.2.6. If $I_0 = \emptyset$ then stop
  4.3. If $I^o \neq \emptyset$ then company: update $r_i = a \cdot r_i$    $\forall i \in I^o$
```

a represents the increase rate of the prizes and $\widehat{I_k}$ is the set of tasks that contractor k is committed to. In order to find I^s, the set of company's tasks to commit on in the current iteration, the contractor solves an optimization problem aimed at maximizing his net profit. The formulation of the problem (a prize collecting problem) is given below.

The prize collecting problem of the contractor

Notations

$I = \widehat{I_k} \cup I^o \cup \{k\}$ – set of all locations in the problem.

Decision variables

$$x_{ij} = \begin{cases} 1 & if\ contractor\ k\ travels\ from\ i\ to\ j \\ 0 & otherwise \end{cases}$$

u_j – arrival time of contractor at location j

T– total working time of contractor

E– Duration of overtime

Model

$$Max\left\{\sum_{i,j\in I^c} r_i x_{ij} - \sum_{i,j\in I} c_{ij}x_{ij} - Tf_1 - Ef_2\right\} \tag{21}$$

$s.t.$

$$\sum_{i\in I} x_{ki} = 1 \tag{22}$$

$$\sum_{i\in I} x_{ik} = 1 \tag{23}$$

$$\sum_{j\in I} x_{ij} \le 1 \qquad \forall i \in I^o \tag{24}$$

$$\sum_{j\in I} x_{ij} = 1 \qquad \forall i \in \widehat{I_k} \tag{25}$$

$$\sum_{j\in I} x_{ij} = \sum_{j\in I} x_{ji} \qquad \forall i \in I \tag{26}$$

$$T = \sum_{i,j\in I} x_{ij}s_i + \sum_{i,j\in I} x_{ij}t_{ij} \tag{27}$$

$$u_j \ge u_i + t_{ij} + s_i - M\left(1 - x_{ij}\right) \qquad \forall i \in I \qquad \forall j \in I\backslash\{k\} \tag{28}$$

$$E \ge u_i + s_i + t_{ik} - L \qquad i \in I\backslash\{k\} \tag{29}$$

$$E \le M - L \tag{30}$$

$$\boldsymbol{x_{ij}} \in \{0,1\} \tag{31}$$

$$E \ge 0 \tag{32}$$

$$u_j \ge 0 \tag{33}$$

The model aims at maximizing the sum of prizes net of costs (regular time, over time and traveling). Constraints (22)-(23) ensure that the contractor's route begins and ends at his depot. Constraint (24) allows the contractor to not serve some offered tasks and constraint (25) obliges him to serve his committed tasks. Constraint (26) ensures flow conservation. Constraint (27) calculates the total working time. Constraint (28) relates the variable u_j that keeps track on the arrival time of the contractor at each task to the routing variable x_{ij}. Constraints (29)-(30) limit and calculate the overtime. The three last constraints define the decision variables. The set I^s is defined as

$$I^s = \left\{i \middle| \exists j, x_{ij} = 1,\ \ i \notin \widehat{I_k}\right\} \tag{34}$$

4.2 Exchanging tasks between the contractors

Stage A ends when all the company's tasks are allocated to the contractors. The goal of Stage B is decreasing the total operational cost for the contractors by allowing them to exchange company's tasks among themselves. An *Exchange* of task i with task j is defined from the point of view of the contractor that serves task i, and it means removing task i from his route and inserting task j into his route instead. The payments received in Stage A from the company are not exchanged.

This framework considers the exchange of one task with another. More general versions would consider the exchange of sets of tasks for a different (possibly empty) set.

We offer the following protocol for stage B:

- The company communicates the location and duration of its service tasks to all contractors.
- Each contractor calculates the profit caused by an exchange of task i (that he serves) with task j (that he does not serve). This change is noted by a_{ij}.
- a_{ij} is communicated to a central controller (3rd party agent).
- The central controller finds a set of exchanges that maximizes the reduction in the total cost of the contractors.
- The exchanges are carried out and possibly money is transferred between the controller and the contractors.
- The process is repeated until no set of exchanges that reduces the total cost for all the contractors exists.

The resolution of the communicated information by the contractors, i.e., a_{ij}, is likely to affect the performance of the protocol. Two resolution levels are considered in this framework:

1. complete information with money transfers – the size and the sign (positive/negative) of the cost change
2. Partial information without money transfers – the sign (positive/negative) of the cost change only.

Considering the first level (complete information) requires adapting the proposed protocol. For this case, exchange of tasks also includes **money transfers** between the contractors done through the mediation of a central controller. We later describe the exact method by which the amount of transferred money is set.

4.2.1 Calculation of the change in the operational cost

Each contractor is required to calculate the change of cost that an exchange of task i (that he serves) with task j (that another contractor serves) inflicts. For this end, the contractor calculates its cost in the current status (with i in his route and without j in the route) and also its cost in the alternative status (without i his route and with j in the route) by solving a minimum cost optimization problem (1)-(14) presented earlier for the two options. Subtracting the latter from the former gives us a_{ij}. For the case of complete information, the change of cost is communicated to the central controller (size and sign). If an exchange is infeasible, than $a_{ij} = -\infty$ is communicated.

When partial information is transferred, only the sign is communicated to the central controller. When an exchange decreases the cost of the contractor a positive sign is communicated. A negative sign is communicated when the exchange increases the cost. Since the size of change is not communicated, the central controller has no option but to address all desirable exchanges in the same manner. Therefore, for these exchanges $a_{ij} = 1$ is considered. Similar considerations lead the central controller to forbid undesirable exchanges completely. Therefore, for these exchanges $a_{ij} = -\infty$.

We later show that in the case of complete information, an exchange that increases some contractor's cost is not forbidden since it can reduce the total cost for all contractors. This contractor will be compensated by a money transfer.

4.2.2 Finding best sets of feasible exchanges

After a_{ij} is defined for all $i, j \in I^C$ the central controller finds the best set of feasible exchanges. As defined earlier from the point of view of the contractor serving task i, an exchange of task i with task j means removing task i from the contractor's route and inserting task j into his route. An exchange of this sort cannot be done without a complementary exchange (or exchanges) by which some contractor receives task i into his route and some contractor removes task j from his route. The anticipated complementary exchange is the reverse exchange. The contractor who serves task j exchanges it with task i.

Nonetheless, this is not the only complementary exchange. Consider the following example: contractor 1 wishes to exchange task i with task j. Contractor 2 wishes to exchange task j with task m and contractor 3 wishes to exchange task m with task i. None of the desirable exchanges has an anticipated complementary exchange. However, a solution to satisfy all contractors is clear: contractor 1 removes i; contractor 2 removes j; contractor 3 removes m; contractor 1 receives j; contractor 2 receives m; contractor 3 receives i. It is possible to include more than two contractors (tasks) in a set of exchanges so long as all the tasks that are removed by the contractors are also received by the contractors. This example also illustrates that a set of feasible exchanges constitutes a circle in a directed graph.

The graph's nodes are the company's tasks. An arc (i, j) from node i to node j exists if tasks i and j are served by different contractors. Its weight is the profit that the contractor who serves j can enjoy if he removes j from his route and inserts task i instead, i.e. a_{ji}. The existence of a circle in this graph indicates the fact that all tasks in the circle (represented by nodes) can be removed and received by contractors.

Therefore, finding best sets of feasible exchanges is equivalent to maximizing the total weight of disjoint circles in a directed graph. The mathematical model of this optimization problem is presented below.

Notations

$\widetilde{I_k}$ – set of company's tasks allocated to contractor k

$I^C = \cup_k \widetilde{I_k}$ - set of all company's tasks

For simplicity of the model we assume $a_{ij} = 0$ for all pairs of tasks i, j that are initially allocated to the same contractor.

Decision Variables

$$y_{ij} = \begin{cases} 1 & if\ task\ i\ is\ exchanged\ with\ task\ j \\ 0 & otherwise \end{cases}$$

Model

$$Max \left\{ \sum_{i,j \in I^C} a_{ij} y_{ij} \right\} \tag{35}$$

$s.t.$

$$\sum_{j \in I^C} y_{ij} = \sum_{j \in I^C} y_{ji} \qquad \forall i \in I^C \tag{36}$$

$$\sum_{j \in I^C} y_{ij} \leq 1 \qquad \forall i \in I^C \tag{37}$$

$$\sum_{\substack{i \in \widetilde{I_k} \\ j \in I^C}} y_{ij} \leq 1 \qquad k \in \{1..K\} \tag{38}$$

$$y_{ij} \in \{0,1\} \tag{39}$$

This model aims at maximizing the sum of weights of the circles, while ensuring that all the removed tasks are received by other contractors (constraint (36)), ensuring that each task is to be exchanged at most once (constraint (37)) and that each contractor is allowed to exchange one task, at most (constraint (38)). This is important since a_{ij} values are not correct if a contractor exchanges more than one task in each iteration.

In the case of partial information where $a_{ij} \in \{1, -\infty\}$ solving (35) means simply maximizing the number of performed exchanges. All exchanges that are chosen reduce costs for the corresponding contractors.

In the case of complete information solving this problem does in fact maximizes the overall profit for the system. However, this does not necessarily mean that all exchanges chosen reduce costs for each contractor. Consider the example for a_{ij} given in Table 3.

Table 3: Example instance for a_{ij}

	1	2	3
1		-2	$-\infty$
2	$-\infty$		8
3	8	3	

The data in the table can be represented by the graph given in figure 2.

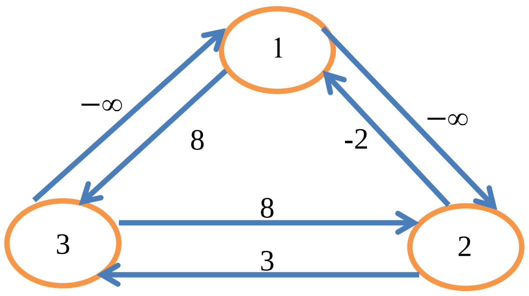

Figure 2: Auxiliary graph for example

5 possible circles can be found in the graph. They are presented in figure 3a – 3e respectively.

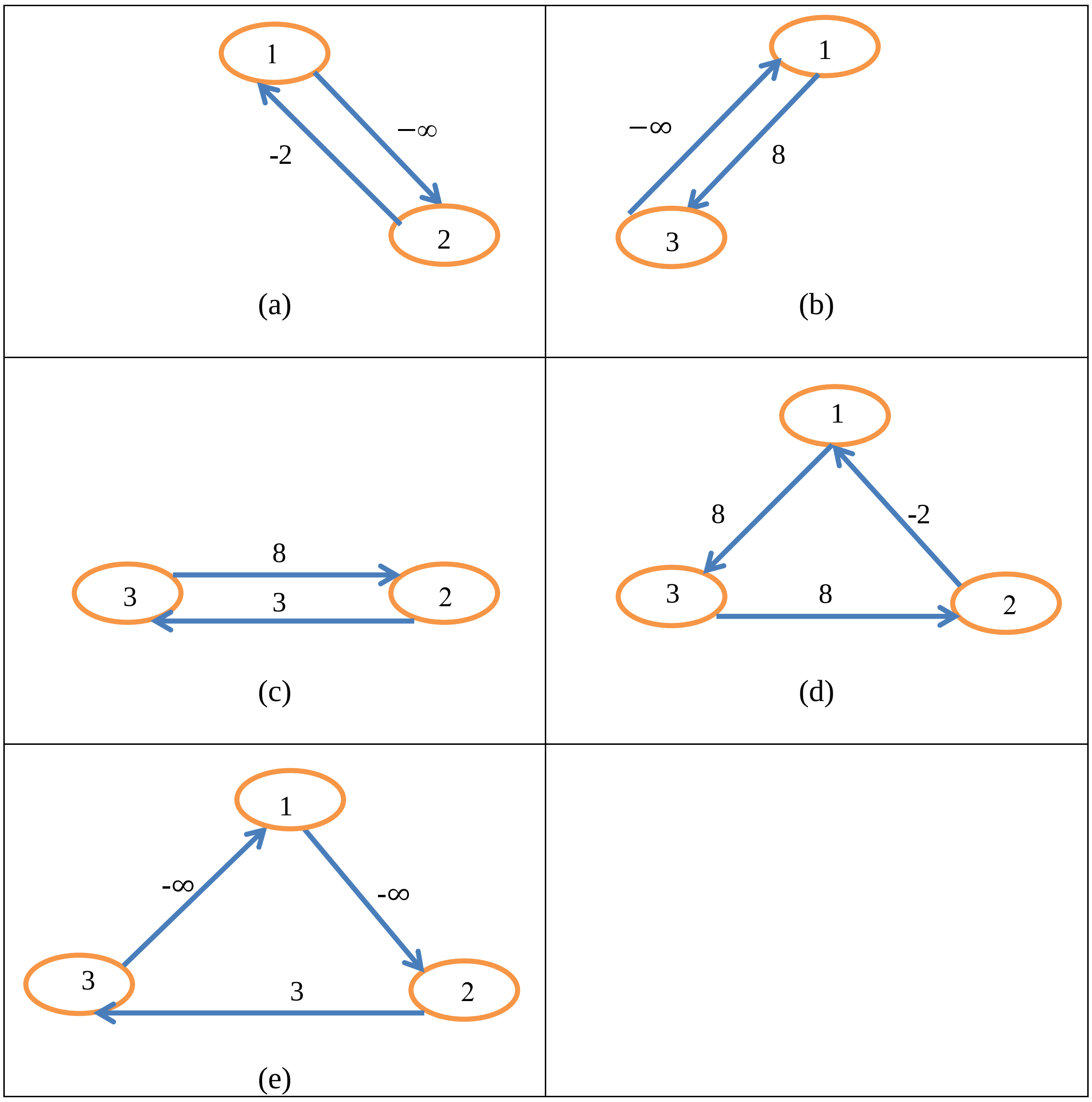

Figure 3a – 3e: Possible circles found in graph

In the example instance given above, maximizing the overall profit for the system results in applying exchanges according to circle (d) since the overall weight of circle (d) is 14, which is highest. However, applying the exchanges according to (d) means that the contractor who serves task 1 exchanges it with task (2). This **increases** his cost by 2 units. In order to make this arrangement acceptable by this contractor money transfers are needed to be applied.

4.2.3 Money transfers

We have shown that in the case of complete information, a set of exchanges that best reduces the overall cost for contractors can make the situation of some contractors worse compared to their situation prior to the exchange. This can be resolved by money transfers. The idea is that contractors who benefit considerably from the chosen exchanges will subsidies contractors whose benefits from the exchanges are relatively small or negative. How is this possible and in what way should we determine the subsidy?

Let us consider that the system as a whole (all contractors combined) benefit from applying the best set of possible exchanges since the total profit is positive. This benefit can be distributed among the all contractors in a way that will ensure each of them will gain from the exchanges. Deciding how to distribute this benefit derives the corresponding subsidies for the contractors.

Let $\tilde{C}$ be a set of tasks that constitute a circle in the obtained solution and $\tilde{S}$ be the set of contractors that own the tasks in $\tilde{C}$ (one contractor owns one task). Implementing the exchanges implied by $\tilde{C}$ reduces the system's cost by $\sum_{i,j\in\tilde{C}} a_{ij}$. If contractor $k \in \tilde{S}$ exchanges task i with task j then his cost reduces by a_{ij}. Supposing that the desired profit of contractor k is π_k derives a corresponding subsidy that is $\pi_k - a_{ij}$. Clearly, if $\sum_k \pi_k = \sum_{i,j\in\tilde{C}} a_{ij}$ then the sum of subsidies is 0.

This means that no external insertion of money is required in order to perform the money transfers.

Since a_{ij}s are known, what is left to be determined is π_k – desired profit for contractor k. We suggest that the profit of each contractor will be equal. i.e.: $\pi_k = \pi = \frac{\sum_{i,j\in\tilde{C}} a_{ij}}{|\tilde{C}|}$. The corresponding subsidy for contractor is $\frac{\sum_{i,j\in\tilde{C}} a_{ij}}{|\tilde{C}|} - a_{ij}$.

For the example given above, the proposed distribution of profit and the corresponding subsidies are given in Table 4:

Table 4: Required money transfers in example

	Contractor who originally owns (1)	Contractor who originally owns (2)	Contractor who originally owns (3)
Saving in cost from the exchange	-2	8	8
Total saving for system	-2+8+8 = 14		
Profit for each contractor	$\frac{14}{3}$		
Money transfer (subsidy)	$\frac{14}{3} - (-2) = \frac{20}{3}$	$\frac{14}{3} - 8 = -\frac{10}{3}$	$\frac{14}{3} - 8 = -\frac{10}{3}$

The contractor who originally owns (2) transfers $\frac{10}{3}$ units of money. So does the contractor who owns (3). These units are transferred to the contractor who originally owns (1), who is therefore subsided by $\frac{20}{3}$ units of money.

Not only does this definition of π_k not discriminatory, it is also Nash's solution to a bargaining game for more than two players (1950).

4.2.4 Stage B – Discussion

Stage B is constructed in a way that ensures all contractors are better off in each of its iterations. This is true for the case of partial information, as well as for the case of complete information (by allowing money transfers). However, stage B does not ensure that truthful bidding is an optimal strategy for the contractors. Consider the following example given in figure 4:

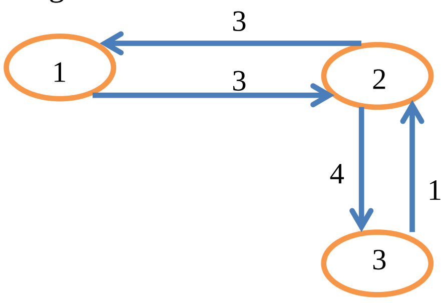

Figure 4: Example graph of truthful bidding

In this case, the optimal solution for the system is that the contractor who owns task (1) replaces it with task (2), whose owner receives (1). The total profit for the system is 6. Each of the two contractors gains 3 units of money from the exchange and no money transfer is performed hence each of the two contractors is better off by 3. The contractor who owns (3) is not a part of the solution and therefore gains nothing.

However, had the contractor who owned (3) known $a_{ij}$for all $i, j \in I^C$ of all other contractors, he could have communicated a false value for his own possible profits in such a way as to ensure to be a part of the solution and by that make that solution be sub-optimal for the system. Consider the altered graph displayed in figure 5.

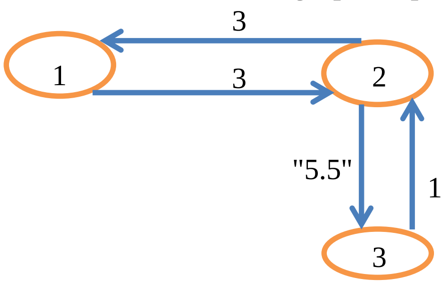

Figure 5: Example graph of untruthful bidding

If the contractor who owns (3) decides to communicate $a_{32} = 5.5$ then in the eyes of the controller the best set of exchanges is replacing (2) with (3) and the total saving is 1+5.5=6.5. The estimated profit each contractor will enjoy is $\frac{6.5}{2} = 3.25$. In order to ensure that profit, the contractor who owns (3) will have to pay $5.5 - 3.25 = 2.25$ units of money as subsidy to the contractor who owns 2. However, his net profit is still positive and is equal to 4-2.25 = 1.75.

This is of course better than gaining nothing in the case of truthful bidding. The system as a whole lost 6-(4+1) = 1 unit of money that could have been saved. An example for profitable untruthful bidding can also be shown for stage B with partial information, i.e., when only the sign of a_{ij} is communicated to the controllers.

Consider the following example given in figure 6:

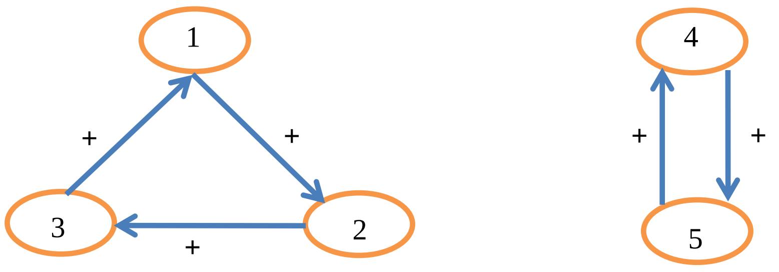

Figure 6: Example graph of truthful reporting

Suppose that contractor k_1 owns tasks (1) and (4), contractor k_2 owns tasks (2) and (5) and contractor k_3 owns task (3). Since all possible exchanges are profitable, reporting truthfully leads to executing the exchanges implied in the circle that contains the higher number of arcs, that is, the circle that contains 3 arcs. All contractors are better off.

Now, suppose that the exchanges implied in the 2-arc circle yield higher profit for contractor k_2. Had this contractor have complete information regarding the profitability of all exchanges (and thus, regarding all possible circles) he could have reported a false value for his profit from participating in the 3-arc circle, and thus preventing the feasibility of this circle. Figure 7 illustrates this report.

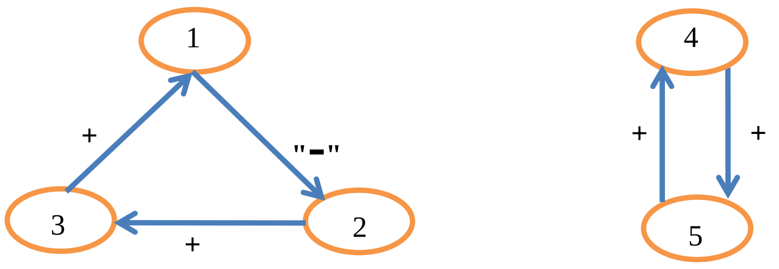

Figure 7: Example graph of untruthful reporting

By reporting untruthfully, contractor k_2 manipulated the controllers that choose the exchanges preferred by him, although they are the less profitable from the perspective of the society.

Indeed, this is a problem in the structure of stage B. However, we still propose to include it as an improvement heuristic of the solution generated by applying stage A. We present both practical and theoretical reasons for our recommendation.

First, in practice, in order for a contractor to successfully manipulate the system, he needs to have information regarding the a_{ij}s of the other contractors which is not likely. Without this information each contractor can secure a non-negative profit by truthfully report his $a_{ij}s$.

Second, follows from the proposition below,

Proposition: in both versions of stage B and disregarding the information available to the contractors the mechanisms can only improve the profit of each one of the contractors.

Proof: We will first show that under both mechanisms and disregarding the actions of the other contractors a contractor can secure himself a non-negative improvement in his profit by bidding truthfully. Thus a profit maximizer contractor will never submit a bid that will result in a loss.

For the version of stage B without money transfers, the claim is trivial. In this case only exchanges that are reported by the contractors as profitable to them are carried out. Thus at each iterations any truthful bidder can only improve his profit.

For the version of stage B with money transfers, consider a contractor that owns task i_1 but do not own task j_1 and let $\bar{a}_{i_1 j_1}$ be the true (possibly negative) profit of this contractor from trading i_1 for j_1. Now assuming he bids truthfully, that is $a_{i_1 j_1} = \bar{a}_{i_1 j_1}$ and that the exchange is carried out then he will receive the direct profit $a_{i_1 j_1}$ and the subsidy $\frac{\sum_{i,j\in\tilde{C}} a_{ij}}{|\tilde{C}|} - a_{i_1 j_1} = \frac{\sum_{i,j\in\tilde{C}} a_{ij}}{|\tilde{C}|} - \bar{a}_{i_1 j_1}$. That is in total $\frac{\sum_{i,j\in\tilde{C}} a_{ij}}{|\tilde{C}|}$. Now, clearly this is a positive value, because regardless of the truthfulness of the bids of the other contractors the circle $\tilde{C}$ is selected by the mechanism only if $\sum_{i,j\in\tilde{C}} a_{ij}$ is positive. This concludes the proof. ■

5 Numerical Experiments

We have conducted numerical experiments in order to evaluate the effectiveness of the mechanism developed in the previous chapter. The setting of the experiments includes problem instances and parameters of the proposed variations of the mechanism. The results of the experiments are presented and analyzed with respect to the performance measures defined in Chapter 3.

5.1 Problem Instances

An instance of the problem is characterized by locations and durations of all service tasks (those of the company and those pre-commited by the contractors) as well as by the locations of the company and contractors. Since the locations of the company and the contractors are not likely to change on a daily basis, these locations remain fixed for all generated instances of the problem. The locations of the service tasks, however, do change across the problem instances.

The number of tasks in each instance was set to 40, half of which are pre-committed tasks and the rest are company's tasks. The number of contractors was set to four, which seems quite reasonable, so the size of the generated instances is large enough to demonstrate the outcomes of the proposed mechanism. A realistic setting of the problem is likely to be larger. However, testing the mechanism with larger instances would have required developing and implementing stronger solution methods (exact or heuristic) for the underlying routing problem, which is out of the scope of this thesis. Recall that our focus here is to study the ability of the mechanism to induce efficient allocation of the task to the contractors. Therefore, the dimension of the test instances was selected such that the routing problem could be solved (or approximated) in a reasonable time using the methods presented in Chapter 3.

For each instance the locations of the tasks were randomly generated on a 100×100 square using a continuous uniform distribution $(x_i \sim U(0,100), y_i \sim U(0,100))$. The company's depot is located in the square's center, i.e. at point $(50,50)$. Next, the square has been divided into 4 quarters, and the contractors have been placed in the centers of the quarters, so that contractor 1 is placed in point $(25,25)$, contractor 2 in $(25,75)$, contractor 3 in $(75,75)$ and contractor 4 in $(75,25)$. The durations of service tasks (in minutes) are generated using the normal distribution with the expectancy of 30 and the standard deviation of 10 ($S_i \sim N(30, 10^2)$).

The length of a regular working day is set at 8 hours, i.e. $L = 480$ minutes, and M, maximal length of working day with overtime is set at 12 hours, i.e. $L = 720$ minutes. The cost of a regular working period is set at 200 cost units per hour (3.33 per minute) i.e. $f_1 = 3.33$. The cost of overtime (in addition to regular cost) is set at 100 money units per hour (1.66 per minute) i.e. $f_2 = 1.66$. The traveling cost and traveling time (in minutes) from location i to location j is equal to the Euclidian distance between i

and j, i.e. $c_{ij} = t_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. This definition implies that the cost of traveling is 1 unit of money per unit of distance (km for example) and that traveling speed is one. Clearly, this does not limit the generality of the results.

As mentioned earlier, 20 tasks out of 40 generated in each instance are company's tasks. These 20 tasks are to be chosen randomly in each instance. The other 20 tasks are set as pre-committed tasks of the contractors.

We have used a parametric probabilistic procedure that relates pre-committed tasks to the contractors with a probability that decreases with the traveling time. This is because practically, a task is most likely to be served by its nearest contractor. For pre-committed task i we defined the probability that the task belongs to contractor k as

$$\Pr(i \in I_k) = \frac{\left(\frac{1}{t_{ik}}\right)^{\alpha}}{\sum_k \left(\frac{1}{t_{ik}}\right)^{\alpha}} \quad (40)$$

where t_{ik} is the traveling time between the locations of task i and contractor k. Parameter α controls the clustering structure of the instance. The larger the value of α is, the higher is the probability of relating a task to its nearest neighbor. However, since we wish to create balanced instances we fixed the number of tasks to be related to each of the four contractors to five (exactly one quarter of the number of pre-committed tasks).

5.1.1 Generated instances

We generated 20 datasets of 40 service tasks locations and times. For each dataset the locations of the 20 company tasks were randomly selected. The rest of the tasks (pre-committed) were related to contractors using the mechanism described above once with $\alpha = 1$ and once with $\alpha = 3$. That is, we created 40 instances of the problem in total.

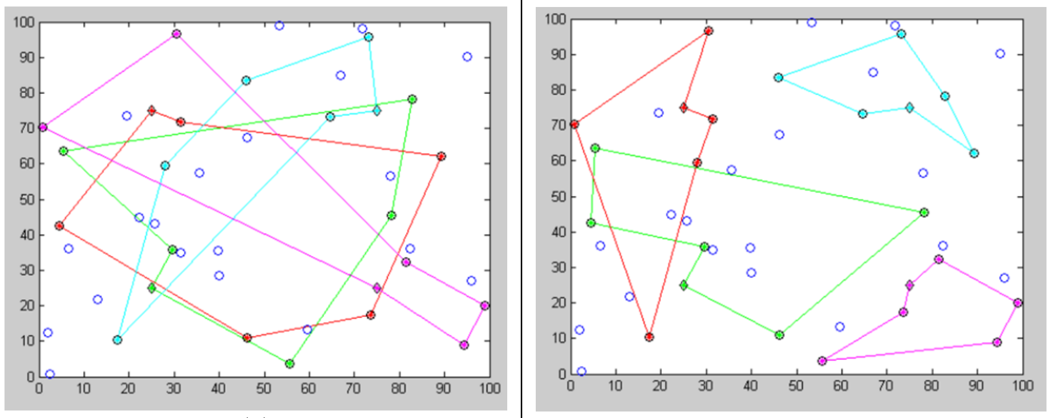
(a) (b)

Figure 8a – 8b: Optimal initial routes for $\alpha = 1$ compared to $\alpha = 3$

Figure 8 exemplifies two such instances with the initial routes of the contractors before committing to the company's tasks. Blue circles represent the company's tasks and black circles represent pre-committed tasks. Each diamond represents one contractor. The route and pre-committed locations of each contractor are marked in a different color.

Clearly, the allocation of pre-committed tasks to contractors for $\alpha = 1$ (Figure 6a) is considerably different from the allocation for $\alpha = 3$ (Figure 6b). While in the case of $\alpha = 3$ most of the pre-committed tasks are related to their nearest contractor (or at least – second nearest), for $\alpha = 1$ this relation seems almost random. Consequentially, the initial routes for $\alpha = 3$ are much shorter than the initial routes for $\alpha = 1$. Hence, the total initial cost of the contractors is expected to be significantly lower for $\alpha = 3$.

5.1.2 Total cost for heuristic allocation

In practice, companies that outsource their service tasks to several contractors typically allocate the tasks based on simple geographic considerations. A service zone is defined for each contractor and service calls are allocated based on this zoning. If a contractor is too loaded to handle tasks from his zone, the task is allocated to a contractor in a neighboring zone.

In order to benchmark our method, we imitate this heuristic allocation procedure as follows. The heuristic approach aims at allocating each task to its nearest contractor, assuming that his pre-committed tasks are also nearby. In order to balance the work load of each contractor, the number of company's tasks allocated to a single contractor is limited to $\left\lceil \frac{N}{K} \right\rceil$ tasks (5 in our case). Once all these tasks are allocated to a contractor he is removed from further consideration.

A pseudo-code of the heuristic method is presented below:

```
1. Calculate the distances between all company's tasks and all contractors' depots.
2. Do for all company's tasks
    2.1. Find task $i$ and contractor $k$ for whom the distance is minimal
    2.2. Allocate task $i$ to contractor $k$
    2.3. Set the distances between task $i$ and all contractors to $\infty$
    2.4. If contractor $k$ is allocated with $\left\lceil \frac{N}{K} \right\rceil$ company's tasks
        2.4.1. Set the distances between contractor $k$and all tasks to $\infty$
```

After applying this heuristic procedure, the shortest route is found by TSP and the total working time (service and routing) is calculated. The total cost of the heuristic allocation is the sum of the total costs of the contractors.

5.2 Solution parameters

In this section, we discuss and determine the value of the parameters of the suggested variations of the mechanism.

5.2.1 Mechanism parameters

First, we discuss the value of the parameters for the variations of Stage A.

For the combinatorial auction, n_{max} must be defined. As noted, the efficiency of the solution increases with n_{max}, however, the value of n_{max} is limited due to the complexity of the combinatorial auction and the need to conduct it in a reasonable time. We therefore selected $n_{max} = 8$. For Vickrey auction with small clusters the value of n_{max} was set to 4.

For the sequential negotiation mechanism, two possible values for $r_i = g(s_i)$, the initial prize for task i as a function of its duration in the sequential negotiation were considered:

1. $g(s_i) = 100\% \times f_1 \times s_i = 100\% \times \frac{10}{3} \times s_i$
2. $g(s_i) = 150\% \times f_1 \times s_i = 150\% \times \frac{10}{3} \times s_i$

Recall that s_i is the service time in minutes and f_1 is the cost of one minute of regular working day. For the two values of r_i, the increase rate of the prize was set to $a = 1.1$.

The sum of the prizes paid to the contractors is expected to be higher for the higher value of initial prizes. The total cost for the generated allocation is expected to be higher as well, since with higher initial prizes it is more likely that tasks will be allocated less efficiently.

Note that in the first case (paying exactly 100% of the working cost for each task i) r_i is the minimal prize for which a contractor that will be willing to serve task i may be found. This is the case only if the task is located exactly on the route of a contractor, who has not exploited all his regular time hours. Allocating the task to the contractor in such a case incurs neither additional traveling cost, nor overtime cost. Even in this case, the net profit from serving the task is zero. Therefore, it is reasonable from the company's point of view to begin the negotiation offering this prize.

Since the working relation between the company and the contractors is of a long term nature, any sustainable arrangement must be profitable for all the parties. By changing the value of the initial prizes it is possible to control the long term expected profit of the contractor and to make this arrangement appealing enough for the parties.

A summary of the chosen parameters for variations of Stage A is presented in Table 5.

Table 5: Parameters for variations of Stage A

Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiation		Heuristic
$n_{max} = 8$	$n_{max} = 4$	$r_i = g(s_i) = 100\% \times s_i \times \frac{10}{3}$ $a = 1.1$	$r_i = g(s_i) = 150\% \times s_i \times \frac{10}{3}$ $a = 1.1$	$\left\lceil \frac{N}{K} \right\rceil = \frac{20}{4} = 5$

5.2.2 Computational features

Experiments were carried out using a Intel® core ™ i7-2600 CPU @ 3.40Ghz processor and 16GB of RAM. The MILP models were solved using IBM CPLEX 12.5.1.

The TSP (presented in section 3.3.3) and the prize collecting problem of the contractor (presented in section 4.1.2) were solved directly using CPLEX with time limits of 10 seconds and 1800 seconds, respectively. For the TSP, time limit of 10 seconds was usually enough to obtain the optimal solution. The time limit for the prize collecting problem is considerably larger since it is more complicated, dealing with usually larger number of tasks. Furthermore, it contains a decision element – whether to carry out a proposed task, that does not exist in the contractor's optimization problem. Time limit of 1800 seconds was enough to obtain an optimal solution for most instances of the prize collecting problem.

The winner determination problem in the combinatorial auction and the problem of finding the best set of exchanges (see sections 4.1.1 and 4.2.2 respectively) were solved to optimality in less than one second.

The time limit for finding the lower bounds of both of the central planner problems and the company’s problem (see sections 3.3.1 and 3.3.2 respectively) was set to four hours. However since this is an iterative process and running time was checked only at the end of each iteration, the actual running time for the process was sometimes longer. The lower bounds obtained were typically tight.

5.3 Experimental procedure

As previously noted, 40 instances were generated. For each instance all variations of Stage A (Vickrey auction, combinatorial auction, Vickrey auction with small clusters, both variations of the sequential negotiation and heuristic allocation) were applied. The allocation obtained for each variation was kept. Next, the two variations of Stage B were applied for each allocation of Stage A.

Note that Stage B is considered also for the allocation generated heuristically. This is valuable since applying Stage B, performed between the contractors themselves, is possible even without applying collaborative mechanisms designed to generate the initial allocation.

5.4 Results

We now present the results of the experiments; first, for the default status (without outsourcing) and then for all treatments specified previously. The results are analyzed using the performance measures defined in Chapter 3.

5.4.1 Default status

The default status is defined by the total cost for the company without outsourcing and by the total cost for the contractors when they serve their pre-committed tasks only, when the latter depends on α. The total cost for the contractors for $\alpha = 1$ and for $\alpha = 3$, and for all instances are presented in Table 6.

Table 6: Total cost for the contractors without outsourcing

Instance number	$\alpha = 1$	$\alpha = 3$
1	5,072.45	4,928.08
2	5,792.12	4,849.90
3	4,565.54	4,541.21
4	6,075.61	4,842.06
5	6,147.64	4,703.01
6	4,651.40	4,844.56
7	4,852.56	4,219.34
8	5,241.73	4,413.67
9	5,006.22	4,090.52
10	5,193.90	4,644.92
11	5,177.30	4,287.46
12	5,668.04	5,572.77
13	5,158.39	5,333.96
14	5,984.35	4,903.86
15	5,006.41	4,623.18
16	5,908.19	4,759.78
17	5,248.12	4,621.00
18	6,322.44	5,402.11
19	5,867.85	5,516.29
20	5,278.79	4,776.51

The costs are considerably lower for $\alpha = 3$, as anticipated.

Running times and the obtained lower bounds for the total cost of the company without outsourcing are presented in Table 7.

Table 7: Lower bounds for the total cost for the company without outsourcing

Instance number	Value	Running time (seconds)	Comments
1	4,392.91	17,161	
2	3,887.04	12,560	Optimal value
3	4,000.97	9,184	Optimal value
4	3,753.33	6,946	Optimal value
5	4,311.26	15,579	
6	3,930.30	15,198	
7	3,708.15	15,036	
8	4,285.52	20,388	
9	3,988.73	14,756	
10	4,474.78	15,631	
11	3,609.71	442	Optimal value
12	4,031.77	15,836	
13	3,858.65	16,165	
14	3,866.91	11,824	Optimal value
15	3,400.73	49	Optimal value
16	4,255.02	14,778	
17	3,843.58	15,331	
18	3,752.76	1,425	Optimal value
19	3,700.60	16,422	
20	4,416.40	16,128	

Assuming these lower bounds are good estimations for the total costs for the company gives us total costs for the system (contractors and company) that are presented in Table 8.

Table 8: Total cost for the system in default status

		$\alpha = 1$		$\alpha = 3$	
Instance number	Total cost for the company without outsourcing	Total cost for the contractors	Total cost for the system	Total cost for the contractors	Total cost for the system
1	4,392.91	5,072.45	9,465.36	4,928.08	9,320.99
2	3,887.04	5,792.12	9,679.16	4,849.90	8,736.94
3	4,000.97	4,565.54	8,566.51	4,541.21	8,542.18
4	3,753.33	6,075.61	9,828.94	4,842.06	8,595.39
5	4,311.26	6,147.64	10,458.90	4,703.01	9,014.27
6	3,930.30	4,651.40	8,581.70	4,844.56	8,774.86
7	3,708.15	4,852.56	8,560.71	4,219.34	7,927.49
8	4,285.52	5,241.73	9,527.25	4,413.67	8,699.19
9	3,988.73	5,006.22	8,994.95	4,090.52	8,079.25
10	4,474.78	5,193.90	9,668.68	4,644.92	9,119.70
11	3,609.71	5,177.30	8,787.01	4,287.46	7,897.17
12	4,031.77	5,668.04	9,699.81	5,572.77	9,604.54
13	3,858.65	5,158.39	9,017.04	5,333.96	9,192.61
14	3,866.91	5,984.35	9,851.26	4,903.86	8,770.77
15	3,400.73	5,006.41	8,407.14	4,623.18	8,023.91
16	4,255.02	5,908.19	10,163.21	4,759.78	9,014.80
17	3,843.58	5,248.12	9,091.70	4,621.00	8,464.58
18	3,752.76	6,322.44	10,075.20	5,402.11	9,154.87
19	3,700.60	5,867.85	9,568.45	5,516.29	9,216.89
20	4,416.40	5,278.79	9,695.19	4,776.51	9,192.91

5.4.2 Mechanism results

A solution to the decentralized problem is defined by an allocation of the company's tasks to the contractors and by the sum of the prizes that the company pays the contractors.

Total cost for the contractors

An allocation can be evaluated by its total cost for the contractors. This cost needs to be compared to the cost of the optimal allocation that a central planner with complete information could have found. Lower bounds for these costs are presented in Table 9 for all instances.

Table 9: Lower bounds for optimal cost for the system with outsourcing

	$\alpha = 1$			$\alpha = 3$		
Instance number	Value	Running time (seconds)	Comments	Value	Running time (seconds)	Comments
1	7,779.69	14,566		7,824.62	15,232	
2	8,049.80	17,554		7,221.82	2,138	Optimal value
3	6,947.65	827	Optimal value	6,997.63	14,526	
4	8,446.17	15,948		7,380.88	317	Optimal value
5	8,779.09	15,634		7,742.64	379	Optimal value
6	7,092.55	15,001		7,275.30	14,714	
7	7,186.92	15,562		6,983.97	14,983	
8	8,144.45	15,130		7,472.00	702	Optimal value
9	7,498.11	14,465		6,747.02	5,819	Optimal value
10	8,258.01	14,545		7,714.24	8,403	Optimal value
11	7,360.57	15,218		6,498.73	1,137	Optimal value
12	8,615.89	20,191		8,382.35	30,089	
13	7,236.78	19,295		7,513.61	22,955	
14	8,444.39	16,224		7,508.69	15,087	
15	7,084.75	19,375		7,040.39	353	Optimal value
16	8,243.94	14,416		7,804.34	519	Optimal value
17	7,941.98	15,000		7,301.47	9,492	Optimal value
18	8,652.95	14,465		7,820.74	15,002	
19	8,217.27	15,143		7,839.24	15,097	
20	8,143.69	15,675		8,025.45	3,106	Optimal value

It appears that the lower bounds for $\alpha = 3$ are tighter and in many cases the model is solved to optimality. This is because the central allocation problem is simpler for $\alpha = 3$ where the locations of the pre-committed tasks of each contractor are more clustered.

The values in Table 9 are either optimal values for the central planner problem or even better (super optimal). Let us now examine the total cost generated by the suggested mechanism. We begin with presenting the total cost of the allocations generated by applying all variations of Stage A only, without applying Stage B.

Total cost for all variations of Stage A for $\alpha = 1$ are presented in Table 10. For each instance, the best result is highlighted. Optimal allocations are marked by *.

Table 10: Total cost for the system where $\alpha = 1$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)	Lower Bound
1	8,286.50	**8,236.02**	8,337.15	8,256.81	8,266.58	8,731.69	7,779.69
2	8,247.68	**8,233.19**	8,525.96	8,280.96	8,334.31	8,788.40	8,049.80
3	*** 6,947.65 ***	*** 6,947.65 ***	7,165.67	*** 6,947.65 ***	7,309.80	7,734.98	6,947.65
4	**8,754.40**	8,845.24	9,052.40	8,788.95	8,906.53	9,559.13	8,446.17
5	9,502.72	**9,398.03**	9,650.53	9,419.45	9,439.62	10,040.73	8,779.09
6	**7,135.19**	7,212.65	7,286.18	7,163.86	7,214.22	7,743.36	7,092.55
7	7,752.48	**7,632.09**	7,981.50	7,686.71	7,839.27	8,234.74	7,186.92
8	8,382.21	**8,301.97**	8,447.85	8,359.60	8,487.59	8,669.23	8,144.45
9	8,229.64	**8,073.02**	8,131.98	8,152.05	8,179.99	8,392.12	7,498.11
10	**8,375.01**	8,442.16	8,505.77	8,375.01	8,454.86	8,704.11	8,258.01
11	7,591.17	7,564.64	**7,480.08**	7,603.17	7,590.47	7,854.78	7,360.57
12	8,959.38	**8,845.67**	8,898.85	8,865.63	9,049.52	9,377.40	8,615.89
13	7,778.82	**7,645.04**	7,828.87	7,739.59	7,753.32	8,464.54	7,236.78
14	8,746.91	**8,698.67**	8,906.58	8,735.79	8,821.91	9,386.67	8,444.39
15	7,585.07	**7,429.68**	7,619.71	7,570.47	7,871.63	7,626.25	7,084.75
16	8,854.39	**8,849.32**	9,108.57	8,852.16	8,850.17	9,738.38	8,243.94
17	8,483.17	8,276.70	8,377.09	**8,147.31**	8,404.00	8,236.28	7,941.98
18	8,948.65	8,994.01	9,264.23	**8,945.85**	8,999.72	9,639.17	8,652.95
19	8,761.50	8,714.72	9,101.39	**8,704.65**	8,802.92	9,312.26	8,217.27
20	8,480.37	**8,400.98**	8,542.93	8,440.42	8,428.70	8,835.62	8,143.69

Total cost for all variations of Stage A for $\alpha = 3$ are presented in Table 11. The structure of this table is the same as Table 10 above.

Table 11: Total cost for the system where $\alpha = 3$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)	Lower Bound
1	8,071.56	**7,985.52**	8,309.70	8,025.02	8,027.67	8,401.74	7,824.62
2	*** 7,221.82 ***	*** 7,221.82 ***	7,437.61	7,250.86	7,287.23	7,912.04	7,221.82
3	7,089.41	7,179.43	7,525.25	7,158.86	**7,082.64**	7,781.13	6,997.63
4	7,404.30	*** 7,380.88 ***	7,539.89	7,404.30	7,465.77	8,027.10	7,380.88
5	**7,764.26**	7,821.84	7,941.08	7,796.30	7,894.23	8,189.30	7,742.64
6	7,447.29	**7,376.49**	7,535.43	7,550.65	7,592.08	7,646.72	7,275.30
7	7,232.81	**7,137.19**	7,380.34	7,171.33	7,239.81	7,814.13	6,983.97
8	7,510.27	7,563.11	**7,487.17**	7,595.83	7,914.25	7,896.91	7,472.00
9	**6,771.11**	6,780.42	6,833.99	6,792.17	6,890.50	7,026.31	6,747.02
10	7,792.16	7,821.31	8,051.45	7,897.59	8,071.67	**7,744.66**	7,714.24
11	**6,506.42**	6,542.59	6,933.40	6,538.46	6,773.85	6,972.02	6,498.73
12	8,594.27	**8,566.44**	8,762.16	8,607.65	8,714.79	9,001.12	8,382.35
13	7,847.28	**7,817.55**	8,163.41	7,845.06	7,823.25	8,526.62	7,513.61
14	**7,552.96**	7,571.87	7,846.32	7,576.37	7,598.47	7,847.23	7,508.69
15	*** 7,040.39 ***	*** 7,040.39 ***	7,177.01	7,079.17	7,616.42	7,136.32	7,040.39
16	7,852.01	**7,830.63**	7,929.45	7,837.26	8,315.46	8,093.60	7,804.34
17	7,357.13	7,353.42	7,492.69	**7,328.63**	7,381.37	7,380.08	7,301.47
18	8,133.10	**8,056.60**	8,254.56	8,070.49	8,244.09	8,095.92	7,820.74
19	8,281.50	8,216.80	8,339.41	**8,202.17**	8,218.23	8,560.44	7,839.24
20	8,201.28	**8,027.63**	8,416.53	8,076.99	8,255.83	8,178.86	8,025.45

The optimality gap of a solution with total cost TC is defined as: $100\% \times \left(1 - \frac{Lower\ Bound}{TC}\right)$. This is an upper bound on the *price of anarchy*. The price of anarchy for all variations of Stage A for $\alpha = 1$ is presented in Table 12. In the last two rows we present the sample average of the results and its standard deviation.

Table 12: Prices of anarchy where $\alpha = 1$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)
1	6.12%	5.54%	6.69%	5.78%	5.89%	10.90%
2	2.40%	2.23%	5.58%	2.79%	3.41%	8.40%
3	0.00%	0.00%	3.04%	0.00%	4.95%	10.18%
4	3.52%	4.51%	6.70%	3.90%	5.17%	11.64%
5	7.61%	6.59%	9.03%	6.80%	7.00%	12.57%
6	0.60%	1.67%	2.66%	1.00%	1.69%	8.40%
7	7.30%	5.83%	9.96%	6.50%	8.32%	12.72%
8	2.84%	1.90%	3.59%	2.57%	4.04%	6.05%
9	8.89%	7.12%	7.79%	8.02%	8.34%	10.65%
10	1.40%	2.18%	2.91%	1.40%	2.33%	5.13%
11	3.04%	2.70%	1.60%	3.19%	3.03%	6.29%
12	3.83%	2.60%	3.18%	2.82%	4.79%	8.12%
13	6.97%	5.34%	7.56%	6.50%	6.66%	14.50%
14	3.46%	2.92%	5.19%	3.34%	4.28%	10.04%
15	6.60%	4.64%	7.02%	6.42%	10.00%	7.10%
16	6.89%	6.84%	9.49%	6.87%	6.85%	15.35%
17	6.38%	4.04%	5.19%	2.52%	5.50%	3.57%
18	3.30%	3.79%	6.60%	3.27%	3.85%	10.23%
19	6.21%	5.71%	9.71%	5.60%	6.65%	11.76%
20	3.97%	3.06%	4.67%	3.52%	3.38%	7.83%
Average	**4.57%**	**3.96%**	**5.91%**	**4.14%**	**5.31%**	**9.57%**
Standard deviation	0.58%	0.45%	0.59%	0.52%	0.50%	0.71%

The price of anarchy for all variations of Stage A for $\alpha = 3$ is presented in Table 13. The structure of this table is the same as Table 12 above.

Table 13: Prices of Anarchy where $\alpha = 3$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)
1	3.06%	2.01%	5.84%	2.50%	2.53%	6.87%
2	0.00%	0.00%	2.90%	0.40%	0.90%	8.72%
3	1.29%	2.53%	7.01%	2.25%	1.20%	10.07%
4	0.32%	0.00%	2.11%	0.32%	1.14%	8.05%
5	0.28%	1.01%	2.50%	0.69%	1.92%	5.45%
6	2.31%	1.37%	3.45%	3.65%	4.17%	4.86%
7	3.44%	2.15%	5.37%	2.61%	3.53%	10.62%
8	0.51%	1.20%	0.20%	1.63%	5.59%	5.38%
9	0.36%	0.49%	1.27%	0.66%	2.08%	3.97%
10	1.00%	1.37%	4.19%	2.32%	4.43%	0.39%
11	0.12%	0.67%	6.27%	0.61%	4.06%	6.79%
12	2.47%	2.15%	4.33%	2.62%	3.81%	6.87%
13	4.25%	3.89%	7.96%	4.23%	3.96%	11.88%
14	0.59%	0.83%	4.30%	0.89%	1.18%	4.31%
15	0.00%	0.00%	1.90%	0.55%	7.56%	1.34%
16	0.61%	0.34%	1.58%	0.42%	6.15%	3.57%
17	0.76%	0.71%	2.55%	0.37%	1.08%	1.07%
18	3.84%	2.93%	5.26%	3.09%	5.14%	3.40%
19	5.34%	4.59%	6.00%	4.42%	4.61%	8.42%
20	2.14%	0.03%	4.65%	0.64%	2.79%	1.88%
Average	**1.63%**	**1.41%**	**3.98%**	**1.74%**	**3.39%**	**5.70%**
Standard deviation	0.37%	0.30%	0.48%	0.31%	0.44%	0.76%

The average optimality gaps for $\alpha = 3$ are lower than for $\alpha = 1$. Therefore, it seems that all variations of Stage A perform better for $\alpha = 3$. This can be an indication to a weakness of the mechanism for $\alpha = 1$. Another possible explanation for the higher optimality gap may be the fact that the lower bounds for the $\alpha = 1$ instances are looser. In our view the latter explanation is more likely because while many of the $\alpha = 3$ instances were solved to optimality there was only one optimal solution for the $\alpha = 1$ instances. All of the proposed mechanisms performed well in both cases. The average optimality gap ranges between 3.96% to 5.91% in the $\alpha = 1$instances and between 1.41% and 3.98% in the $\alpha = 3$ instances.

For both values of α, the sequential combinatorial auctions, sequential negotiations with 100% of tasks working time cost as initial prize and Vickrey auction produce good allocations compared to their alternatives.

The sequential combinatorial auctions divide the problem into small number of relatively large sub-problems ($n_{max} = 8$ while $n = 20$) and each sub-problem is solved exactly to optimality.

The procedure of the sequential negotiation announces all tasks to contractors (one contractor at a time) and the latter commits on a subset of the proposed tasks. Low initial prizes, combined with moderate increasing rate of the prizes, ensure that tasks are, in general, allocated efficiently.

An interesting finding is concerned with the Vickrey auction. A priori, we would anticipate it to result in worse allocations than does the combinatorial auction or the sequential negotiation, since according to the Vickrey auction, only one task per auction is allocated based on the task's marginal cost. So, what can be the reason for the Vickrey auction to perform well?

We try to find an answer to that question by examining the dynamics of the Vickrey auction in details. Table 14 summarizes the announced tasks, winning contractors, winning contractors' bids and second bid (winner's payment) for $\alpha = 1$ and for $\alpha = 3$ for the 5th instance (as an example). In the last two rows we present the sample average and median in the experiments.

Table 14: Dynamics of Vickrey auction for $\alpha = 1$ and for $\alpha = 3$ – 5th instance

	$\alpha = 1$				$\alpha = 3$			
Task announced	Winning contractor	Lowest bid	Second-lowest bid	$\frac{second\ bid}{first\ bid}$ (%)	Winning contractor	Lowest bid	Second-lowest bid	$\frac{second\ bid}{first\ bid}$ (%)
6	2	96.81	97.11	100.31%	4	107.8	112.31	104.18%
17	3	126.15	138.97	110.16%	1	143.6	171.53	119.45%
29	3	118	122.19	103.55%	1	128.95	157.38	122.05%
16	1	84.95	88.95	104.71%	1	80.71	88.95	110.21%
25	3	77	85.95	111.62%	2	77	116.76	151.64%
8	4	128.38	130.08	101.32%	2	130.08	175.07	134.59%
24	2	146.15	203.15	139.00%	2	141.95	218.01	153.58%
27	3	44.52	46	103.32%	2	30.02	92.6	308.46%
7	3	202.39	227.36	112.34%	2	179.26	346.52	193.31%
35	3	284.73	331.38	116.38%	2	276.44	488.56	176.73%
2	2	180.44	181.01	100.32%	1	203.97	245.82	120.52%
5	1	171.48	178.83	104.29%	3	194.80	254.57	130.68%
23	3	227.09	242.05	106.59%	3	152.62	436.14	285.77%
10	3	98.58	206.32	209.29%	3	70.67	458.7	649.07%
9	3	166.31	183.13	110.11%	3	131.94	344.45	261.07%
13	1	415.57	442.95	106.59%	3	253.11	619.3	244.68%
34	4	200.58	205.57	102.49%	1	256.53	263.36	102.66%
37	2	121.04	121.64	100.50%	4	103.98	163.65	157.39%
12	4	209.84	321.28	153.11%	4	137.12	386.7	282.02%
14	2	255.07	266.11	104.33%	2	260.7	298.65	114.56%
average		167.75	191.00	115.02%		153.06	271.95	196.13%
median		156.23	182.07	105.65%		139.53	250.19	152.61%

For $\alpha = 3$, considerable difference between the first and second bids exists. This indicates that the winning contractor's route is much closer to the allocated task than all the other routes. This is a property of the solution caused by the initial short routes of contractors serving their pre-committed tasks. Given this setting, it's reasonable that the contractor whose route is the nearest to a task (to whom the task is allocated by the Vickrey auction) is in fact the contractor to whom the task should be allocated when the allocation problem is solved centrally. This is evident if we examine the actual optimal routing of the 5th instance compared to the initial routes of the contractors (previously displayed for this instance). These are given in Figure 9a – 9b.

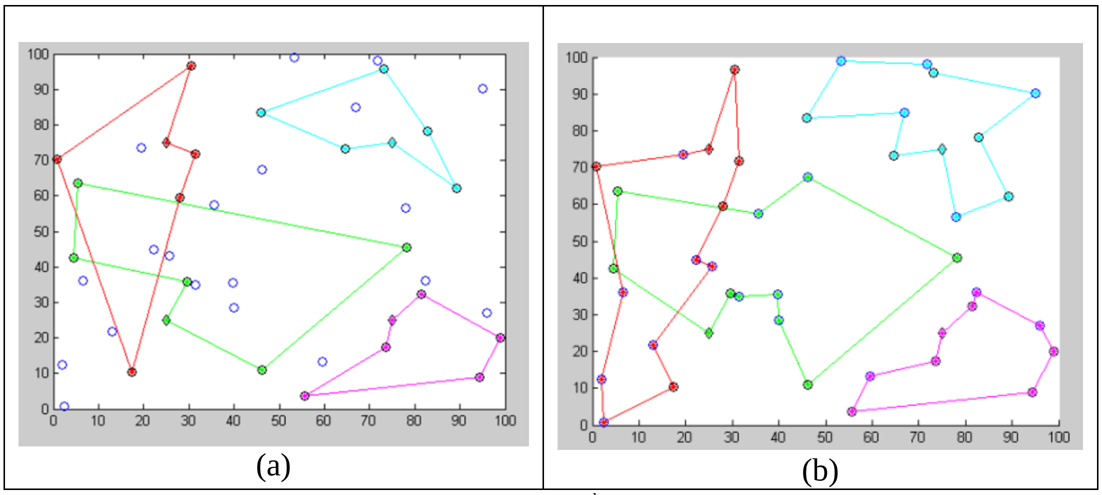

Figure 9a – 9b: Optimal routing for the 5th instance compared to the initial route

The order by which tasks are announced in the Vickrey auction is also helpful in achieving good allocations for $\alpha = 3$. The next announced task is the nearest to the task previously announced. In an optimal solution for $\alpha = 3$, tasks located at the same area are generally allocated to the same contractor. Announcing tasks from the same area one after another brings us closer to the optimal solution since it allows the contractor that has committed to a certain task to commit to a nearby task and not, for example, to a faraway task. Hence, an allocation similar to the optimal one is generated.

For $\alpha = 1$, the differences between the first bids and second bids are very small (for half of the tasks the difference is below 5%). This indicates that contractors' routes are long and intersect one another. In this case, one might expect that considering only one task at a time may be far from the optimal solution. However, based on the summary given above, it seems that even if a certain task was allocated to a contractor that is not the one serving it in the optimal solution, the cost does not increase considerably. This is the result of the similarity between the contractors' routes. Here as well, we note that announcing tasks one after another enables a contractor to commit on a set of nearby tasks (one by one) and by doing so to create a good solution (even if it is somewhat different from the optimal one).

As anticipated, if the initial prizes in the sequential negotiation are high, tasks are allocated to contractors that are not the contractors that serve them in the optimal solution. This is evident both for $\alpha = 1$ and for $\alpha = 3$ when comparing between optimality gaps of the two versions of the sequential negotiation protocol.

The Vickrey auction with small clusters performs poorer than other options. This is the case since a contractor commits on a set of tasks in each iteration. This commitment lowers his ability to commit to the following tasks, even if these tasks are much closer to his original route. Hence, the generated allocation is relatively far from optimal.

Naturally, the heuristic allocation that does not directly consider the pre-committed tasks of the contractors yields worse allocations. Nonetheless, considerable differences in its performance for $\alpha = 1$ and $\alpha = 3$ are noticed. For $\alpha = 3$, the pre-committed tasks of each contractor are close to his locations. Allocating tasks to contractors on a basis of closeness to depot, as the heuristic does, is likely to maintain the routes short and the allocation close to optimal. This is clearly not the case for $\alpha = 1$, where pre-committed tasks are spread almost randomly and often distant from their contractors.

Value of cooperation

Let us now consider the value of cooperation for all variations of Stage A. The value of cooperation is equal to the savings when the system applies the suggested mechanism compared to the default status. If the default status' cost is equal to $TC(default)$ and the cost of the suggested allocation is TC then the value of cooperation is equal to $TC(default) \ - TC$. Note that this expression has little since if it is not normalized with the maximal value of cooperation. The maximal value of cooperation is equal to the difference between the default status' cost and the optimal cost of the central planner, $TC(default) \ - Lower\ bound$. In other words, the value of cooperation is to be defined as

$$value\ of\ cooperation = \frac{TC(default)\ - TC}{TC(default)\ - Lower\ bound}.$$

The value of cooperation for the $\alpha = 1$ and $\alpha = 3$ instances is presented in Tables 15 and 16. In the last two rows we present the sample average of the results and its standard deviation.

Table 15: Value of cooperation for $\alpha = 1$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)
1	69.93%	72.93%	66.93%	71.70%	71.12%	43.52%
2	87.86%	88.74%	70.78%	85.81%	82.54%	54.67%
3	100.00%	100.00%	86.53%	100.00%	77.63%	51.37%
4	77.71%	71.14%	56.16%	75.21%	66.71%	19.51%
5	56.92%	63.15%	48.12%	61.88%	60.68%	24.89%
6	97.14%	91.93%	87.00%	95.21%	91.83%	56.30%
7	58.83%	67.60%	42.16%	63.62%	52.51%	23.73%
8	82.81%	88.61%	78.06%	84.44%	75.18%	62.05%
9	51.13%	61.59%	57.65%	56.31%	54.45%	40.27%
10	91.71%	86.95%	82.44%	91.71%	86.05%	68.38%
11	83.83%	85.69%	91.62%	82.99%	83.88%	65.35%
12	68.31%	78.80%	73.89%	76.96%	59.99%	29.74%
13	69.55%	77.07%	66.74%	71.76%	70.99%	31.03%
14	78.50%	81.93%	67.15%	79.29%	73.17%	33.02%
15	62.17%	73.92%	59.55%	63.27%	40.50%	59.05%
16	68.19%	68.46%	54.95%	68.31%	68.41%	22.14%
17	52.93%	70.89%	62.16%	82.14%	59.81%	74.40%
18	79.21%	76.02%	57.02%	79.41%	75.62%	30.66%
19	59.72%	63.18%	34.57%	63.93%	56.66%	18.96%
20	78.30%	83.42%	74.27%	80.87%	81.63%	55.40%
Average	**73.74%**	**77.60%**	**65.89%**	**76.74%**	**69.47%**	**43.22%**
Standard deviation	3.29%	2.44%	3.49%	2.70%	3.00%	4.12%

Table 16: Value of cooperation for $\alpha = 3$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Heuristic (baseline)
1	83.50%	89.25%	67.58%	86.61%	86.43%	61.43%
2	100.00%	100.00%	85.76%	98.08%	95.68%	54.44%
3	94.06%	88.23%	65.84%	89.56%	94.50%	49.27%
4	98.07%	100.00%	86.91%	98.07%	93.01%	46.79%
5	98.30%	93.77%	84.39%	95.78%	88.08%	64.87%
6	88.53%	93.25%	82.65%	81.64%	78.88%	75.23%
7	73.63%	83.76%	57.99%	80.14%	72.88%	12.01%
8	96.88%	92.58%	98.76%	89.91%	63.96%	65.38%
9	98.19%	97.49%	93.47%	96.61%	89.23%	79.04%
10	94.46%	92.38%	76.01%	86.95%	74.57%	97.84%
11	99.45%	96.86%	68.92%	97.16%	80.33%	66.16%
12	82.66%	84.94%	68.92%	81.57%	72.80%	49.37%
13	80.13%	81.90%	61.30%	80.26%	81.56%	39.67%
14	96.49%	94.99%	73.25%	94.64%	92.89%	73.18%
15	100.00%	100.00%	86.11%	96.06%	41.43%	90.25%
16	96.06%	97.83%	89.66%	97.28%	57.77%	76.10%
17	95.21%	95.53%	83.56%	97.67%	93.13%	93.24%
18	76.59%	82.32%	67.48%	81.28%	68.27%	79.37%
19	67.90%	72.59%	63.69%	73.66%	72.49%	47.65%
20	84.94%	99.81%	66.50%	95.59%	80.27%	86.86%
Average	**90.25%**	**91.87%**	**76.44%**	**89.93%**	**78.91%**	**65.41%**
Standard deviation	2.26%	1.73%	2.70%	1.79%	3.20%	4.82%

The value of cooperation is derived from the total cost for the contactors. Therefore, variations of Stage A that minimize the total cost maximize also the value of cooperation. These are the combinatorial auction, the Vickrey auction and the sequential negotiation with minimal initial prizes (roughly 90% of the maximal value for $\alpha = 3$ and 75% of the maximal value for $\alpha = 1$). Here too, we believe that the less tight lower bounds for $\alpha = 1$ result in the value of cooperation to be lower than for $\alpha = 3$.

Value of considering pre-committed tasks

The value of considering pre-committed tasks in the suggested mechanism is the difference in the value of cooperation between the heuristic allocation (which considers contractors' depots and company's tasks only) and the variations of Stage A that do consider pre-committed tasks. The ratio between the latter and the former is presented for $\alpha = 1$ and for $\alpha = 3$ in Tables 17-18. In the last three rows we present the sample average, sample median and standard deviation of the sample average.

Table 17: Value of considering pre-committed tasks for $\alpha = 1$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize
1	160.68%	167.56%	153.78%	164.73%	163.39%
2	160.70%	162.33%	129.46%	156.97%	150.98%
3	194.69%	194.69%	168.47%	194.69%	151.13%
4	398.26%	364.59%	287.81%	385.45%	341.87%
5	228.66%	253.69%	193.31%	248.57%	243.75%
6	172.54%	163.31%	154.53%	169.12%	163.12%
7	247.94%	284.88%	177.69%	268.12%	221.32%
8	133.45%	142.80%	125.80%	136.09%	121.17%
9	126.95%	152.93%	143.15%	139.82%	135.19%
10	134.12%	127.16%	120.56%	134.12%	125.84%
11	128.28%	131.12%	140.19%	126.99%	128.35%
12	229.66%	264.93%	248.43%	258.73%	201.70%
13	224.11%	248.33%	215.05%	231.21%	228.73%
14	237.70%	248.09%	203.34%	240.10%	221.56%
15	105.27%	125.17%	100.84%	107.14%	68.58%
16	308.08%	309.27%	248.25%	308.61%	309.07%
17	71.14%	95.27%	83.54%	110.40%	80.39%
18	258.37%	247.96%	185.99%	259.01%	246.65%
19	314.98%	333.24%	182.31%	337.17%	298.81%
20	141.33%	150.56%	134.05%	145.98%	147.34%
Average	**198.85%**	**208.39%**	**169.83%**	**206.15%**	**187.45%**
median	181.12%	183.62%	161.50%	181.91%	163.26%
Standard deviation	18.66%	17.85%	12.00%	18.37%	17.25%

Table 18: Value of considering pre-committed tasks for $\alpha = 3$ – Stage A

Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize
1	135.92%	145.28%	110.01%	140.98%	140.69%
2	183.67%	183.67%	157.51%	180.15%	175.74%
3	190.89%	179.06%	133.62%	181.76%	191.78%
4	209.59%	213.71%	185.73%	209.59%	198.78%
5	151.52%	144.54%	130.09%	147.64%	135.77%
6	117.68%	123.95%	109.86%	108.52%	104.84%
7	612.83%	697.18%	482.68%	667.06%	606.65%
8	148.19%	141.61%	151.07%	137.53%	97.84%
9	124.24%	123.35%	118.27%	122.24%	112.90%
10	96.55%	94.43%	77.69%	88.88%	76.22%
11	150.33%	146.42%	104.17%	146.86%	121.42%
12	167.42%	172.04%	139.60%	165.21%	147.45%
13	202.00%	206.47%	154.54%	202.34%	205.61%
14	131.86%	129.82%	100.10%	129.33%	126.94%
15	110.81%	110.81%	95.42%	106.44%	45.91%
16	126.23%	128.55%	117.82%	127.83%	75.92%
17	102.12%	102.46%	89.62%	104.74%	99.88%
18	96.49%	103.71%	85.02%	102.40%	86.01%
19	142.49%	152.35%	133.67%	154.58%	152.13%
20	97.79%	114.91%	76.56%	110.05%	92.41%
Average	**164.93%**	**170.72%**	**137.65%**	**166.71%**	**149.74%**
median	143.07%	139.21%	118.04%	139.25%	124.18%
Standard deviation	25.47%	29.45%	19.79%	28.12%	26.62%

For all variations of Stage A, the value of considering pre-committed tasks decreases for $\alpha = 3$. This strengthens our conjecture that the heuristic does work better for $\alpha = 3$.

Effect of Stage B

Stage B enables the contractors to reduce their total cost by exchanging tasks, which allows, in turn, to increase the value of cooperation. Two versions of Stage B are considered: with money transfers (the complete version) and without money transfers (partial version).

The optimality gaps of the generated solutions for all the combinations of the six variations of Stage A and two variations of Stage B ($2 \times 6 = 12$) and for two values of α ($\alpha = 1,3$) are presented in Tables 19-20. In the last two rows we present the sample average of the results and its standard deviation.

The effect of Stage B can be evaluated by the improvement of the optimality gap as: $100\% \times \left(1 - \frac{optimality\ gap\ after\ stage\ B}{optimality\ gap\ after\ stage\ A}\right)$. Clearly, this ratio is not defined if the optimality gap of Stage A is 0. In such a case there is no need to run Stage B. The Stage B effect for all variations of Stage A and Stage B and for both values of α is presented in Tables 21-22. In the last two rows we present the sample average of the results and its standard deviation.

Table 19: Optimality gaps for $\alpha = 1$ – after Stage B

Instance number	Vickrey auctions without money transfers	Vickrey auctions with money transfers	Sequential combinatorial auctions without money transfers	Sequential combinatorial auctions with money transfers	Vickrey auctions with small clusters without money transfers	Vickrey auctions with small clusters with money transfers	Sequential negotiations with100% initial prize without money transfers	Sequential negotiations with 100% initial prize with money transfers	Sequential negotiations with150% initial prize without money transfers	Sequential negotiations with150% initial prize with money transfers	Heuristic (baseline) without money transfers	Heuristic (baseline) with money transfers
1	6.12%	4.98%	5.54%	5.35%	6.51%	6.08%	4.85%	4.77%	5.35%	4.24%	6.78%	4.24%
2	2.40%	2.40%	2.23%	2.23%	4.30%	3.45%	2.79%	2.79%	3.41%	2.50%	4.16%	3.76%
3	0.00%	0.00%	0.00%	0.00%	3.04%	1.09%	0.00%	0.00%	4.95%	3.05%	10.00%	2.10%
4	3.52%	3.52%	4.51%	3.92%	5.17%	3.86%	3.90%	3.52%	5.17%	4.43%	4.45%	3.50%
5	7.38%	6.48%	6.59%	6.44%	8.01%	7.06%	6.80%	6.42%	6.73%	6.32%	8.96%	8.30%
6	0.60%	0.60%	1.67%	1.45%	2.20%	2.00%	1.00%	1.00%	1.69%	1.69%	3.00%	1.84%
7	6.98%	5.84%	5.83%	5.66%	8.30%	7.11%	6.00%	5.55%	8.32%	7.13%	7.89%	5.81%
8	2.84%	2.84%	1.90%	1.90%	3.59%	2.53%	2.57%	2.57%	4.04%	3.07%	4.69%	3.39%
9	8.89%	7.94%	7.12%	7.07%	7.53%	7.17%	8.02%	7.43%	8.34%	6.48%	8.75%	7.57%
10	1.40%	1.40%	1.90%	1.43%	2.91%	1.76%	1.40%	1.40%	2.33%	1.63%	4.28%	1.43%
11	3.04%	2.58%	2.70%	1.24%	1.60%	1.37%	2.75%	2.75%	2.77%	1.99%	4.37%	2.31%
12	3.83%	3.83%	2.60%	2.01%	2.22%	2.22%	2.82%	2.74%	4.79%	2.92%	5.92%	4.48%
13	6.47%	6.47%	5.34%	5.08%	6.02%	5.77%	6.14%	5.83%	6.66%	6.14%	9.80%	5.50%
14	2.95%	2.66%	2.53%	2.53%	5.19%	3.70%	3.34%	2.41%	4.28%	2.57%	6.70%	2.71%
15	6.60%	4.75%	4.64%	4.64%	7.02%	5.87%	6.42%	5.32%	9.67%	6.12%	6.21%	4.93%
16	6.89%	6.89%	6.84%	6.84%	9.25%	8.18%	6.87%	6.85%	6.85%	6.83%	13.77%	8.83%
17	5.86%	3.50%	4.04%	3.60%	5.19%	3.14%	2.52%	2.52%	3.28%	2.59%	3.57%	3.20%
18	3.30%	3.30%	3.79%	3.70%	6.26%	3.80%	3.27%	3.27%	3.85%	3.38%	5.57%	3.62%
19	6.21%	5.54%	5.71%	5.71%	8.27%	5.72%	5.60%	5.12%	6.65%	5.13%	7.64%	5.83%
20	3.78%	3.71%	3.06%	3.01%	3.61%	3.13%	2.98%	2.91%	3.38%	3.01%	3.40%	2.38%
Average	**4.45%**	**3.96%**	**3.93%**	**3.69%**	**5.31%**	**4.25%**	**4.00%**	**3.76%**	**5.13%**	**4.06%**	**6.50%**	**4.29%**
Standard deviation	0.56%	0.49%	0.46%	0.48%	0.53%	0.50%	0.50%	0.47%	0.50%	0.43%	0.63%	0.49%

Table 20: Optimality gaps for $\alpha = 3$ – after Stage B

Instance number	Vickrey auctions without money transfers	Vickrey auctions with money transfers	Sequential combinatorial auctions without money transfers	Sequential combinatorial auctions with money transfers	Vickrey auctions with small clusters without money transfers	Vickrey auctions with small clusters with money transfers	Sequential negotiations with100% initial prize without money transfers	Sequential negotiations with 100% initial prize with money transfers	Sequential negotiations with150% initial prize without money transfers	Sequential negotiations with150% initial prize with money transfers	Heuristic (baseline) without money transfers	Heuristic (baseline) with money transfers
1	3.06%	2.08%	2.01%	2.01%	4.18%	4.17%	1.70%	1.70%	2.53%	2.22%	5.34%	1.84%
2	0.00%	0.00%	0.00%	0.00%	2.90%	2.70%	0.40%	0.40%	0.90%	0.74%	6.10%	1.39%
3	1.29%	1.29%	2.53%	2.53%	7.01%	4.24%	2.25%	1.40%	1.20%	1.05%	3.80%	1.10%
4	0.32%	0.32%	0.00%	0.00%	2.11%	2.11%	0.32%	0.32%	1.14%	1.14%	3.49%	2.55%
5	0.00%	0.00%	0.42%	0.42%	2.50%	2.41%	0.69%	0.37%	1.92%	1.63%	5.45%	4.37%
6	2.31%	2.11%	1.37%	1.37%	3.45%	2.50%	3.65%	2.44%	4.17%	3.93%	3.87%	1.91%
7	3.44%	2.74%	2.15%	1.71%	5.37%	3.86%	2.61%	2.61%	3.53%	3.04%	4.42%	1.62%
8	0.51%	0.51%	1.20%	0.98%	0.20%	0.20%	1.63%	1.63%	5.59%	0.88%	4.12%	0.45%
9	0.36%	0.29%	0.49%	0.49%	1.27%	1.00%	0.66%	0.58%	2.08%	0.23%	2.40%	0.32%
10	1.00%	1.00%	1.37%	0.65%	1.65%	1.94%	2.32%	1.15%	3.31%	1.79%	0.39%	0.39%
11	0.12%	0.12%	0.67%	0.67%	6.27%	4.46%	0.61%	0.61%	1.05%	1.05%	5.19%	0.39%
12	1.74%	1.74%	2.15%	2.15%	3.15%	3.15%	2.47%	1.70%	3.81%	1.65%	2.50%	1.95%
13	4.25%	3.92%	3.59%	3.36%	7.61%	5.27%	3.45%	3.45%	3.96%	3.19%	8.57%	3.19%
14	0.59%	0.59%	0.83%	0.27%	1.92%	0.43%	0.39%	0.39%	1.18%	1.18%	1.19%	0.75%
15	0.00%	0.00%	0.00%	0.00%	1.90%	1.53%	0.55%	0.55%	4.38%	0.65%	0.59%	0.59%
16	0.61%	0.61%	0.34%	0.34%	1.58%	1.58%	0.42%	0.42%	1.13%	0.41%	2.85%	1.54%
17	0.76%	0.76%	0.71%	0.71%	2.55%	1.86%	0.37%	0.37%	1.08%	1.08%	1.07%	0.08%
18	3.84%	3.84%	2.93%	2.93%	4.26%	4.26%	3.09%	3.09%	5.14%	3.50%	3.40%	2.78%
19	4.79%	4.68%	4.59%	4.59%	4.87%	4.53%	4.42%	4.41%	4.61%	4.59%	5.37%	4.69%
20	2.14%	1.31%	0.03%	0.03%	4.65%	1.29%	0.00%	0.00%	2.79%	1.76%	0.26%	0.26%
Average	**1.56%**	**1.40%**	**1.37%**	**1.26%**	**3.47%**	**2.67%**	**1.60%**	**1.38%**	**2.78%**	**1.79%**	**3.52%**	**1.61%**
Standard deviation	0.36%	0.33%	0.30%	0.30%	0.46%	0.34%	0.31%	0.28%	0.36%	0.28%	0.50%	0.31%

Table 21: Improvement ratio for $\alpha = 1$ – after Stage B

Instance number	Vickrey auctions without money transfers	Vickrey auctions with money transfers	Sequential combinatorial auctions without money transfers	Sequential combinatorial auctions with money transfers	Vickrey auctions with small clusters without money transfers	Vickrey auctions with small clusters with money transfers	Sequential negotiations with100% initial prize without money transfers	Sequential negotiations with 100% initial prize with money transfers	Sequential negotiations with150% initial prize without money transfers	Sequential negotiations with150% initial prize with money transfers	Heuristic (baseline) without money transfers	Heuristic (baseline) with money transfers
1	0.00%	18.59%	0.00%	3.49%	2.69%	9.03%	16.06%	17.47%	9.23%	27.95%	37.86%	61.08%
2	0.00%	0.00%	0.00%	0.00%	22.96%	38.25%	0.00%	0.00%	0.00%	26.76%	50.47%	55.22%
3	0.00%	0.00%	0.00%	0.00%	0.00%	64.30%	0.00%	0.00%	0.00%	38.49%	1.80%	79.32%
4	0.00%	0.00%	0.00%	13.09%	22.87%	42.39%	0.00%	9.65%	0.00%	14.38%	61.75%	69.96%
5	3.03%	14.92%	0.00%	2.15%	11.26%	21.77%	0.00%	5.59%	3.85%	9.63%	28.68%	33.93%
6	0.00%	0.00%	0.00%	12.66%	17.07%	24.76%	0.00%	0.00%	0.00%	0.00%	64.34%	78.07%
7	4.34%	20.00%	0.00%	3.00%	16.58%	28.56%	7.76%	14.60%	0.00%	14.32%	37.99%	54.35%
8	0.00%	0.00%	0.00%	0.00%	0.00%	29.60%	0.00%	0.00%	0.00%	23.98%	22.58%	43.92%
9	0.00%	10.68%	0.00%	0.71%	3.41%	8.01%	0.00%	7.34%	0.00%	22.23%	17.88%	28.96%
10	0.00%	0.00%	12.95%	34.44%	0.00%	39.62%	0.00%	0.00%	0.00%	30.01%	16.41%	72.10%
11	0.00%	15.06%	0.00%	53.88%	0.00%	14.52%	13.73%	13.73%	8.54%	34.22%	30.49%	63.35%
12	0.00%	0.00%	0.00%	22.70%	30.34%	30.34%	0.00%	2.56%	0.00%	38.96%	27.09%	44.85%
13	7.16%	7.16%	0.00%	4.92%	20.46%	23.75%	5.55%	10.28%	0.00%	7.78%	32.44%	62.05%
14	14.78%	23.12%	13.35%	13.35%	0.00%	28.63%	0.00%	27.74%	0.00%	39.96%	33.21%	73.00%
15	0.00%	27.93%	0.00%	0.00%	0.00%	16.45%	0.00%	17.10%	3.28%	38.79%	12.59%	30.61%
16	0.00%	0.00%	0.00%	0.00%	2.56%	13.87%	0.00%	0.26%	0.00%	0.27%	10.30%	42.45%
17	8.20%	45.13%	0.00%	11.08%	0.00%	39.45%	0.00%	0.00%	40.26%	52.85%	0.00%	10.46%
18	0.00%	0.00%	0.00%	2.51%	5.06%	42.44%	0.00%	0.00%	0.00%	12.26%	45.60%	64.63%
19	0.00%	10.87%	0.00%	0.00%	14.92%	41.15%	0.00%	8.51%	0.00%	22.96%	35.06%	50.40%
20	4.74%	6.64%	0.00%	1.75%	22.76%	33.06%	15.14%	17.26%	0.00%	11.02%	56.59%	69.59%
Average	**2.11%**	**10.00%**	**1.31%**	**8.99%**	**9.65%**	**29.50%**	**2.91%**	**7.61%**	**3.26%**	**23.34%**	**31.16%**	**54.42%**
Standard deviation	0.91%	2.82%	0.93%	3.19%	2.36%	3.16%	1.29%	1.88%	2.10%	3.33%	4.24%	4.25%

Table 22: Improvement ratio for $\alpha = 3$ – after Stage B

Instance number	Vickrey auctions without money transfers	Vickrey auctions with money transfers	Sequential combinatorial auctions without money transfers	Sequential combinatorial auctions with money transfers	Vickrey auctions with small clusters without money transfers	Vickrey auctions with small clusters with money transfers	Sequential negotiations with100% initial prize without money transfers	Sequential negotiations with 100% initial prize with money transfers	Sequential negotiations with150% initial prize without money transfers	Sequential negotiations with150% initial prize with money transfers	Heuristic (baseline) without money transfers	Heuristic (baseline) with money transfers
1	0.00%	31.94%	0.00%	0.00%	28.40%	28.65%	31.75%	31.75%	0.00%	12.25%	22.29%	73.21%
2	0.00%	0.00%	0.00%	0.00%	0.00%	7.04%	0.00%	0.00%	0.00%	17.50%	30.08%	84.08%
3	0.00%	0.00%	0.00%	0.00%	0.00%	39.59%	0.00%	37.75%	0.00%	12.20%	62.24%	89.11%
4	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	56.70%	68.29%
5	100.00%	100.00%	58.07%	58.07%	0.00%	3.41%	0.00%	46.81%	0.00%	15.14%	0.00%	19.81%
6	0.00%	8.49%	0.00%	0.00%	0.00%	27.50%	0.00%	33.00%	0.00%	5.84%	20.25%	60.74%
7	0.00%	20.47%	0.00%	20.19%	0.00%	28.05%	0.00%	0.00%	0.00%	13.96%	58.40%	84.77%
8	0.00%	0.00%	0.00%	18.46%	0.00%	0.00%	0.00%	0.00%	0.00%	84.34%	23.47%	91.72%
9	0.00%	18.17%	0.00%	0.00%	0.00%	21.32%	0.00%	12.57%	0.00%	89.19%	39.65%	92.03%
10	0.00%	0.00%	0.00%	52.75%	60.68%	53.76%	0.00%	50.33%	25.17%	59.57%	0.00%	0.00%
11	0.00%	0.00%	0.00%	0.00%	0.00%	28.86%	0.00%	0.00%	74.06%	74.06%	23.60%	94.31%
12	29.33%	29.33%	0.00%	0.00%	27.38%	27.38%	5.69%	35.19%	0.00%	56.84%	63.66%	71.68%
13	0.00%	7.90%	7.60%	13.63%	4.46%	33.82%	18.35%	18.35%	0.00%	19.37%	27.85%	73.14%
14	0.00%	0.00%	0.00%	67.43%	55.48%	90.09%	56.41%	56.41%	0.00%	0.00%	72.41%	82.72%
15	0.00%	0.00%	0.00%	0.00%	0.00%	19.77%	0.00%	0.00%	42.05%	91.47%	56.28%	56.28%
16	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	81.67%	93.26%	20.35%	57.00%
17	0.00%	0.00%	0.00%	0.00%	0.00%	27.28%	0.00%	0.00%	0.00%	0.00%	0.00%	92.59%
18	0.00%	0.00%	0.00%	0.00%	18.89%	18.89%	0.00%	0.00%	0.00%	31.77%	0.00%	18.29%
19	10.30%	12.32%	0.00%	0.00%	18.76%	24.46%	0.00%	0.38%	0.00%	0.37%	36.31%	44.34%
20	0.00%	38.91%	0.00%	0.00%	0.00%	72.17%	100.00%	100.00%	0.00%	36.81%	86.35%	86.35%
Average	**6.98%**	**13.38%**	**3.28%**	**11.53%**	**10.70%**	**27.60%**	**10.61%**	**21.13%**	**11.15%**	**35.70%**	**35.00%**	**67.02%**
Standard deviation	5.26%	5.48%	2.98%	4.99%	4.33%	5.33%	5.84%	6.28%	5.78%	7.93%	5.98%	6.30%

Tables 19-20 show that applying Stage B can significantly improve the tasks' allocation obtained in Stage A. As expected, allowing money transfers in Stage B results in a consistently larger improvement compared with the simpler version where no money transfers are allowed. The combinatorial auction method remains the best protocol for Stage A, even after these improvements.

From Tables 21-22 we observe that, as expected, the contribution of Stage B, is significant if the allocation of Stage A is far from the optimum. For the Vickrey auction with small clusters, the effect of Stage B is approximately 10% without money transfers and 30% with money transfers. The heuristic allocation improves even more. The improvement ratio of Stage B is more than 30% without money transfers and well over 50% with money transfers (for $\alpha = 3$ it is equal to 67%). In fact, applying the heuristic method and then Stage B results in a fairly good allocation for both values of α. The value of considering pre-committed tasks in this case decreases significantly.

Profitability of Outsourcing

So far, we have seen that the suggested mechanism yields good allocation of tasks and hence creates a large value of cooperation. What we have not discussed yet is the distribution of this value between the parties. This distribution is determined by the sum of rewards the contractors are paid for their services in each variation of Stage A.

Profitability of outsourcing from the company's perspective can be evaluated by comparing the sum of rewards it pays the contractors with its default cost when no outsourcing is applied. The sum of rewards paid in each variation of Stage A (except the heuristic allocation) for $\alpha = 1$ and for $\alpha = 3$ is presented in Table 23.

Table 23: Sum of rewards paid in all variations of Stage A for $\alpha = 1$ and for $\alpha = 3$

		$\alpha = 1$					$\alpha = 3$				
Instance number	Total cost for the company without outscoring	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize
1	4,392.91	4,986.97	4,076.66	3,941.50	3,333.79	3,791.93	4,491.27	4,191.61	4,008.68	3,250.98	3,758.82
2	3,887.04	3,889.50	3,468.10	3,372.46	2,645.86	3,299.75	4,714.52	3,743.33	3,557.90	2,531.88	3,067.73
3	4,000.97	4,417.05	3,633.19	3,987.20	2,523.39	3,459.84	4,271.20	3,866.12	3,748.33	2,768.00	3,248.44
4	3,753.33	3,502.63	3,432.49	3,366.05	2,855.76	3,312.19	4,598.11	3,914.50	4,131.26	2,708.14	3,194.07
5	4,311.26	3,820.03	4,059.55	3,805.90	3,422.42	3,950.19	5,439.03	4,308.69	4,181.19	3,264.02	3,837.11
6	3,930.30	4,881.09	3,748.99	3,671.98	2,640.33	3,095.22	4,400.93	3,591.28	3,500.52	2,866.88	3,267.82
7	3,708.15	3,512.09	3,386.37	3,500.59	3,005.65	3,759.77	3,744.23	3,657.88	3,962.04	3,127.90	3,718.89
8	4,285.52	4,758.86	4,218.43	4,324.04	3,297.29	3,885.36	5,178.90	4,602.44	4,404.32	3,338.27	4,319.52
9	3,988.73	3,735.99	3,523.08	3,764.14	3,307.53	3,840.50	4,607.65	3,984.34	3,815.85	2,841.17	3,375.95
10	4,474.78	4,328.95	4,263.55	4,051.55	3,356.29	3,878.68	4,709.60	4,633.20	4,494.02	3,426.24	4,078.84
11	3,609.71	3,679.59	3,199.72	3,423.01	2,549.26	3,126.29	4,673.00	3,773.20	3,606.34	2,356.10	2,973.98
12	4,031.77	4,512.44	3,999.00	3,982.84	3,363.94	3,808.93	4,740.36	3,967.82	3,798.39	3,227.94	3,596.76
13	3,858.65	3,441.84	3,425.31	3,377.40	2,744.18	3,167.43	3,440.94	3,460.56	3,510.05	2,616.49	3,162.01
14	3,866.91	3,444.29	3,367.07	3,302.48	2,899.54	3,383.21	3,637.70	3,731.07	3,341.36	2,829.65	3,240.36
15	3,400.73	3,436.71	3,339.25	3,324.04	2,703.58	3,876.95	4,740.53	3,777.64	3,725.54	2,571.42	4,018.56
16	4,255.02	4,634.42	3,891.81	3,942.08	3,128.61	3,532.51	5,536.77	4,535.71	4,432.11	3,277.81	4,616.91
17	3,843.58	4,142.57	3,404.01	3,585.66	3,046.29	3,674.46	4,172.28	3,909.49	3,737.89	2,866.56	3,500.20
18	3,752.76	3,521.20	3,540.56	3,314.30	2,754.52	3,269.81	4,553.71	3,672.47	3,774.98	2,811.95	3,344.43
19	3,700.60	3,592.03	3,547.10	3,376.92	3,006.45	3,844.58	4,090.82	3,711.06	3,409.69	2,838.60	3,441.91
20	4,416.40	4,054.22	4,225.46	3,958.63	3,326.25	4,053.79	5,232.09	4,666.76	4,573.02	3,444.88	4,184.35

Sum of rewards as a percentage of the company's original cost without outsourcing is presented in Table 24. In the last three rows we present the sample average, sample median and standard deviation of the sample average.

Table 24: Sum of the rewards paid in all variations of Stage A as a percentage of the company's original cost

	$\alpha = 1$					$\alpha = 3$				
Instance number	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize	Vickrey auctions	Sequential combinatorial auctions	Vickrey auctions with small clusters	Sequential negotiations with 100% of tasks working time cost as initial prize	Sequential negotiations with 150% of tasks working time cost as initial prize
1	113.52%	92.80%	89.72%	75.89%	86.32%	102.24%	95.42%	91.25%	74.01%	85.57%
2	100.06%	89.22%	86.76%	68.07%	84.89%	121.29%	96.30%	91.53%	65.14%	78.92%
3	110.40%	90.81%	99.66%	63.07%	86.48%	106.75%	96.63%	93.69%	69.18%	81.19%
4	93.32%	91.45%	89.68%	76.09%	88.25%	122.51%	104.29%	110.07%	72.15%	85.10%
5	88.61%	94.16%	88.28%	79.38%	91.62%	126.16%	99.94%	96.98%	75.71%	89.00%
6	124.19%	95.39%	93.43%	67.18%	78.75%	111.97%	91.37%	89.06%	72.94%	83.14%
7	94.71%	91.32%	94.40%	81.06%	101.39%	100.97%	98.64%	106.85%	84.35%	100.29%
8	111.05%	98.43%	100.90%	76.94%	90.66%	120.85%	107.40%	102.77%	77.90%	100.79%
9	93.66%	88.33%	94.37%	82.92%	96.28%	115.52%	99.89%	95.67%	71.23%	84.64%
10	96.74%	95.28%	90.54%	75.00%	86.68%	105.25%	103.54%	100.43%	76.57%	91.15%
11	101.94%	88.64%	94.83%	70.62%	86.61%	129.46%	104.53%	99.91%	65.27%	82.39%
12	111.92%	99.19%	98.79%	83.44%	94.47%	117.58%	98.41%	94.21%	80.06%	89.21%
13	89.20%	88.77%	87.53%	71.12%	82.09%	89.17%	89.68%	90.97%	67.81%	81.95%
14	89.07%	87.07%	85.40%	74.98%	87.49%	94.07%	96.49%	86.41%	73.18%	83.80%
15	101.06%	98.19%	97.74%	79.50%	114.00%	139.40%	111.08%	109.55%	75.61%	118.17%
16	108.92%	91.46%	92.65%	73.53%	83.02%	130.12%	106.60%	104.16%	77.03%	108.51%
17	107.78%	88.56%	93.29%	79.26%	95.60%	108.55%	101.71%	97.25%	74.58%	91.07%
18	93.83%	94.35%	88.32%	73.40%	87.13%	121.34%	97.86%	100.59%	74.93%	89.12%
19	97.07%	95.85%	91.25%	81.24%	103.89%	110.54%	100.28%	92.14%	76.71%	93.01%
20	91.80%	95.68%	89.63%	75.32%	91.79%	118.47%	105.67%	103.55%	78.00%	94.75%
Average	**100.94%**	**92.75%**	**92.36%**	**75.40%**	**90.87%**	**114.61%**	**100.29%**	**97.85%**	**74.12%**	**90.59%**
median	98.56%	92.13%	91.95%	75.60%	87.87%	116.55%	99.92%	97.12%	74.76%	89.06%
Standard deviation	2.28%	0.85%	1.02%	1.25%	1.91%	2.90%	1.24%	1.57%	1.10%	2.28%

The sum of the prizes the company pays the contractors increases as α increases for the three sealed-bid auctions, i.e., for Vickrey, sequential combinatorial and Vickrey auction done with small clusters. This is true, since payments for the sealed-bid auctions are determined by the winner's contribution to the system and that the contribution is much higher if contractors' routes are small and separate, i.e. for large values of α. However, α does not significantly affect the sum of prizes awarded in the sequential negotiation mechanism.

For $\alpha = 1$, all the variations of Stage A other than the vickrey auction are profitable for the company, since the sum of prizes it pays the contractors is smaller than its default cost without outsourcing. Sequential negotiation with low initial prizes is the most profitable allocation method from the company's perspective. For $\alpha = 3$, the sequential combinatorial auction is marginally no longer profitable for the company. However, note that this conclusion is based on a conservative assumption saying that the cost structure of the company is identical to those of the contractors. In practice, most of the companies outsource tasks that are out of their core business because they believe that these tasks can be carried out more efficiently by specialized contractors.

Vickrey auction, for both values of α, is the least profitable for the company. For both values of α, the sum of prizes that the company pays the contractors is larger than its original cost. What can be the reason?

Vickrey auction yields the largest sum of rewards to contractors (for both values of α), since the reward, which is the second-lowest bid, is equal to the marginal cost of one task. Sum of the marginal costs, that is the sum of rewards the company pays, is usually significantly greater than the cost of a combined set of tasks (certainly when these tasks are located closely). When the company does offer a set of tasks jointly, then the sum of prizes it pays decreases, even if the payment is set by the Vickrey procedure, as is the case for Vickrey auction done with small clusters.

Let us now compare the profits of the company and the contractors for both values of α. The profit for the contractors is equal to the sum of prizes net of their increase in costs. The profit for the company is equal to its original cost without outsourcing net of the sum of prizes it awards to the contractors. These are presented in Tables 25-26. The sum of the profit for the company and the profit for contractors is equal to the value of cooperation.

Note: in the rest of this section we consider profits for contractors and profits for the company generated after stage A. Stage B increases the value of cooperation and hence, the contractors profits. However, we do not address it in this section since it does not change the company's profits.

Table 25: Profits of the company and the contractors - $\alpha = 1$

	Vickrey auctions		Sequential combinatorial auctions		Vickrey auctions with small clusters		Sequential negotiations with 100% of tasks working time cost as initial prize		Sequential negotiations with 150% of tasks working time cost as initial prize	
Instance number	company	contractors	company	contractors	company	contractors	company	contractors	company	contractors
1	-594.06	1772.92	316.25	913.09	451.41	676.8	1059.12	149.43	600.98	597.80
2	-2.46	1433.94	418.94	1027.03	514.58	638.62	1241.18	157.02	587.29	757.56
3	-416.08	2034.95	367.78	1251.09	13.77	1387.07	1477.58	141.29	541.13	715.58
4	250.7	823.84	320.84	662.86	387.28	389.26	897.57	142.42	441.14	481.27
5	491.23	464.95	251.71	809.16	505.36	303.01	888.84	150.61	361.07	658.21
6	-950.792	2397.3	181.31	1187.738	258.32	1037.198	1289.97	127.86	835.08	532.40
7	196.06	612.174	321.778	606.846	207.56	371.654	702.50	171.51	-51.62	773.06
8	-473.34	1618.38	67.09	1158.19	-38.52	1117.92	988.23	179.42	400.16	639.50
9	252.74	512.57	465.65	456.28	224.59	638.38	681.20	161.70	148.23	666.73
10	145.83	1147.84	211.23	1015.29	423.23	739.68	1118.49	175.18	596.10	617.72
11	-69.88	1265.72	409.99	812.38	186.7	1120.23	1060.45	123.39	483.42	713.12
12	-480.666	1221.096	32.769	821.368	48.927	752.03	667.83	166.35	222.84	427.45
13	416.81	821.41	433.34	938.66	481.25	706.92	1114.47	162.98	691.22	572.50
14	422.62	681.73	499.84	652.75	564.43	380.25	967.37	148.10	483.70	545.65
15	-35.98	858.046	61.48	915.976	76.69	710.736	697.15	139.52	-476.22	1011.72
16	-379.4	1688.22	363.21	950.68	312.94	741.7	1126.41	184.64	722.51	590.53
17	-298.99	907.52	439.57	375.43	257.92	456.69	797.29	147.10	169.12	518.58
18	231.56	894.99	212.2	868.99	438.46	372.51	998.24	131.11	482.95	592.53
19	108.57	698.38	153.5	700.23	323.68	143.38	694.15	169.65	-143.98	909.51
20	362.18	852.64	190.94	1103.27	457.77	694.49	1090.15	164.62	362.61	903.88

Table 26: Profits of the company and for the contractors - $\alpha = 3$

	Vickrey auctions		Sequential combinatorial auctions		Vickrey auctions with small clusters		Sequential negotiations with 100% of tasks working time cost as initial prize		Sequential negotiations with 150% of tasks working time cost as initial prize	
Instance number	company	contractors	company	contractors	company	contractors	company	contractors	company	contractors
1	-98.36	1347.79	201.30	1134.17	384.23	627.06	1141.93	154.04	634.09	659.23
2	-827.48	2342.59	143.71	1371.40	329.14	970.19	1355.16	130.92	819.31	630.40
3	-270.23	1723.00	134.85	1227.91	252.64	764.30	1232.97	150.36	752.53	707.02
4	-844.78	2035.87	-161.17	1375.68	-377.93	1433.42	1045.19	145.90	559.26	570.36
5	-1127.77	2377.78	2.57	1189.86	130.07	943.12	1047.24	170.73	474.15	645.89
6	-470.63	1798.20	339.02	1059.35	429.78	809.65	1063.42	160.79	662.48	520.30
7	-36.08	730.76	50.27	740.02	-253.89	801.04	580.25	175.90	-10.74	698.42
8	-893.38	2082.30	-316.92	1453.00	-118.80	1330.82	947.25	156.11	-34.00	818.94
9	-618.92	1927.05	4.39	1294.44	172.88	1072.38	1147.56	139.52	612.78	575.97
10	-234.82	1562.36	-158.42	1456.81	-19.24	1087.49	1048.54	173.57	395.94	652.09
11	-1063.29	2454.05	-163.49	1518.07	3.37	960.40	1253.61	105.11	635.73	487.60
12	-708.59	1718.86	63.95	974.15	233.38	609.00	803.83	193.06	435.01	454.74
13	417.71	927.62	398.09	976.97	348.60	680.60	1242.16	105.39	696.64	672.72
14	229.21	988.60	135.84	1063.06	525.55	398.90	1037.26	157.14	626.55	545.75
15	-1339.80	2323.33	-376.91	1360.43	-324.81	1171.71	829.31	115.43	-617.83	1025.33
16	-1281.75	2444.54	-280.69	1464.86	-177.09	1262.44	977.21	200.33	-361.89	1061.23
17	-328.70	1436.15	-65.91	1177.07	105.69	866.20	977.02	158.93	343.38	739.83
18	-800.95	1822.72	80.29	1017.98	-22.22	922.53	940.81	143.57	408.33	502.44
19	-390.22	1325.61	-10.46	1010.55	290.91	586.57	862.00	152.72	258.69	739.97
20	-815.69	1807.31	-250.36	1415.64	-156.62	933.00	971.52	144.39	232.05	705.02

The relative distribution of the value of cooperation between the contractors and the company as a percentage of total value of cooperation is presented in Tables 27-28. In the last three rows we present the sample average, sample median and standard deviation of the sample average.

Table 27: Sharing the value of cooperation - $\alpha = 1$

	Vickrey auctions		Sequential combinatorial auctions		Vickrey auctions with small clusters		Sequential negotiations with 100% of tasks working time cost as initial prize		Sequential negotiations with 150% of tasks working time cost as initial prize	
Instance number	company	contractors	company	contractors	company	contractors	company	contractors	company	contractors
1	-50.39%	150.39%	25.73%	74.27%	40.01%	59.99%	87.64%	12.36%	50.13%	49.87%
2	-0.17%	100.17%	28.97%	71.03%	44.62%	55.38%	88.77%	11.23%	43.67%	56.33%
3	-25.70%	125.70%	22.72%	77.28%	0.98%	99.02%	91.27%	8.73%	43.06%	56.94%
4	23.33%	76.67%	32.62%	67.38%	49.87%	50.13%	86.31%	13.69%	47.82%	52.18%
5	51.37%	48.63%	23.73%	76.27%	62.52%	37.48%	85.51%	14.49%	35.42%	64.58%
6	-65.73%	165.73%	13.24%	86.76%	19.94%	80.06%	90.98%	9.02%	61.07%	38.93%
7	24.26%	75.74%	34.65%	65.35%	35.83%	64.17%	80.38%	19.62%	-7.15%	107.15%
8	-41.34%	141.34%	5.48%	94.52%	-3.57%	103.57%	84.63%	15.37%	38.49%	61.51%
9	33.02%	66.98%	50.51%	49.49%	26.03%	73.97%	80.82%	19.18%	18.19%	81.81%
10	11.27%	88.73%	17.22%	82.78%	36.39%	63.61%	86.46%	13.54%	49.11%	50.89%
11	-5.84%	105.84%	33.54%	66.46%	14.29%	85.71%	89.58%	10.42%	40.40%	59.60%
12	-64.92%	164.92%	3.84%	96.16%	6.11%	93.89%	80.06%	19.94%	34.27%	65.73%
13	33.66%	66.34%	31.58%	68.42%	40.50%	59.50%	87.24%	12.76%	54.70%	45.30%
14	38.27%	61.73%	43.37%	56.63%	59.75%	40.25%	86.72%	13.28%	46.99%	53.01%
15	-4.38%	104.38%	6.29%	93.71%	9.74%	90.26%	83.32%	16.68%	-88.93%	188.93%
16	-28.99%	128.99%	27.64%	72.36%	29.67%	70.33%	85.92%	14.08%	55.03%	44.97%
17	-49.13%	149.13%	53.93%	46.07%	36.09%	63.91%	84.42%	15.58%	24.59%	75.41%
18	20.55%	79.45%	19.63%	80.37%	54.07%	45.93%	88.39%	11.61%	44.91%	55.09%
19	13.45%	86.55%	17.98%	82.02%	69.30%	30.70%	80.36%	19.64%	-18.81%	118.81%
20	29.81%	70.19%	14.75%	85.25%	39.73%	60.27%	86.88%	13.12%	28.63%	71.37%
Average	**-2.88%**	**102.88%**	**25.37%**	**74.63%**	**33.59%**	**66.41%**	**85.78%**	**14.22%**	**30.08%**	**69.92%**
Median	5.55%	94.45%	24.73%	75.27%	36.24%	63.76%	86.39%	13.62%	41.73%	58.27%
Standard deviation	8.43%		3.19%		4.75%		0.78%		7.89%	

Table 28: Sharing the value of cooperation - $\alpha = 3$

	Vickrey auctions		Sequential combinatorial auctions		Vickrey auctions with small clusters		Sequential negotiations with 100% of tasks working time cost as initial prize		Sequential negotiations with 150% of tasks working time cost as initial prize	
Instance number	company	contractors	company	contractors	company	contractors	company	contractors	company	contractors
1	-7.87%	107.87%	15.07%	84.93%	37.99%	62.01%	88.11%	11.89%	49.03%	50.97%
2	-54.62%	154.62%	9.49%	90.51%	25.33%	74.67%	91.19%	8.81%	56.52%	43.48%
3	-18.60%	118.60%	9.90%	90.10%	24.84%	75.16%	89.13%	10.87%	51.56%	48.44%
4	-70.93%	170.93%	-13.27%	113.27%	-35.81%	135.81%	87.75%	12.25%	49.51%	50.49%
5	-90.22%	190.22%	0.22%	99.78%	12.12%	87.88%	85.98%	14.02%	42.33%	57.67%
6	-35.45%	135.45%	24.24%	75.76%	34.68%	65.32%	86.87%	13.13%	56.01%	43.99%
7	-5.19%	105.19%	6.36%	93.64%	-46.40%	146.40%	76.74%	23.26%	-1.56%	101.56%
8	-75.14%	175.14%	-27.90%	127.90%	-9.80%	109.80%	85.85%	14.15%	-4.33%	104.33%
9	-47.31%	147.31%	0.34%	99.66%	13.88%	86.12%	89.16%	10.84%	51.55%	48.45%
10	-17.69%	117.69%	-12.20%	112.20%	-1.80%	101.80%	85.80%	14.20%	37.78%	62.22%
11	-76.45%	176.45%	-12.07%	112.07%	0.35%	99.65%	92.26%	7.74%	56.59%	43.41%
12	-70.14%	170.14%	6.16%	93.84%	27.70%	72.30%	80.63%	19.37%	48.89%	51.11%
13	31.05%	68.95%	28.95%	71.05%	33.87%	66.13%	92.18%	7.82%	50.87%	49.13%
14	18.82%	81.18%	11.33%	88.67%	56.85%	43.15%	86.84%	13.16%	53.45%	46.55%
15	-136.22%	236.22%	-38.32%	138.32%	-38.35%	138.35%	87.78%	12.22%	-151.62%	251.62%
16	-110.23%	210.23%	-23.70%	123.70%	-16.32%	116.32%	82.99%	17.01%	-51.75%	151.75%
17	-29.68%	129.68%	-5.93%	105.93%	10.87%	89.13%	86.01%	13.99%	31.70%	68.30%
18	-78.39%	178.39%	7.31%	92.69%	-2.47%	102.47%	86.76%	13.24%	44.83%	55.17%
19	-41.72%	141.72%	-1.05%	101.05%	33.15%	66.85%	84.95%	15.05%	25.90%	74.10%
20	-82.26%	182.26%	-21.49%	121.49%	-20.17%	120.17%	87.06%	12.94%	24.76%	75.24%
Average	**-49.91%**	**149.91%**	**-1.83%**	**101.83%**	**7.03%**	**92.97%**	**86.70%**	**13.30%**	**26.10%**	**73.90%**
Median	-50.97%	150.97%	0.28%	99.72%	11.50%	88.51%	86.86%	13.15%	46.86%	53.14%
Standard deviation	9.78%		4.01%		6.51%		0.84%		11.42%	

The results above show that the suggested mechanism is always profitable to the contractors (even before applying Stage B). In fact, the profit for the contractors is very high in the sequential combinatorial auctions, which appears to be the most efficient mechanism.

Note that it is possible to achieve a relatively efficient allocation profitable to all parties (even under our conservative assumption) by using the Vickery auction with small clusters and then apply stage B. This may be the most attractive line of action since it is also computationally less involved than the combinatorial auction.

5.4.3 Summary

The allocations generated by the suggested mechanism are close to the optimal allocation of tasks to contractors that a central planner could have found. This follows from rather small optimality gaps of the obtained solutions. The average optimality gaps and their standard deviations are presented in Table 29.

Table 29: Average optimality gaps for generated allocations

		$\alpha = 1$			$\alpha = 3$		
		Stage A	Stage B - no money transfers	Stage B - with money transfers	Stage A	Stage B - no money transfers	Stage B - with money transfers
Vickrey auctions	Average	4.57%	4.45%	3.96%	1.63%	1.56%	1.40%
	Std.	0.58%	0.56%	0.49%	0.37%	0.36%	0.33%
Sequential combinatorial auctions	Average	3.96%	3.93%	3.69%	1.41%	1.37%	1.26%
	Std.	0.45%	0.46%	0.48%	0.30%	0.30%	0.30%
Vickrey auctions with small clusters	Average	5.91%	5.31%	4.25%	3.98%	3.47%	2.67%
	Std.	0.59%	0.53%	0.50%	0.48%	0.46%	0.34%
Sequential negotiations with low initial prizes	Average	4.14%	4%	3.76%	1.74%	1.60%	1.38%
	Std.	0.52%	0.50%	0.47%	0.31%	0.31%	0.28%
Sequential negotiations with high initial prizes	Average	5.31%	5.13%	4.06%	3.39%	2.78%	1.79%
	Std.	0.50%	0.50%	0.43%	0.44%	0.36%	0.28%
Heuristic	Average	9.57%	6.5%	4.29%	5.70%	3.52%	1.61%
	Std.	0.71%	0.63%	0.49%	0.76%	0.50%	0.31%

Another feature of our mechanism is a high value of cooperation. Average values and standard deviations are presented in Table 30 (percentage of the gap between the default status and a lower bound of the central planner solution).

Table 30: Average value of cooperation for generated allocations

		$\alpha = 1$			$\alpha = 3$		
		Stage A	Stage B - no money transfers	Stage B - with money transfers	Stage A	Stage B - no money transfers	Stage B - with money transfers
Vickrey auctions	Average	73.74%	74.44%	77.55%	90.25%	90.77%	91.79%
	Std.	3.29%	3.19%	2.62%	2.26%	2.15%	1.93%
Sequential combinatorial auctions	Average	77.6%	77.81%	79.27%	91.87%	92.13%	92.79%
	Std.	2.44%	2.47%	2.55%	1.73%	1.73%	1.71%
Vickrey auctions with small clusters	Average	65.89%	69.63%	76.21%	76.44%	79.63%	84.48%
	Std.	3.49%	3.08%	2.61%	2.70%	2.56%	1.91%
Sequential negotiations with low initial prizes	Average	76.74%	77.48%	78.96%	89.93%	90.76%	92%
	Std.	2.70%	2.67%	2.41%	1.79%	1.80%	1.69%
Sequential negotiations with high initial prizes	Average	69.47%	70.70%	77.31%	78.91%	83%	89.48%
	Std.	3.00%	2.88%	2.25%	3.20%	240.31%	1.69%
Heuristic	Average	43.22%	62.98%	75.74%	65.41%	79.67%	90.53%
	Std.	4.12%	3.16%	2.70%	4.82%	2.71%	1.87%

Table 31 presents the value of considering pre-committed tasks, i.e., the gap between the allocation yielded by the mechanism and the allocation obtained from a baseline heuristic. As shown, Stage B significantly increases the value of cooperation for all methods and in particular for the heuristic allocation. Therefore, the value of considering pre-committed tasks significantly decreases. The average values of considering pre-committed tasks and their deviations are presented in Table 31.

Table 31: Average values considering pre-committed tasks

		$\alpha = 1$			$\alpha = 3$		
		Stage A	Stage B - no money transfers	Stage B - with money transfers	Stage A	Stage B - no money transfers	Stage B - with money transfers
Vickrey auctions	Average	198.85%	122.5%	102.86%	164.93%	115.73%	101.83%
	Std.	18.66%	7.33%	1.71%	25.47%	4.00%	2.41%
Sequential combinatorial auctions	Average	208.39%	128.94%	105.37%	170.72%	117.52%	102.94%
	Std.	17.85%	6.99%	2.25%	29.45%	3.75%	2.15%
Vickrey auctions with small clusters	Average	169.83%	113.94%	101.17%	137.65%	101.29%	93.67%
	Std.	12.00%	6.15%	2.18%	19.79%	3.75%	2.08%
Sequential negotiations with low initial prizes	Average	206.15%	127.48%	104.92%	166.71%	115.81%	102.09%
	Std.	18.37%	6.81%	1.84%	28.12%	3.82%	2.20%
Sequential negotiations with high initial prizes	Average	187.45%	116.12%	102.94%	149.74%	106.35%	99.23%
	Std.	17.25%	6.69%	2.20%	26.62%	4.67%	1.95%

Stage B with money transfers eliminates large share of the value of considering pre-committed tasks in Stage A. This implies that a viable protocol is to apply the heuristics at Stage A and then apply Stage B with money transfers. Such an approach will yield an allocation that is only slightly less efficient than the ones obtained by the other alternatives and it is much easier to implement.

The value of cooperation is distributed in a way that generally benefits the contractors. In some cases, the company is even worse off compared to the default status. Average values of the distribution of the value of cooperation between contractors and the company after Stage A and their standard deviations are presented in Table 32.

Table 32: Distribution of value of cooperation between contractors (after Stage A)

		$\alpha = 1$		$\alpha = 3$	
		company	contractors	company	contractors
Vickrey auctions	Average	-2.88%	102.88%	-49.91%	149.91%
	Std.	8.43%		9.78%	
Sequential combinatorial auctions	Average	25.37%	74.63%	-1.83%	101.83%
	Std.	3.19%		4.01%	
Vickrey auctions with small clusters	Average	33.59%	66.41%	7.03%	92.97%
	Std.	4.75%		6.51%	
Sequential negotiations with low initial prizes	Average	85.78%	14.22%	86.7%	13.3%
	Std.	0.78%		0.83%	
Sequential negotiations with high initial prizes	Average	30.08%	69.92%	26.1%	73.9%
	Std.	7.89%		11.42%	

In vickrey auction the company is significantly worse off (compare to the default protocol) while the contractors are much better off. It is likely that this cannot be a stable solution.

6 Conclusions and future research

The Decentralized Field Service Routing Problem addresses a company that should serve a set of tasks. The company is interested in providing this service to its clients by outsourcing the tasks to contractors at a smallest cost. The contractors are interested in maximizing their net profit.

For solving the decentralized problem, a solution mechanism, which suggests good solutions both from the society's point of view and from all the agents' points of view, is required.

6.1 Conclusions

We adopt a general framework of a 2-stage task allocation mechanism for the DFSRP. In the first stage, an initial allocation of service tasks to contractors is generated, and in the second, the contractors trade tasks between themselves.

Three of the procedures used in the first stage, namely, sequential Vickrey auctions, sequential combinatorial auctions and the sequential negotiation protocol yielded fairly good allocations from the society's point of view. That is, the cost of the generated allocations' is close to the optimal cost that could be obtained by a central planner. The sequential combinatorial auction mechanism is the first application of the GVA in the area of field service routing. To the best of our knowledge this is the first strategy proof mechanism used to solve a decentralized routing problem.

The Vickrey auctions are shown to be efficient and can be calculated easily. However, this mechanism is unprofitable from the company's point of view and thus unlikely to be implemented in practice. At the same time, the Vickrey auctions with small clusters are shown to be profitable but relatively inefficient.

Unprofitability from the company's point of view is a characteristic of the generated solution that is evident when comparing the sum of rewards the company paid to the optimal cost of serving all tasks by herself. However, recall that in this work the cost was estimated quite conservatively by assuming the company has the same cost structure as the contractors. Furthermore, the option of serving all tasks solely is usually not available for the companies. Note that outsourcing is often applied as a method that deals with a large variety in the required workload. Outsourcing is an alternative to employing a large enough service team on a regular basis which may result in large operational costs and low utilization of the team when the workload is less than maximal.

The typical alternative to outsourcing based on a collaborative mechanism (such as the ones proposed in this study) is outsourcing using simple heuristic methods for the allocation of the tasks. The heuristic solution is generally far from optimal and causes companies to transfer higher payments to the contractors that compensate for the inefficient solution. Note that the implementation of combinatorial auctions in real life in the area of freight transportation services has proven to save millions of dollars (see Porter (2002)). We believe that this can be the case for field service routing and scheduling as well.

However, if for some reason, combinatorial auctions (or other collaborative mechanisms) cannot be implemented, the solution generated by simple heuristics can be improved significantly by applying the second stage of the proposed mechanism, in which contractors are able to exchange tasks among themselves. Even though Stage B may encourage a speculative behavior of the agents, no party can be worse off due to the outcome of this stage and it enables increasing the profits of some parties.

Recall that a realistic setting of the DFSRP is likely to include a larger number of contractors or a larger number of tasks when compared to the instances presented in this thesis. More than often, both are larger. With respect to the sealed bid allocation mechanisms, we believe that the larger the number of contractors is, the payment made by the company to the contractors is smaller. Note that this payment is based on the contribution of each contractor to the system which is likely to be smaller as more contractors are included. Additionally, a larger number of contractors further reduces the ability of a single contractor to apply speculative considerations in the sequential negotiation procedure. That is, all allocation procedures are likely to be more sustainable as the number of contractors increases. A larger number of contractors is also likely to enlarge the number of relatively efficient solutions. In this context, a larger number of tasks has the same effect, although it may cause the various optimization problems to be significantly more involved.

Hence, the computational complexity of the various mechanisms should be considered when deciding on the appropriate one. For example, although the Vickrey auction on small clusters yields inefficient solutions, it can be carried out quite easily and is profitable, even under our conservative assumptions, for all parties. Its relative computational simplicity may make this mechanism more appealing for all parties, than, say, the combinatorial auctions. The latter yield efficient solutions but require a very involved procedure of calculating an exponential number of bids.

6.2 Future research

The decentralized problem stated in this work does not consider time windows in which tasks should begin and end and does not consider possible differences in the skills of the contractors. However, these are very common in real life problems. Therefore, a natural direction for future research is to include these features.

Additional characteristics of the field scheduling problem such as state regulation, labor union agreements etc. are also to be considered.

The problem formulated and solved in this thesis suggests that a single client (the company) purchases services from multiple suppliers (the contractors). Another possible configuration consists of multiple clients, that is different companies, that are interested in purchasing some services from the contractors. We believe the decentralized problem implied by the multi-client configuration can also be solved using the mechanisms suggested in this work. For that end, these need to be adapted. The most important adaptation is finding a method to determine the payments of each client to the contractors. An even more general setting of the problem should deal with multiple clients that are the customers themselves that are interested in choosing their service provider.

The DFSRP studied in this thesis is a static problem where all the tasks are known in advance. In practice, information about the required tasks is received dynamically over time. Therefore, a great value can be obtained from formulating and solving the dynamic version of the problem. Clearly, this is a much more involved problem.

6.2.1 Future research – Stage A

Several allocation procedures for Stage A have been presented in this work. In the sequential combinatorial auctions, the proposed procedure groups the tasks into clusters and then each cluster's tasks are announced in a combinatorial auction. Clustering is a reasonable way by which computational complexity of the problem can be reduced.

Additional work can be done as to increase the size of the cluster even further, and thus enabling a solution closer to optimum. This requires reducing the running time of the calculation of the bids by several orders of magnitudes. A reduction of this sort can be achieved by calculating the bids using heuristic methods instead of by integer programming. However, an increase in the cluster's size makes the winner determination problem, which is NP-Hard, extremely hard to solve. In this case, a solution of the problem by carefully tailored algorithms is recommended (see for example Sandholm (2002)).

Furthermore, recall that the number of clusters reduces as the size of each cluster increases. Further work exploring the relation between the number of clusters and the optimality of the allocations is desirable. Note that the high efficiency generated by applying the Vickrey auction, which is a combinatorial auction done on clusters with size 1, indicates that this relation may not be trivial.

The characterization of clusters may affect the optimality of the generated solution. This thesis suggests that each cluster should be homogeneous in the locations of its members. However, homogeneousness of this sort does not necessarily contribute to creating the best allocations. Thus, further research employing the sequential combinatorial auctions with non-homogenous clusters is recommended. This will enable us to find the clusters with the most suitable properties for the mechanism.

For the sequential negotiation procedure, additional research is required in order to find the optimal strategies of the contractors when speculations are considered. Next, a comparison between the contractors' profits when following their optimal strategies and their profits following myopic strategies can be helpful in gaining insights regarding the usefulness of the no-speculation assumption that is the base of the sequential negotiation mechanisms.

6.2.2 Future research – stage B

In Stage B, we implemented a task exchange procedure in which the contractors are required to consider all possible exchanges (of size 1) of company's tasks they own with company's tasks they do not own. Applying various algorithmic considerations may save the need to consider most of the possible exchanges and therefore reduce the running time significantly. Additionally, Stage B can be generalized as to include the exchange of more than one task at a time. This will increase the number of possible exchanges and therefore stresses the importance of applying some algorithmic considerations to speed-up the computations.

The value sharing mechanism proposed for Stage B does not ensure truthful biddings. Thus, a further goal is to design a strategy proof value sharing mechanism. However, a mechanism not requiring external subsidy is recommended. Nonetheless, if a strategy proof mechanism cannot be devised, one may consider an adapted version of Stage B where unprofitable exchanges are not allowed. Since the net profit from each exchange would be non-negative, this version does not require any money transfers and thus, cancels the need to design a value sharing mechanism. However, it may yield less efficient solutions.

References

[1] L.M. Ausubul and P. Milgrom. The lovely but lonely Vickrey auction. In: P. Cramton et al. (Eds.) *Combinatorial Auctions*, Cambridge, MA, 57–90, 2006.

[2] A. Bachem, W. Hochstättler and M. Malich. The simulated trading heuristic for solving vehicle routing problems. *Discrete Applied Mathematics*, 65:47-72, 1996.

[3] I. Beniaminy, D. Yelin, U. Zahavi and M. Zardin. When the rubber meets the road: Bio-inspired field service scheduling in the real world. In: F.B. Pereira and J. Tavares (Eds.) *Bio-inspired Algorithms for the Vehicle Routing Problem*, SCI 161: 191-213, Springer, Heidelberg, 2009.

[4] C. Boutilier, M. Goldszmidt, B. and Sabata. Sequential auctions for the allocation of resources with complementarities. *Proc. IJCAI-99, Stockholm, Sweden,* pages 527–534, 1999.

[5] C. Caplice and Y. Sheffi. Combinatorial Auctions for Truckload Transportation. In: P. Cramton et al. (Eds.) *Combinatorial Auctions*, Cambridge, MA, 1016-1069, 2006.

[6] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11: 17-33, 1971.

[7] T. Groves. Incentives in teams. *Econometrica*, 41: 617-631, 1973.

[8] E. Hadjiconstantinou and D. Roberts. Routing under uncertainty: an application in the scheduling of field service engineers. In: P. Toth and D. Vigo. (Eds.) *The vehicle routing problem*, SIAM Monographs on Discrete Mathematics and Applications: 331–352, Philadelphia, 2002.

[9] G.Q. Haung and S.X. Xu. Truthful multi-unit transportation procurement auctions for logistics e-marketplaces. *Transportation Research Part B*, 47(1): 127-148, 2013.

[10] G.Q. Haung and S.X. Xu. Efficient auctions for distributed transportation procurement. *Transportation Research Part B* under second-round review, 2014.

[11] R. Kohout and K. Erol. In-time agent-based vehicle routing with a stochastic improvement heuristic. *Eleventh Conference on Innovative Applications of Artificial Intelligence, Orlando, FL*, 1999.

[12] E. Koutsoupias and CH. Papadimitriou. Worst-case equilibria. *16th International Symposium on Theoretical Aspects of Computer Science,* pages 404-413, 1999.

[13] J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7: 48-

50, 1956.

[14] H.W. Leong and M. Liu. A multi-agent algorithm for vehicle routing problem with time window. *Proceedings of the SAC 2006, Dijon France*, pages 23-27, 2006.

[15] P. Milgrom. Putting Auction Theory to Work: The Simultaneous Ascending Auction. *Journal of Political Economy*, 108(2): 245-272, 2000.

[16] C.E. Miller, A.W. Tucker and R.A. Zemlin. Integer programming formulations and traveling salesman problems. J. Association for computing machinery, 7: 326-329, 1960.

[17] J.F. Nash. The Bargaining Problem. *Econometrica*, 18: 155-162, 1950.

[18] D. Parkes. Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency. PhD Thesis, University of Pennsylvania, 2001.

[19] D. Porter, D.P. Torma, J. O. Ledyard, J.A. Swanson and M. Olson. The First Use of a Combined-Value Auction for Transportation Services. *Interfaces*, 32(5):4-12, 2002.

[20] S. J. Rassenti, V.L. Smith and L. Bulfin. A combinatorial auction mechanism for airport time slot allocation. *Bell Journal of Economics*, 13: 402-407, 1982.

[21] M. Rothkopf. Thirteen reasons the Vickrey–Clarke–Groves process is not practical. *Operations Research*, 55(2): 191-197, 2007.

[22] T. Sandholm. An implementation of the contract net protocol based on marginal cost calculations. *Proc. AAAI-93, Washington, DC*, pages 256–262, 1993.

[23] T. Sandholm. Issues in computational Vickrey auctions. *International journal of electronic commerce*, 4: 107-129, 2000.

[24] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135: 1-54, 2002.

[25] R.G. Smith. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Trans. on Computers*, C-29(12):1104-1113, 1980.

[26] M.M. Solomon, Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2): 254-265, 1987.

[27] S. R. Thangiah, O. Shmygelska and W. Mennell. An agent architecture for vehicle routing problems. *Proceedings of SAC 2001, Las Vegas, NV*, pages 517-521, 2001.

[28] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders.

Journal of Finance, 16: 8-37, 1961.

[29] J. Xu and S.Y. Chiu. Effective Heuristic Procedures for a Field Technician Scheduling Problem. *Journal of Heuristics*, 7: 495-509, 2001.

[30] S. Zamir , M. Maschler, and E. Solan. *Game Theory*, Magnes University Press Jerusalem, Israel, 2008.

[31] M. Zerdin, A. Gibrekhterman, U. Zahavi and D. Yellin. Optimization strategies for Restricted Candidate Lists in Field Service Scheduling. In: M. Koppen et al. (Eds.) *Intelligent Computational Optimization in Engineering*, SCI 366: 55-83, Springer, Heidelberg, 2011.

[32] D. Zhenggang, C. Linning and L. Zheng, Improved multi-agent system for the vehicle routing problem with time windows, *Tsinghua Science Technology*, 14(3): 407-412, 2009.

תקציר

חברות גדולות נוטות לבצע מיקור חוץ של משימות השירות שלהן בבית הלקוח לקבלנים מומחים קטנים יותר. הקבלנים הינם ישויות עסקיות עצמאיות מהחברות המקצות להן משימות. לכן, שיתוף של מידע פרטי בין הצדדים השונים אינו אפשרי. כתוצאה מכך, הפרקטיקה הנהוגה במציאות היא הקצאה של משימות לקבלנים שמבוצעת על ידי החברה על בסיס כללים היוריסטיים כאלו ואחרים, דוגמת חלוקה לאזורים. הקצאה היוריסטית מהסוג הזה מתעלמת לחלוטין ממשימות של חברות אחרות אשר הוקצו לקבלנים בשלב קודם. כך מתקבלת הקצאת משימות שאינה יעילה. במחקר זה מפותח אלגוריתם שיתופי מבוזר דו שלבי אשר מתמודד עם הבעיה ופותר אותה באופן קרוב לאופטימלי.

השלב הראשון של האלגוריתם מיועד ליצירת הקצאה ראשונית, לא בהכרח אופטימלית של המשימות לקבלנים. במסגרת המחקר, מימשנו כמה פרוצדורות להקצאת המשימות, וביניהן מכרזי ויקרי סדרתיים, מכרזים קומבינטוריים סדרתיים וכן מו״מ סדרתיים. הפרוצדורה של המכרזים הקומבינטוריים הסדרתיים מממשת את מנגנון ויקרי המוכלל (Generalized Vickrey auction) אשר פותח בשנות ה-60 וה-70. אחד מהיישומים של מנגנון זה הינו פתרון בעיית מקסום סך ערך ההקצאה של פריטים רבים (ושונים) בקרב סוכנים מתחרים בעלי פונקציות ערך שונות. הייחודיות במנגנון זה הינה שיישומו בבעיה גורם לכך שאמירת אמת מצד הסוכנים לגבי העדפותיהם הינה אסטרטגיה שלטת במשחק הנוצר.

בשלב השני, הקבלנים רשאים להחליף משימות בינם לבין עצמם על מנת להפחית את עלויותיהם. לשלב זה הוגדרו ומומשו שתי גרסאות במחקר זה. האחת, כשהעברות כספים מותרות, והאחרת – כשהעברות כספים אינן מותרות.

הקצאת המשימות שהמנגנון מוליד נמדדת באמצעות מספר מדדי ביצוע. מצאנו כי פרוצדורות ההקצאה הממומשות בשלב הראשון מביאות להקצאה יעילה יחסית של המשימות לקבלנים וכי השלב השני אף משפר הקצאה זו. ההקצאה המתקבלת באמצעות המנגנון היא יעילה הרבה יותר בהשוואה להקצאה המתקבלת מיישום היוריסטיקת מדף סבירה. למעשה, עלות ההקצאה המתקבלת קרובה מאוד לגבול התחתון של עלות ההקצאה האופטימלית. כלומר, עלות חוסר הסדר הינה קטנה מאוד.

אוניברסיטת תל אביב
הפקולטה להנדסה ע״ש איבי ואלדר פליישמן
בית הספר לתארים מתקדמים ע״ש זנדמן-סליינר

בעיית ניתוב כלי הרכב המבוזרת

חיבור זה הוגש כעבודת גמר לקראת התואר ״מוסמך אוניברסיטה״
בהנדסת תעשייה

ע״י

אדיסון אברהם

העבודה נעשתה במחלקה להנדסת תעשייה
תחת הנחייתם של ד״ר טל רביב ופרופ׳ יבגני חמלניצקי

אב תשע״ד

אוניברסיטת תל אביב
הפקולטה להנדסה ע״ש איבי ואלדר פליישמן
בית הספר לתארים מתקדמים ע״ש זנדמן-סליינר

בעיית ניתוב כלי הרכב המבוזרת

חיבור זה הוגש כעבודת גמר לקראת התואר ״מוסמך אוניברסיטה״
בהנדסת תעשייה

ע״י

אדיסון אברהם

אב תשע״ד