ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός Μηχανισμών με Επαλήθευση

Διπλωματική Εργασία

Εμμανουήλ Β. Ζαμπετάκης

Επιβλέπων Καθηγητής
Δημήτρης Φωτάκης
Επίκουρος Καθηγητής Ε.Μ.Π.

ΕΡΓΑΣΤΗΡΙΟ ΛΟΓΙΚΗΣ ΚΑΙ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ
Αθήνα, Φεβρουάριος 2014

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων Τομεας
Πληροφορικης και Υπολογιστων

Σχεδιασμός Μηχανισμών με Επαλήθευση

Διπλωματική Εργασία

Εμμανουήλ Β. Ζαμπετάκης

Επιβλέπων Καθηγητής
Δημήτρης Φωτάκης
Επίκουρος Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή επιτροπή την 19η Φεβρουαρίου 2014

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
....	....	....
Στάθης Ζάχος	Άρης Παγουρτζής	Δημήτρης Φωτάκης
Καθηγητής Ε.Μ.Π.	Επίκ. Καθηγητής Ε.Μ.Π.	Επίκ. Καθηγητής Ε.Μ.Π.

Εργαστηριο Λογικης και Επιστημης Υπολογιστων
Αθήνα, Φεβρουάριος 2014

(Υπογραφή)

....

Εμμανουηλ Ζαμπετακης

Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων Τομεας
Πληροφορικης και Υπολογιστων

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Ευχαριστίες

Κατά την εκπόνηση της διπλωματικής μου εργασίας είχα την τύχη να συνεργαστώ με πολλούς ιδιαίτερα αξιόλογους ανθρώπους τους οποίους θέλω ευχαριστήσω θερμά!

Αρχικά θα ήθελα να ευχαριστήσω τους καθηγητές του Εργαστηρίου Λογικής και Επιστήμης Υπολογιστών κ. Ζάχο, κ. Παγουρτζή και κ. Φωτάκη για την υποστήριξη τους καθώς και για την καθοδήγηση που μου προσέφεραν και συνεχίζουν να μου προσφέρουν σε επιστημονικό, ακαδημαϊκό αλλά και προσωπικό επίπεδο.

Ιδιαίτερα θέλω να ευχαριστήσω τον κ. Φωτάκη που ήταν επιβλέπων στην εργασία αυτή και που χάρη στις πολύτιμες συμβουλές του μπορέσαμε μαζί να ολοκληρώσουμε με επιτυχία τον αρχικό στόχο της εργασίας αυτής και να δημοσιεύσουμε τα αποτελέσματα της στο συνέδριο Conference on Web and Internet Economics που έγινε τον Δεκέμβριο του 2013 στη Βοστώνη. Η καθοδήγηση του είναι ανεκτίμητη και θα με συνοδεύει σίγουρα σε όλα τα επόμενα μου ακαδημαϊκά και μη βήματα.

Το περιβάλλον του Εργαστηρίου Λογικής και Επιστήμης Υπολογιστών είναι από τα πιο υγιή και αξιόλογα πράγματα που συνάντησα στο Πολυτεχνείο. Τόσο η διάθεση για συνεργασία και συζήτηση επί των επιστημονικών θεμάτων όσο και η φιλική ατμόσφαιρα προσφέρουν ένα περιβάλλον ιδανικό τόσο για εκπόνηση εργασιών όσο και για καθημερινή συναναστροφή. Θα ήθελα λοιπόν να ευχαριστήσω θερμά όλους τους υπεύθυνους για την δημιουργία αυτού του περιβάλλοντος που εκτός από τους καθηγητές είναι τα μέλη του εργαστηρίου. Τους ευχαριστώ θέρμα για τις συμβουλές που μου δώσανε, για τις συζητήσεις που είχαμε καθώς και για την καθημερινή τους υποστήριξη. Διαλέγοντας μια τυχαία σειρά και προσπαθώντας να μην ξεχάσω κανέναν σας ευχαριστώ πολύ Θανάση, Σωτήρη, Μάρκο, Αντώνη, Γιώργο, Λυδία, Βασίλη, Φώτη, Ανδρέα, Ελένη, Θοδωρή, Ναταλία, Βαγγέλη, Χριστίνα, Μάριε, Ματούλα, Μάκη, Κυριάκο, Μανώλη! Θέλω επίσης να ευχαριστήσω και του συμφητητές μου που δεν ανήκουν στο εργαστήριο και μου στάθηκαν κατά την διάρκεια των σπουδών μου! Ευχαριστώ Νικόλα, Ελένη, Γιώργο, Ηλία, Ανώνη, Γιάννη, Βασίλη, Ειρήνη, Γρηγόρη!

Τέλος θα ήθελα να ευχαριστήσω όλη μου την οικογένεια, τους οποίους δεν αναφέρω ονομαστικά γιατί είναι πολύ (ναι περισσότεροι από τα μέλη του εργαστηρίου!), για την καθημερινή υποστήριξη, τις συζητήσεις αλλά και τις λύσεις που δώσανε σε όλη την διάρκεια των σπουδών μου αλλά και αυτής της εργασίας. Το να έχει κανείς ένα πολύ καλό και επιστημονικά και προσωπικά περιβάλλον στο εργαστήριο είναι κάτι που όπως είπα είναι απίστευτα σημαντικό. Το να συναντά όμως το ίδιο περιβάλλον και από προσωπική και από επιστημονική άποψη σπίτι του είναι πράγμα ανεκτίμητο. Ιδιαίτερα ευχαριστώ για τις καθημερινές επιστημονικές συζητήσεις τα αδέρφια μου Αλεξάνδρα και Κώστα καθώς, τους γονείς μου Βασίλη και Μαρία καθώς και τον θείο μου τον Φώτη. Επίσης ήθελα να ευχαριστήσω τους γονείς μου για έναν ακόμα λόγο, ίσονος σημασίας μπροστά στην βοήθεια και την υποστήριξη που μου προσφέρουν τα τελευταία χρόνια, για την βοήθεια τους στην επιλογή του τίτλου που δημοσιεύσαμε αυτήν την εργασία ('Truthfulness Flooded Domains...¨) στο συνέδριο που ανέφερα.

Για όλα τα παραπάνω και για πολλά άλλα που προφανώς ξεχνάω σας ευχαριστώ πολύ όλους!

Περίληψη

Η βασική θεωρία σχεδιασμού μηχανισμών υποθέτει ότι οι παίκτες μπορούν να ακολουθήσουν οποιαδήποτε από τις δυνατές στρατηγικές. Επομένως ο μηχανισμός δεν μπορεί να χρησημοποιήσει κάποια πληροφορία η οποία μπορεί να προσφέρεται εύκολα. Κάνοντας μια μικρή παραλαγή αυτού του μοντέλου μπορούμε να υποθέσουμε ότι οι δυνατές στρατηγικές για κάθε παίκτη είναι περιορισμένες και εξαρτώνται από την πραγματική κατάσταση του παίκτη. Σ' αυτην την εργασία μελετάμε τον τρόπο με τον οποίο αλλάζει ο σχεδιαμός μηχανισμών όταν υποθέτουμε αυτόν τον περιορισμό στις στρατηγικές που ονομάζουμε *επαλήθευση*.

Η πρώτη διαφορά που συναντάμε είναι ότι στον σχεδιασμό μηχανισμών με επαλήθευση εμφανίζεται η δυνατότητα υλοποίησης μη-φιλαλήθη συναρτήσεων κοινωνικής επιλογής. Παρουσιάζουμε τις αιτίες που συμβαίνει αυτό καθώς και τους λογούς που κάνουν αυτόν τον τρόπο υλοποίησης μη εφαρμόσιμο όταν απαιτούμε την ύπαρξη κυρίαρχης στρατηγικής. Ύστερα παρουσιάζουμε την δύναμη που αποκτάει αυτός ο τρόπος υλοποίησης όταν απαιτήσουμε ύπαρξη nash ισορροπίας.

Σ' αυτήν την εργασία επίσης μελετάμε τους λόγους που κάνουν την συμμετρική επαλήθευση μη χρήσιμη στην υλοποίηση συναρτήσεων κοινωνικής επιλογής. Για πρώτη φορά μελετάμε γενική συμμετρική επαλήθευση εφαργμοσμένη σε οποιδήποτε σύνολο δυνατών στρατηγικών. Επειδή η απλούστερη μορφή συμμετρικής επαλήθευσης είναι η τοπική επαλήθευση τα αποτελέσματα μας μπορύν να εφαρμοστούν στην μελέτη της σχέσης μεταξύ τοπικής και ολικής φιλαλήθειας.

Για να ολοκληρώσουμε την εικόνα της ανάλυσης εξετάζουμε και αποδεικνύουμε την δύναμη και την σημασία της μη συμμετρικής επαλήθευσης στην φιλαλήθη υλοποίηση συναρτήσεων κοινωνικής επιλογής.

Λέξεις Κλειδιά

Σχεδιασμός Μηχανισμός, Επαλήθευση, Υλοποίηση Συνάρτησης Κοινωνικής Επιλογής, Φιλαλήθεια, Τοπικό σε Γενικό

Abstract

The basic mechanism design model assumes that each agent can follow any of its strategies, independently of its type. Thus the mechanism cannot use any "real-world" information about the agents. This is the norm in mechanism design and it models well the negotiation stage in which agents do nothing but communicate. A simple type of modification to the model suggests itself: a problem definition may limit the set of strategies available to each agent as a function of its type. In this work we investigate the way mechanism design changes under the assumption of verification.

The first deference in the mechanism design with verification is the existence of non-truthful implementations of social choice functions. We present the reason why this way of implementation is not really helpful in the solution concept of dominant strategy equilibrium and we present a way it could be useful using the concept of Nash equilibrium.

In this work, we also investigate the reasons that make symmetric partial verification essentially useless in virtually all domains. Departing from previous work, we consider any possible (finite or infinite) domain and general symmetric verification. We identify a natural property, namely that the correspondence graph of a symmetric verification M is strongly connected by finite paths along which the preferences are consistent with the preferences at the endpoints, and prove that this property is sufficient for the equivalence of truthfulness and M -truthfulness. Since the simplest symmetric verification is the local verification, specific cases of our result are closely related, in the case without money, to the research about the equivalence of local truthfulness and global truthfulness.

To complete the picture, we consider asymmetric verification, and prove that a social choice function is M -truthfully implementable by some asymmetric verification M if and only if f does not admit a cycle of profitable deviations.

Key Words

Mechanism Design, Verification, Implementation of Social Choice Function, Truthfulness, Local to Global

Contents

Chapter 1

Introduction

An algorithm in computer science could be described as a rule which is based on some information in order to choose the *correct* solution from a set of possible solutions. Of course the term *correct* in the previous sentence is not well defined a priory and in every algorithmic problem could take a different meaning.

One basic assumption about algorithms is that all the information they need in order to choose the solution is always easily available. This assumption is a very reasonable in a lot of settings where the information that the algorithm needs it is saved in a safe place. But if one would try to apply an algorithm on a society then this assumption cannot hold because every person in the society has a part of the information that the algorithm needs. This creates a big problem to the implementation of the algorithm since some people could prefer not to tell the information they have or to lie about it because they don't like the result of the algorithm. Therefore when someone applies a rule on a society should also pay attention on the incentives of every single person in the society. In order to solve this complicated problem we need to use some tools to describe the behaviour of people who are called from now on *strategic agents* or just *agents*. This tools are available in the mathematical theory named as *game theory* and the description of the result of an algorithm when it is applied to strategic agents is called *mechanism desing*.

In mechanism design we assume that there is a principal who seeks to implement an algorithm which is based on some private information of some strategic agents. From now on we call this algorithm *social choice function*. More formally a social choice function is a function that maps the private information of the agents to a result which we call *outcome*. Exploiting their power over the outcome, the agents may lie about their private information if they find it profitable. Then the agent should change the social choice function to a *mechanism* which after taking into account the incentives of the agents finally implements the social choice function. The searching for such mechanism is the basic goal of mechanism design. As we will see from the very beginning of the mechanism design there are a lot of negative results implying that there in a very general setting there no way of implementing any non trivial social choice function. For this reason the principal may offer payments to (or collect payments from) the agents or find ways of partially verifying their statements, thus restricting the false statements available to them. A social choice function is then *implementable* if there is a payment scheme under which there is a mechanism which implements it. Since many social choice functions still are not implementable, a central research direction in mechanism design is to identify sufficient and necessary conditions under which large classes of functions are implementable.

In this thesis we seek a deeper understanding of the power of partial verification in mechanism design both in the case where also money are available and in the case not. This is a question going back to the work of Green and Laffont [10].

In this chapter we give the basic definitions and the preliminary results that we need for the next chapters. In the Chapter 2 we introduce the concept of *nontruthful implementations* and we present some already known results and some new results on this concept. Afterwards we seek our attention into *truthful implementations* of social choice functions. In Chapter 3 we explore the power of *symmetric verification* in a very general framework. In Chapter 4 we present the work of Archer and Kleinberg [2] which gives a local to global characterization of truthfulness and it is a stronger version of the results presented in Chapter **??** but in a more restricted setting. In Chapter **??** we give some results and some deeper explanation on the very useful tool of *asymmetric* verification which has a lot of applications. In Chapter 6 we present a new model for verification produced by Caragiannis et. al. [6] where the verification is not a deterministic process but a random process. Finally we give some conclusions and we present some open and interesting problems based on this work.

1.1 Model and Notation

We start with a more formal description of our model. We restate that in the mechanism design context we have a set of agents with private inputs and the principal which is the designer of the mechanism and wants to *implement* a social choice function f on the agents' input. So if we assume that D is the set of possible inputs of agents and O the set of possible outcomes of the mechanism then the principal wants to implement a function $f : D^n \to O$ where n is the number of agents.

It is known from the discussion in [2] that when proving theorems for mechanism design we could suppose, without loss of generality, that we have only one agent and the principal wants to implement a function $f : D \to O$. If money are available then the principal has the opportunity to use a payment function $p : D \to \mathbb{R}$ in order to implement his social choice function.

In this context we give the following definitions.

- there is a set O of the set of possible *outcomes* of the mechanism

- each agent has a private information about the outcomes, abstracted as a *valuation function* or *type* $u : O \to \mathbb{R}$
- the set of all types is the domain $D = \mathbb{R}^O$
- the principal wants to implement a social choice function $f : D \to O$
- when money are available principal also defines a payment function $p : D \to \mathbb{R}$

Now we are now ready to define notion of *implementability* of a social choice function. Intuitively we call a social choice function f implementable if there is a mechanism g such that if we apply g to the input then after the strategic behaviour of the agent the result is the same as the result of f without the strategic behaviour of the agent. To examine the strategic behaviour of the agent we use from game theory the solution concept of *dominant strategy equilibrium.* Of course the existence of the dominant strategy is not always guaranteed as the existence of mixed Nash equilibrium in finite games. But the dominant strategy equilibrium is the most robust solution concept of game theory and gives as a very strong evidence that the mechanism is going to work in practice. Therefore in order to say that g implements f the agent has to have a dominant strategy on the mechanism g and the result in this dominant strategy has to be the same as the result of f in the case where the agent has not a strategic behaviour.

Definition 1 (Implementation without money). *A mechanism $g : D \to O$ – is said to implement a social choice function f if for each $u \in D$ there exists a $v \in D$ such that :*

$$g(v) = f(u)$$

$$u(g(v)) \geq u(g(u')) \quad \forall u' \in D$$

A social choice function f is said implementable if there exists a mechanism g that implements it.

If the principal is also allowed to use money then there is also a payment function $p : D \mapsto \mathbb{R}$ which also changes the utility of the agent. Throughout this work we assume that the agent has quasilinear utility which means that his utility is equal to the valuation plus the payments. In this setting the notion of implementability changes as follows.

Definition 2 (Implementation with money). *A mechanism $g : D \to O$ – together with a payment function $p : D \to \mathbb{R}$ is said to implement with money a social choice function f if for each $u \in D$ there exists a $v \in D$ such that :*

$$g(v) = f(u)$$

$$u(g(v)) + p(v) \geq u(g(u')) + p(u') \quad \forall u' \in D$$

A social choice function f is said implementable with money if there exists a mechanism g and a payment function p that truthfully implements it.

From the very first steps of mechanism design the researchers understood that the requirement of the existence of some dominant strategy equilibrium in this setting is equivalent with the requirement of the existence of a very special dominant strategy equilibrium. Namely they understood that if there exists a mechanism that implements a social choice function then the agent has no reason to misreport from the beginning and therefore the social choice function it can also be used as a mechanism to implement it self. In this case the agent has reason to misreport and it is a dominant strategy for him to tell the truth. The condition that the truthful report is a dominant strategy it is the central concept of mechanism design and it is well known as *truthfulness.* We are now ready to define *truthful implementation* of a social choice function.

Definition 3 (Truthful Implementation without money)**.** *A mechanism – social choice function $f : D \to O$ – is said to be truthfully implementable or truthful if for each $u \in D$:*

$$u(f(u)) \geq u(f(u')) \quad \forall u' \in D$$

In the case money are available to principal we have the notion of *truthfulness with money.*

Definition 4 (Truthful Implementation with money)**.** *A mechanism – social choice function $f : D \to O$ is said to be truthfully implementable with money or truthful with money if there is a payment function $p : D \to \mathbb{R}$ such that for each $u \in D$:*

$$u(f(u)) + p(u) \geq u(f(u')) + p(u') \quad \forall u' \in D$$

The first basic result in mechanism design is the following theorem which is also known as *revelation principle.* We will present a proof of this theorem for the case without money next in this section when we will introduce the graph representation of mechanism design.

Revelation Principle. *A social choice function f is implementable (with money) if and only if it is truthfully implementable (with money).*

1.1.1 Partial Verification

The basic mechanism design model assumes that each agent can follow any of its strategies, in- dependently of its type. Thus the mechanism cannot use any "real-world" information about the agents. This is the norm in mechanism design and it models well the negotiation stage in which agents do nothing but communicate. In many settings in distributed computation though, one could take advantage of the fact that computers actually act (execute a task, route a message, etc.) to gain extra information about the agents' types and actions. A simple type of modification to the model suggests itself: a problem definition may limit the set of strategies available to each agent as a function of its type. More formally in the usual mechanism design setting, an agent with type u is allowed to report any other type $v \in D$. In the *partial verification model* the types that the agent is allowed to report is limited and may depend on u [**?**].

Definition 5. *A misreport correspondence is a function $M : D \to 2^D$, which for each type u specifies the set of types $M(u) \subseteq D$ the agent is allowed to report.*

Assuming that every agent with type x could report a type in the set $M(x)$ the notions of implementation and truthful implementation with verification changes as follows.

Definition 6 ($M-$Implementation without money)**.** *A mechanism $g : D \to O$ – is said to $M-$implement a social choice function f if for each $u \in D$ there exists a $v \in M(u)$ such that :*

$$g(v) = f(u)$$

$$u(g(v)) \geq u(g(u')) \quad \forall u' \in M(u)$$

A social choice function f is said $M-$implementable if there exists a mechanism g that implements it.

If the principal is also allowed to use money then there is also a payment functionConvex Domain. A domain D is *convex* if for every $x, y \in D$ and any $\lambda \in [0, 1]$, the function $z : O \mapsto \mathbb{R}$, with $z(a) = \lambda x(a) + (1 - \lambda)y(a)$, for each $a \in O$, is also in D. $p : D \mapsto \mathbb{R}$ which also changes the utility of the agent. Throughout this work we assume that the agent has quasilinear utility which means that his utility is equal to the valuation plus the payments. In this setting the notion of implementability changes as follows.

Definition 7 ($M-$Implementation with money)**.** *A mechanism $g : D \to O$ – together with a payment function $p : D \to \mathbb{R}$ is said to $M-$implement with money a social choice function f if for each $u \in D$ there exists a $v \in M(u)$ such that :*

$$g(v) = f(u)$$

$$u(g(v)) + p(v) \geq u(g(u')) + p(u') \quad \forall u' \in M(u)$$

A social choice function f is said $M-$implementable with money if there exists a mechanism g and a payment function p that truthfully implements it.

Again a central role in our analysis plays the implementation with every player reporting truthfully in the dominant strategy equilibrium.

Definition 8 (Truthful $M-$Implementation without money)**.** *A mechanism – social choice function $f : D \to O$ – is said to be truthfully $M-$implementable or $M-$truthful if for each $u \in D$:*

$$u(f(u)) \geq u(f(u')) \quad \forall u' \in M(u)$$

Definition 9 (Truthful $M-$Implementation with money)**.** *A mechanism – social choice function $f : D \to O$ is said to be $M-$truthfully implementable with money or $M-$truthful with money if there is a payment function $p : D \to \mathbb{R}$ such that for each $u \in D$:*

$$u(f(u)) + p(u) \geq u(f(u')) + p(u') \quad \forall u' \in M(u)$$

In the first work where the partial verification is examined Green and Lafford [?] notice that the revelation principle no more holds when partial verification is available. Moreover the examine sufficient conditions which the correspondence function M has to satisfy in order to have an equivalence between the notions of $M-$implementability and truthful $M-$implementability. From the fact that revelation principle doesn't hold we can understand that with some kinds of partial verification there is a way of implementing social choice functions *non-truthfully*. We present some attempts to understand the power of non-truthful implementations in Chapter 2. After understanding the difficultly of non-truthful implementations we turn our attention to truthful implementations again in the next chapters.

We now introduce a nice representation of mechanism design, using graph theory, which we call *graph representation* of mechanism design and it will became very useful in the next chapter when we analyse the power of verification. The reason is that the conditions under which the verification is or it's not powerful have a very nice formulation in terms of the graph representation.

1.2 Graph representation

Gui, Müller, and Vohra [11] cast the setting of mechanism design in terms of a (possibly infinite) directed graph G on vertex set D. For each ordered pair of types x, y, G has a directed edge (x, y). Given the social choice function f, we obtain an edge-weighted version of G, denoted G_f, where the weight of each edge (x, y) is $x(o) - x(o')$, with $o = f(x)$ and $o' = f(y)$. This corresponds to the gain of the agent if instead of misreporting y, she reports his true type x. Then, a social choice function f is truthfully implementable if and only if G_f does not contain any negative edges. Moreover, Rochet's theorem [16] implies that a function f is truthfully implementable with money if and only if G_f does not contain any directed negative cycles (see also [19]). More formally we have the following definition for the weighted graph G_f.

Definition 10. *For a given social choice function f we define the weighted graph G_f*

$$G_f = (D, D^2, w) \text{ where } w((x, y)) = x(f(x)) - x(f(y))$$

As we analysed in the previous section, there are many classical impossibility results stating that natural social choice functions (or large classes of them) are not implementable, even with the use of money (see e.g., [13]). Virtually all such proofs seem to crucially exploit that the agent can declare any type in the domain. Hence, Nisan and Ronen [14] suggested that the class of implementable functions could be enriched if we assume *partial verification* [10], which restricts the types that the agent can misreport.

One of the reasons that makes the graph representation very useful in our work is that partial verification comes as a very reasonable extension of the mechanism design represented with G_f, namely instead of assuming that G_f is a complete graph we assume that it is an arbitrary graph and this has an one-to-one and onto correspondence with the assumption of partial verification. As before, we can cast M as a (possibly infinite) directed *correspondence graph* G_M on D. For each ordered pair of types x, y, G_M has a directed edge (x, y) if $y \in M(x)$. Given the social choice function f, we obtain the edge-weighted version $G_{M,f}$ of G_M by letting the edge weights be as in G_f. A social choice function f is truthfully $M-$implementable (resp. with money) if and only if $G_{M,f}$ does not contain any negative edges (resp. directed negative cycles). More formally we have the following definitions for the graph G_M and the weighted graph $G_{M,f}$.

Definition 11. *For a given correspondence M we define the correspondence graph G_M*

$$G_M = (D, \{(x, y) \mid y \in M(x)\})$$

Definition 12. *For a given correspondence M and a given social choice function f we define the weighted graph $G_{M,f}$*

$$G_{M,f} = (D, \{(x, y) \mid y \in M(x)\}, w) \text{ where } w((x, y)) = x(f(x)) - x(f(y))$$

A k-cycle (resp. k-path) in G_M is a directed cycle (resp. path) consisting of k edges.

We say that an edge (x, y) of $G_{M,f}$ is negative if $w(x, y) < 0$. We say that a cycle in $G_{M,f}$ is negative if the total weight of its edges is negative. We let $G^-_{M,f}$ (resp. $G^+_{M,f}$) denote the subgraph of $G_{M,f}$ that consists of all its negative (resp. non-negative) edges.

If there is no verification, we refer to $G_{D,f}$, $G^+_{D,f}$, and $G^-_{D,f}$ as G_f, G^+_f, and G^-_f, respectively. Also, given a gWe say that a verification M is symmetric if G_M is *symmetric*, i.e., for each directed edge $(x, y) \in E(G_M)$, $(y, x) \in E(G_M)$. We say that a verification M is *asymmetric* if G_M is an acyclic tournament.

raph G, we let $V(G)$ be its vertex set and $E(G)$ be its edge set.

A negative edge (x, y) in $G_{M,f}$ represents that misreporting y is a profitable deviation of type x under f. Thus, a social choice function f is M-truthfully implementable if and only if $G_{M,f}$ does not contain any negative edges.

As we have already said the truthfulness conditions in terms of graph representation have a very nice inteprentation which we give formally now. The theorem that gives the implementability without money its obvious and has no explanation of proof. On the other hand the theorem that gives the implementability with money it's completely not obvious and we give here a proof of it.

Theorem 1. *A social choice function f is truthfully $M-$implementable if and only if the graph $G_{M,f}$ has no negative edge.*

Before presenting the theorem that relates the truthful implementation with the graph representation we give some definitions which helps to state the theorem and giving the proof.

Definition 13. *A function f satisfies M-*cycle monotonicity *if for all $k \geq 1$, and all $x_1, \dots, x_k \in D$, such that $x_{i+1} \in M(x_i)$*

$$\sum_{i=1}^{k} x_i(f(x_i)) \geq \sum_{i=1}^{k} x_{i-1}(f(x_i))$$

where the subscripts are modulo k. Equivalently, f is $M-$cyclic monotone if and only if $G_{M,f}$ does not contain any finite negative cycles.

If there is no verification and f has these properties, we simply say that f is cyclic monotone.

Theorem 2. *(Rochet [16]) A social choice function f is truthfully $M-$implementable with money if and only if the graph $G_{M,f}$ has no negative cycle, i.e. f satisfies the $M-$cycle monotonicity property.*

Proof [2]. If a mechanism (f, p) is truthful then

$$x_i(f(x_i)) + p(x_i) \geq x_i(f(x_{i+1})) + p(x_{i+1})$$

for each i and $x_{i+1} \in M(x_i)$. Summing over i we have that f satisfies the $M-$cyclic monotonicity.

Conversely, suppose f is $M-$cyclic monotone. For every two types $x \in M(y)$, $y \in D$ define $l(x, y)$ to be the infimum of the lengths of all finite paths from x to y in $G_{M,f}$. Note that the set of all such path lengths is bounded below by $-w(y, x)$ because otherwise appending the edge (y, x) would yield a negative cycle which contradicts to our assumption. Hence $l(x, y)$ is a well defined real number. Now add a type x_0 and add the edges (x_0, z) for all $z \in D$ with $w(x_0, z) = 0$ and define a payment function by $p(x) = l(x_0, x)$. Observe that

$$p(x) \leq p(y) + w(y, x) \leq p(y) + y(f(y)) - y(f(x))$$

and the assertion that (f, p) is truthful follows by rearranging the terms. □

Using the above theorem we can have a very useful necessary condition for a social choice function to be truthfully $M-$implementable with money which it has been proved to be also sufficient in a wide range of domains.

Definition 14. *A social choice function f satisfies M-*weak monotonicity *if for every $x \in D$ and any $y \in M(x)$*

$$x(f(x)) + y(f(y)) \geq x(f(y)) + y(f(x))$$

Equivalently, f is M-weakly monotone if and only if $G_{M,f}$ does not contain any negative 2-cycles.

If there is no verification, we simply say that f is weakly monotone.

Corollary 1. *If a social choice function f is truthfully $M-$implementable then it satisfies the $M-$weak monotonicity property.*

In the seminal work of Saks and Yu [17] the have proved that weak monotonicity is also a sufficient condition for the truthful implementation of a social choice function with money when the domain D is convex and the outcome space O is finite. In order to express this theorem we now give the definition of the convexity of a domain D.

Definition 15. *A domain D is* convex *if for every $x, y \in D$ and any $\lambda \in [0, 1]$, the function $z : O \mapsto \mathbb{R}$, with $z(a) = \lambda x(a) + (1 - \lambda)y(a)$, for each $a \in O$, is also in D.*

Theorem 3 (Saks and Yu)**.** *If the domain D is convex then a social choice function f is truthfully implementable with money if and only if the graph G_f has no negative $2-$cycle, i.e. f satisfies the weak monotonicity property.*

We present a version of this theorem in terms of verification in the Chapter 3 where we examine the equivalence between weak monotonicity and $M-$weak monotonicity.

1.3 Basic Domains and Verification Definitions

Through this work we prove some propositions and we then give the example for how they apply in some of the most usual cases. For this reason we use as examples of the application of our proposition the following domains:

- convex domains
- strategic voting
- facility location

1.3.1 Strategic Voting

We have n voters and k candidates, with $O = \{o_1, \ldots, o_k\}$ denoting the set of candidates and $V = \{v_1, \ldots, v_n\}$ denoting the set of voters. The type of each voter is a linear order over O. We write $o \succ_i o'$ to denote that voter i prefers o to o', and $v_i(o)$ to denote the rank of candidate o in the linear order of voter i.

We always assume that each type x is a functions from O to $\mathbb{R}$. However, in case of deterministic mechanisms without money, when the preferences are ordinal, we only care about the relative order of outcomes in each type.

1.3.2 k−Facility Location

In k-Facility Location, we place $k \geq 1$ facilities on the real line based on the preferences of n agents. The type of each agent i is determined by $x_i \in \mathbb{R}$, and the set of outcomes is $O = \mathbb{R}^k$. The utility of agent i from an outcome $(y_1, \ldots, y_k) \in O$ is $-\min_j |x_i - y_j|$. If $k = 1$, we simply refer to Facility Location.

Definition 16. *A social choice function f for the $k-$facility location game is a function $f : \mathbb{R}^n \to \mathbb{R}^k$, where n is the number of agents and k is the number of facilities. The inputs of this function are the real locations of the agents.*

Definition 17. *We define $f^i : \mathbb{R}^n \to \mathbb{R}$ the function which computes the position of the i-est facility with*

$$f^1 < f^2 < f^3 < \ ... \ < f^k$$

1.3.3 M$^\varepsilon$ and M$^{\text{swap}}$ Verification

In case of a convex domain or Facility Location, given an $\varepsilon > 0$, we let $M^\varepsilon(x) = \{y \in D : ||x - y|| \leq \varepsilon\}$, for all x, where $|| \cdot ||$ is the l_2 distance in $\mathbb{R}^O$ for convex domains and $|x - y|$ for Facility Location.

If we have a domain D where the agent's types are linear orders on O, for any type $x \in D$, $M^{\text{swap}}(x)$ is the set of all linear orders on O obtained from x by swapping two adjacent outcomes in x.

1.3.4 Symmetric and Asymmetric Verification

In the Chapters 3 and **??** we use some properties of verification that could be applied to any domain. These properties are *symmetricity* and *asymmetricity*.

Definition 18. *A verification is called symmetric if and only if*

$$(a, b) \in E[G_M] \Rightarrow (b, a) \in E[G_M]$$

Definition 19. *A verification is called* asymmetric *if and only if G_M is an acyclic tournament.*

1.3.5 Randomized Domains

A randomized mechanism $f : D \mapsto \Delta(O)$ maps each type x to a probability distribution over O. There two common notions of truthfulness for randomized functions: universal truthfulness and truthfulness-in-expectation. A randomized mechanism is (resp. M-)*universally truthful* if it is a probability distribution over deterministic (resp. M-)truthful mechanisms (even with money).

For truthfulness-in-expectation, we assume that O is finite, for simplicity, and let $f_o(x)$ be the probability of the outcome o if the agent reports x. Then, a randomized mechanism f is (resp. M-)*truthful-in-expectation* if for every type x and any $y \in D$ (resp. $y \in M(x)$),

$$\sum_{o \in O} f_o(x)x(o) \geq \sum_{o \in O} f_o(y)x(o)$$

A randomized mechanism f is (resp. M-)*truthful-in-expectation with money* if there are payments $p : O \mapsto \mathbb{R}$ such that for every $x \in D$ and any $y \in D$ (resp. $y \in M(x)$),

$$\sum_{o \in O} f_o(x)(x(o) + p(o)) \geq \sum_{o \in O} f_o(y)(x(o) + p(o))$$

Chapter 2

Revalation Pronciple and Non – Truthful Implementations using Verification

As we noticed in the introduction the first observation of the researchers in the the model of mechanism design with partial verification was that the revelation principle no more holds. In their paper Green and Lafford [10] give some sufficient conditions for the verification under which the revelation principle remains correct even using partial verification. After that Auletta et. al. claimed that the have found the necessary and sufficient conditions and which revelation principle is correct in the partial verification model. Unfortunately the had a mistake in their work which has been found from Lan Yu who had finally found the correct necessary and sufficient conditions. We present these conditions in the first part of this chapter.

As a consequence of the failure of the revelation principle, in a wide range of verifications, Green and Lafford [10] first present the ability of implementing social choice functions in a non-truthful way. This idea was completely new and very promising by the time Green and Lafford published their work. Unfortunately during the following years no one have found a better application of non-truthful implementation. Auletta et. al. [3] gave an explanation for this proving that the problem of finding a non-truthful implementation of social choice function even if you have the ability to write down the whole domain D is NP-complete and therefore difficult by its nature. In the second part of this chapter we present the NP-completeness proof of Auletta et al. and we also explain why dominant strategy non-truthful implementations are indeed difficult. After that we relax the requirement of dominant strategy equilibrium to the unique pure Nash equilibrium in order to implement a social choice function. Using this new concept we present some very efficient mechanisms with verification in the facility location domain which non-truthfully implement some functions which are known to be non-implementable without verification and which are also non-implementable truthfully using the same kind of verification.

2.1 Revelation Principle in the presence of Partial Verification

We first give a proof of revelation principle in order to make clear why the presence of verification clearly destroys the validity of revelation principle.

Proof of Revelation Principle. Suppose that there is mechanism g which implements the social choice function f and that f is not a truthful social choice function. This means that there must be a type x and a type y such that $x(f(y)) > x(f(x))$ or in other words the edge (x, y) is negative. Since g implements f there must be a $z \in D$ such that $g(z) = f(y)$. Now in the absence of verification an agent with type x is allowed to misreport anything. So he is also allowed to misreport z instead of x. Therefore g could never implement f because the agent with type x would always prefer to report z because he prefer the outcome $f(y) = g(z)$ than the outcome $f(x)$ which he has to take. □

The above proof of revelation principle makes clear that if we restrict x to not be able to misreport anything in the type space then there must be same cases where revelation principle fails.

This is the topic of this section where based on the work of Lan Yu we present the necessary and sufficient conditions for the verification M under which the revelation principle still holds. These conditions are expressed in terms of the graph representation. As a special case of these conditions we take the *Nested-Range Conditions* which was the first sufficient conditions proposed by Green and Lafford [10].

From now on we assume that the correspondence graph is decomposed into strongly connected components $(C_1, C_2, \ldots, C_k)$.

Definition 20. *A verification with correspondence M satisfies strong decomposability if and only if:*

1. *Each strongly connected component C_i of G_M is a complete directed subgraph, i.e. for all $t_1, t_2 \in C_i$ we have $t_1 \in M(t_2)$ and $t_2 \in M(t_1)$.*
2. *Vertices in the same strongly connected component share the same image set, i.e. for all $t_1, t_2 \in C_i$ we have $M(t_1) = M(t_2)$. In other words, if $t_1 \in C_i$, $t_2 \in C_j$ and $t_2 \in M(t_1)$, then for all $t \in C_i$, $t_2 \in M(t)$.*

Based on this definition we can prove the following which describe the exact relation between the structure of G_M and the revelation principle.

Proposition 1. *If M satisfies strong decomposability then a social choice function $f : D \mapsto O$ is $M-$implementable if and only if f is truthfully $M-$implementable.*

The proof of this proposition is based on the following lemma

Lemma 1. *For any $M-$implementable social choice function f, for any cycle C consisting of the vertices $t_1, t_2, \dots t_k$ of G_M, if $M(t_1) = M(t_2) = \dots = M(t_k)$, then C is a nonnegative cycle in $G_{M,f}$, i.e.*

$$\sum_{i=1}^{k} t_i(f(t_i)) - t_i(f(t_{i+1})) \geq 0$$

Proposition 2. *If M does not satisfy strong decomposability, then there exist a set of outcomes O an $M-$implementable social choice function f that is not truthfully $M-$implementable.*

The proof of this proposition is based on the following lemma

Lemma 2. *If the correspondence graph G_M contains an induced directed cycle of length greater than three, then there exist a set of outcomes O, an $M-$implementable social choice function f that is not truthfully $M-$implementable.*

The above propositions have the following consequence.

Theorem 4. *The revelation principle holds for a verification with correspondence M if and only if M satisfies the strong decomposability.*

Obviously since strong decomposability is a tight property we can get the sufficiency of the nested-range condition as a direct corollary of the above theorem.

Definition 21. *A correspondence M satisfies the nested-range condition (NRC) if and only if for any t_1, t_2, $t_3 \in D$, if $t_2 \in M(t_1)$ and $t_3 \in M(t_2)$, then $t_3 \in M(t_1)$.*

Corollary 2. *If M satisfies nested-range condition then a social choice function $f : D \mapsto O$ is $M-$implementable if and only if f is truthfully $M-$implementable.*

2.2 Non-Truthful Implementation

As we have already noticed the failure of revelation principle creates a new opportunity for implementation of a social choice function the non-truthful implementation. The first example of such an implementation was given by Green and Lafford [10].

Example : *Non-truthful implementation of a social choice function*

Consider a setting with $O = \{T, F\},\ D = \{u, v, w\}$ and $x(T) = 1,\ x(F) = 0\ \ \forall x \in D$. Suppose that the correspondence M is given by $M(u) = \{u, v\},\ M(v) = \{v, w\},\ M(w) = \{w\}$ and we would like to implement the social choice function $f(u) = F,\ f(v) = F,\ f(w) = T$. We can set $g(u) = g(v) = F, g(w) = T$: under this mechanism $g(u') = F\ \ \forall u' \in M(u)$ and v, w can both report w to obtain their preferred outcome $g(w) = T$. □

For a lot of years after the presentation of this example no other example was known to the community of a non-truthful implementation. Auletta et al. [3] managed to understand the reason and they have proved that even if you have the ability to work on the entire domain the problem of finding a non-truthful implementation is NP-complete. We present their reduction for both the case with and without money in the next section.

2.2.1 NP-hardness of Non-Truthful Implementability [3]

Implementability without money

Definition 22. *We define the* Implementability *decision problem as follows*

Input: *domain D, outcome set O, social choice function $f : D \to O$ and the graph G_M.*

Output: *there exists an outcome function g that $M-$implements f?*

For the reduction we are going to use $O = \{T, F\}$ and we are going to reduce $3-$Sat problem to the Implementability problem.

Theorem 5. *The* Implementability *problem is NP-hard.*

Reduction. We first notice the following obvious facts.

1. If $f(a) = T$ and $a(T) < a(F)$ then, for all $v \in M(a)$, we have $g(v) = T$.

2. If $f(a) = F$ and $a(T) < a(F)$ then, there exists $v \in M(a)$ such that $g(v) = F$.

3. If $f(a) = T$ and $a(T) > a(F)$ then, there exists $v \in M(a)$ such that $g(v) = T$.

4. If $f(a) = F$ and $a(T) > a(F)$ then, for all $v \in M(a)$, we have $g(v) = F$.

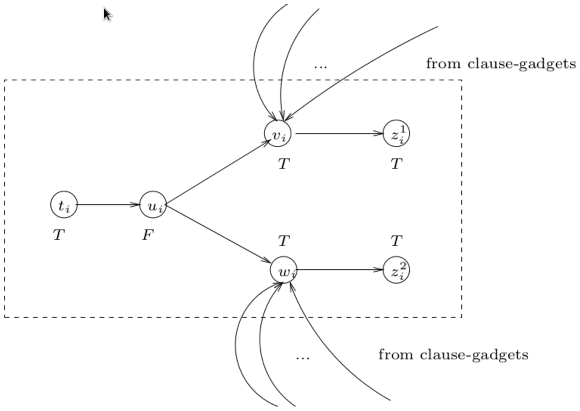


Figure 2.1: The gadget we add to G_M for every variable

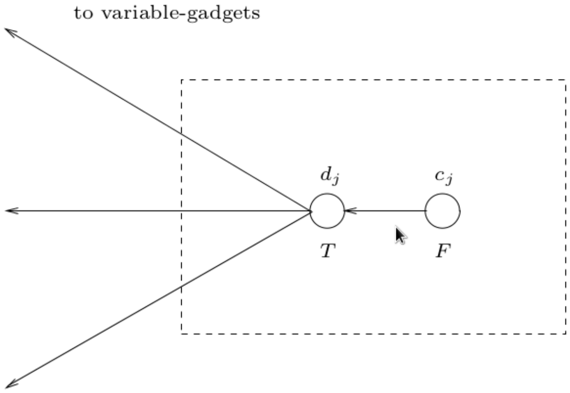


Figure 2.2: The gadget we add to G_M for every clause

Now let Φ a boolean formula in $3-$CNF over the variables $x_1, x_2, \ldots, x_n$ and let $C_1, C_2, \ldots, C_m$ be the clauses of Φ. We construct D, O, M and $f : D \mapsto O$ such that f is $M-$implementable if and only if Φ is satisfiable. The basic step is to construct a correspondence graph G_M representing M and the set of vertex of who's is the set D. We do this by creating variable gadgets (one per variable) and clause gadgets (one per clause).

The variable gadget for the variable x_i is depicted in Fig. 2.1. Each variable x_i of the formula Φ adds six new types to the domain D of the agent, namely t_i, u_i, v_i, w_i, z_i^1 and z_i^2 satisfying the following relations:

$$t_i(F) > t_i(T)$$

$$u_i(F) > u_i(T)$$

$$v_i(T) > v_i(F)$$

$$w_i(T) > w_i(F)$$

The labeling of the vertices in the figure defines the social choice function f that is, $f(t_i) = T$, $f(v_i) = T$, $f(w_i) = T$, $f(z_i) = T$, $f(z_i) = T$, and $f(u_i) = F$. Nodes v_i and w_i have incoming edges from the clause gadgets. The role of these edges will be clear in the following.

We observe that the relation about t_i implies that the social choice function f is not truthfully $M-$implementable. Indeed t_i prefers outcome $F = f(u_i)$ to $T = f(t_i)$ and $u_i \in M(t_i)$. Moreover by our initial observations for any mechanism g implementing f we must have $g(t_i) = g(u_i) = T$. On the other hand, since $f(u_i) = F$ it must be the case that any mechanism g that $M-$implements f assigns outcome F to at least one node in $M(u_i) \setminus \{u_i\}$. Intuitively, the fact that every mechanism g that $M-$implements f must assign F to at least one between v_i and w_i corresponds to assigning "false" to respectively literal x_i and $\bar{x}_i$.

The clause gadget for clause C_j of Φ is depicted in Fig. 2.2. Each clause C_j adds types c_j and d_j to the domain D of the agent such that

$$c_j(T) > c_j(F)$$
$$d_j(T) > d_j(F)$$

As before the labeling defines the social choice function f and we have $f(d_j) = T$ and $f(c_j) = F$. Moreover, directed edges encode correspondence M. Besides the directed edge (c_j, d_j), the correspondence graph contains three edges directed from d_j towards the three variable gadgets corresponding to the variables appearing in the clause C_j. Specifically, if C_j contains the literal x_i then d_j has an outgoing edge to node v_i. If C_j contains the literal $\bar{x}_i$ then d_j has an outgoing edge to node w_i. Similarly to the variable gadget, we observe that the relation about c_j implies that for any mechanism g $M-$implementing f it must be $g(d_j) = F$. Therefore, for g to $M-$implement f it must be the case that, for at least one of the neighbors a of d_j from a variable gadget, we have $g(a) = T$. We will see that this happens if and only if the formula Φ is satisfiable. This concludes the description of the reduction.

We next prove that the reduction is correct. Suppose that Φ is satisfiable, let τ be a satisfying truth assignment and let g be the mechanism defined as follows. For the ith variable gadget we set $g(t_i) = g(u_i) = g(z_i^1) = g(z_i^2) = T$. Moreover, if x_i is true in τ, then we set $g(v_i) = T$ and $g(w_i) = F$. Otherwise we set $g(v_i) = F$ and $g(w_i) = T$. For the jth clause gadget, we set $g(d_j) = g(c_j) = F$. Thus, to prove that the outcome function produced by our reduction $M-$implements f, it is sufficient to show for each type a the corresponding condition of the initial observation holds. We prove that conditions hold only for $a = u_i$ and $a = d_j$, the other cases being immediate. For u_i we have to verify that the second observation holds. Since τ is a truth assignment, for each i vertex u_i has a neighbor vertex for which the mechanism g gives F. For d_j we have to verify that the third observation holds. Since τ is a satisfying truth assignment, for each j there exists at least one literal of C_j that is true in τ therefore vertex d_j has a neighbor vertex for which the outcome function g gives T.

Conversely, consider a mechanism g which $M-$implements the social choice function f. This means that, for each clause C_j, d_j is connected to at least one node a_j from a variable gadget such that $g(a_j) = T$. Then the truth assignment that sets to true the literals corresponding to nodes $a_1, a_2, \ldots, a_m$, and gives arbitrary truth value to the other variables, satisfies the formula. □

Implementability with money

Definition 23. *The* QUASI LINEAR IMPLEMENTABILITY *decision problem is defined as follows.*

Input: *domain D, outcome set O, social choice function $f : D \mapsto O$ and the graph G_M*

Output: *decide whether there exists (g, p) that $M-$implements f?*

Theorem 6. *The* QUASI LINEAR IMPLEMENTABILITY *decision problem is NP-hard.*

2.2.2 Efficient mechanism for Facility Location using the unique pure Nash Equilibrium solution concept

If we are looking for a truthful implementation M^ϵ verification is not useful. But as Green and Lafford [10] showed, if we have partial verification then the revelation principle does not apply and therefore we may have a non-truthful implementation of a social choice function. We present here an example of such an implementation for the 1-facility location game with M^ϵ verification.

Definition 24. *We define the Mechanism 0 as follows*

$$g^1(\mathbf{y}) = g(\mathbf{y}) = \frac{\max_i y_i + \min_i y_i}{2}$$

Is is obvious that the dominant strategy for the leftmost player i_{lt}, under the mechanism 0, is to report $x_{i_{lt}} - \epsilon$ and for the rightmost player i_{rt} is to report $x_{i_{rt}} + \epsilon$. Thus $g(\mathbf{y}) = g(\mathbf{x})$, which means that g achieves the optimal maximum distance. It also known that we are not able to achieve optimal maximum distance without verification. Therefore by proposition 1 there is a social choice function which is not M^ϵ truthfully implementable but instead is M^ϵ implementable.

Remark Notice that although $g(\mathbf{y}) = g(\mathbf{x})$ the implementation is non-truthful.

We now show that at the single agent model the set of social choice functions which are M^ϵ implementable is equal with the set of social choice functions which are M^ϵ truthfully implementable.

Proposition 3. *A scale invariant social choice function f is M^ϵ truthfully implementable if and only if it is M^ϵ implementable by a scale invariant mechanism.*

Proof. It is obvious that if a function f is M^ϵ truthfully implementable then it is M^ϵ implementable.
Now suppose that f is not M^ϵ truthfully implementable but there exists a scale invariant mechanism g that M^ϵ implements f. Let $\mathbf{x} \in \mathbb{R}^n$ an instance, since f is not M^ϵ truthfully implementable there exists an agent i and a location $y_i \in M^\epsilon(x_i)$ such that

$$x_i(f(y_i)) > x_i(f(x_i)) \tag{2.1}$$

Since g M^ϵ implements f

$$\forall x \in M^\epsilon(x_i) \quad g(x) \neq f(y_i) \quad \text{and}$$

$$\exists z_i \in M^\epsilon(y_i) \quad g(z_i) = f(y_i)$$

The only way this could happen is $z_i \notin M^\epsilon(x_i)$. Taking an other instance $x_i' = x_i/r$ where $r > |z_i - x_i|/\epsilon$ all the above conditions still hold. But in this case $z_i \in M^\epsilon(x_i)$ since $|z_i - y_i| < \epsilon$ and therefore we have a contradiction. So g could not implement f. [**?**] □

Remark 1 We could generalize proposition 2 in the case where the mechanism g is not scale invariant. This could be done because using a non-scale invariant mechanism we are not able to implement a scale invariant function.

Remark 2 We see that mechanism 0 M^ϵ implements a social choice function which is impossible to truthfully M^ϵ implement it. This seems to contradicts proposition 2. However this is not true because proposition 2 is proved only in the single agent model whereas mechanism 0 is implementable in the multiagent model.

A scale invariant non-truthful mechanism for a modified game

In this section we propose a slightly modified k-facility location game as defined in section 1 and get a mechanism which achieves the optimal maximum distance of an agent to its facility. More specifically what we change is to ask from agents not only their prefered location but also which of the facilities they want to use. For the next of this work we call this game *Modified k-facility location* or for simplicity *MD k-facility location.*

MD 2-facility location game

Game definition

In the MD 2-facility location we want to place two facilities on the line. Let $F = \{1, 2\}$ be the set of facilities, each agent report a pair $(a, x) \in F \times \mathbb{R}$ where a is her prefered facility and x is her prefered location for this facility. A mechanism for the MD 2-facility location is a function $g : (F \times \mathbb{R})^n \to \mathbb{R}^2$ where n is the number of agents, this function takes as input the report of each agent and returns the location of the two facilities.

The main purpose of a mechanism for the MD 2-facility location game is to implement a social choice function that it is defined for the k-facility location game. For this reason we define the *extension* of a social choice function f for the k-facility location. An extension of a social choice function for the k-facility location is a social choice function for the MD k-facility location where there must be a way to choose the prefered facility for each agent such that the extension places the facilities in the same locations with the starting function.

Definition 25. *A social choice function $f_{ne} : (F \times \mathbb{R})^n \to \mathbb{R}^2$ for the MD facility location is said to be an extension of a social choice function $f : \mathbb{R}^n \to \mathbb{R}^2$ if and only if*

$$\forall \mathbf{x} \in \mathbb{R}^n \quad f(\mathbf{x}) = f_{ne}(\mathbf{p}(\mathbf{x}, f(\mathbf{x})))$$

where

$$p_i(x_i, \mathbf{x}_{-i}, (f^1(\mathbf{x}), f^2(\mathbf{x}))) = (a_i, x_i) \text{ and}$$
$$a_i = \{j \mid j = argmin_{k \in F}(|f^k(\mathbf{x}) - x_i|)\}$$

Definition 26. *A mechanism $g : (F \times \mathbb{R})^n \to \mathbb{R}^2$ is said to implement a social choice function $f : \mathbb{R}^n \to \mathbb{R}^2$ for the 2-facility location game if and only if there exists a extension f_{ne} of f such that g implements f_{ne}.*

Since in this game each agent report a pair the verification could verify every component of the pair. In this work we consider the following verification :

$$M^\epsilon((a, x)) = \{(a,\ y) \in F \times \mathbb{R} \mid |y - x| \leq \epsilon\}$$

The meaning of the above verification is that an agents may misreport in an ϵ area her location but she can't mireport her prefered facility. This new part of verification is very similar with the idea of winner-imposing mechanisms where the mechanism has the ability to *connect* an agent to a facility if this facility is placed on her reported location.

Mechanism definition

Let x_i be the real locations of the agent i and y_i her reported location. We may assume that $x_1 \leq x_2 \leq \ \ldots\ \leq x_n$ and $y_1 \leq y_2 \leq \ \ldots\ \leq y_n$ since the mechanism we define is anonymous. Let lt denote the location of leftmost agent and rt the location of the rightmost agent

$$lt_z = \min_i \{z_i\}\ ,\quad rt_z = \max_i \{z_i\}$$

Also let N^j (with $j \in F$) be the set of agents that have report the facility j as their prefered facility i.e. $N^j = \{i \mid a_i = j\}$ and lt^j be the location of the leftmost agent in N^j and rt^j the location of the rightmost

$$lt_z^j = \min_{i \in N^j} \{z_i\}\ ,\quad rt_z^j = \max_{i \in N^j} \{z_i\}$$

We also define the midpoint md and the maximum distance c as follows :

$$md_z = \frac{lt_z + rt_z}{2}\ ,\quad md_z^j = \frac{lt_z^j + rt_z^j}{2}$$

$$c = \max_i \{c_i\} \ , \quad c_i = |g^{a_i}(\mathbf{p}) - x_i|$$

We are now ready to introduce our mechanism for the MD 2-facility location game. Recall that x_i is the real location of the agent i and y_i is the location agent i reports.

Definition 27. *We define as Mechanism 1 the mechanism $g : (F \times \mathbb{R})^n \to \mathbb{R}^2$ with :*

$$g^1(\mathbf{p}) = \min\{md_y,\ md_y^1\} \ , \quad g^2(\mathbf{p}) = \max\{md_y,\ md_y^2\}$$

We will next analyze Mechanism 1. In this analysis we suppose that each agent i knows the real positions x_k of the other agents and of course their reported locations y_k. We discuss latter how we may implement Mechanism 1 if agents don't have complete information for the game.

Lemma 3. *The dominant strategy for the leftmost agent is to report $(1, lt_x - \epsilon)$. Similarly for the rigthmost agent the dominant strategy is to report $(2, rt_x + \epsilon)$.*

Proof. It is easy to see that, no matter what the other agents will report, when M^ϵ verification is availiable then $lt_x < md$. It is also easy to see that $g^1 \leq md_y$ whereas $g^2 \geq md_y$ this means that in any case the leftmost agent benefits by reporting that she want to use facility 1. Now since the leftmost player will denote facility 1 we will have that $lt_y = lt_y^1$. It is obvious that the leftmost agent wants md_y to be as small as possible and so by the definition of md_y^1 we can easy see that the leftmost agent wants to report the smallest possible lt_y^1. But in our case verication is availiable and so this value is $lt_x - \epsilon$.
Using symmetric arguments we can easily prove the dominat strategy for the rightmost agent. □

Corollary 3. *Mechanism 1 computes $md_y = \frac{lt_y + rt_y}{2} = \frac{lt_x - rt_x}{2} = md_x$.*

Because of the corollaty 1 we will next use $md = md_x = md_y$.

Lemma 4. *The dominant strategy for the agents with $x_i < md$ is to report $(1, y_i)$ with $y_i \in \mathbb{R}$. Similarly the dominant strategy for the agents with $x_i > md$ is to report $(2, y_i)$ with $y_i \in \mathbb{R}$.*

Proof. If an agent with $x_i < md$ reports $(2, y_i)\ \ y_i \in \mathbb{R}$ then since from lemma 1 the rightmost agent reports $(2, rt + \epsilon)$ the distance $|x_i - md^2| \geq md/4$. Now if agent i reports $(1, y_i)\ \ y_i \in \mathbb{R}$ then is easy to see that $|x_i - md^1| \leq md/4$. So we conclude that if $x_i < md$ then the player i would prefer to report $(1, y_i)$. Using symmetric arguments we can easily prove that when $x_i > md$ then the player i would prefer to report $(2, y_i)$. □

Using lemma 2 and similar arguments with lemma 1 we can prove the following.

Lemma 5. *The agent rt_x^1 will report $(1, rt_x^1 + \epsilon)$ and the agent lt_x^2 will report $(2, lt_x^2 - \epsilon)$.*

Now using the above three lemmas we can easily prove the following.

Proposition 4. *Mechanism 1 achivies $c = c*$ with $c*$ the minimum value of c for the 2-facility location game.*

Implementation of the mechanism

As noticed before the analysis of the mechanism holds only if every agent has a complete information about the game. In a different case we have to implement Mechanism 1 in multiple rounds instead of one round. We can easily show that if we run Mechanism 1 for a finite number of times then at last the game converges to the dominant strategy equilibrium.

Proposition 5. *If we play Mechanism 1 in rounds then after a finite number of round the game will converge to the dominand strategy equilibrium.*

If we examine carefully the proof of proposition 3 we will see that the only information that an agents needs to play her dominant strategy is the value *md* and not the exact location of the other agents. Also by lemma 1 we can see that in each round the leftmost and the rightmost agents have one dominant strategy. Therefore playing one round of mechanism 1 we can compute the value of *md*. Then the mechanism could report this value to all the agents and now every agent has the information needed to play her dominant strategy. So it is enough to play Mechanism 1 in two rounds to converge to the dominant strategy equilibrium.

Proposition 6. *There exists a way of implementing Mechanism 1 in two rounds.*

MD k-facility game

In the case of MD k-facility location game we can see that there is no obvious extention of the Mechanism 1 with good behavior. But there are some results that still hold. We give here, without proofs, some obvious results for the MD k-facility location game (recall that at the MD k-facility location game each agent reports not only her prefered location but also her prefered facility from the set of facilities $F = \{1, 2, 3, ..., k\}$).

We define the following sets and the veficition as in the case of MD 2-facility location.

$$N^j = \{ \ i \mid a_i = j \ \}$$

$$M^\epsilon((a, x)) = \{(a, \ y) \in F \times \mathbb{R} \mid |y - x| \leq \epsilon\}$$

Proposition 7. *In every nash equilibrium of every mechanism $g : (F \times \mathbb{R})^n \to \mathbb{R}^k$ with M^ϵ verification the sets N^j is a clustering of set of agents $N = \{1, 2, .., n\}$.*

Proposition 8. *The clustering which gives the optimal maximum distance is a nash equilibrim for every every mechanism $g : (F \times \mathbb{R})^n \to \mathbb{R}^k$ with M^ϵ verification.*

Although the above results seems positive it is easy to find other nash equilibrium for the mechanisms for the MD k-facility location which are not optimal and so the optimal clustering it is not a dominant strategy for every mechanism g.

Ideas for the 3-facility

There are some ideas for changing more the 3-facility location game and ask from agent to report more informations to get a mechanism with good behavior for more than two facilities. We describe intuitively one idea here.

In the case of 3-facility location the basic idea is to ask from agent to report $(a_1, x_1, a_2, x_2) \in F \times \mathbb{R} \times F \times \mathbb{R}$ where

- a_1, x_1 is the prefered facility and the prefered location conditional that the two facilities will be placed left from md and the third right from md
- a_2, x_2 is the prefered facility and the prefered location conditional that the first facility will be placed left from md and the other two right from md.

This new game, lets call it *more modified 3-facility location* (MMD 3-facility location), seems to have some better behaviour than the MD 3-facility location but still is not optimal. However it looks like there is a version of this new game which achieves optimal behaviour. Of course the main question here is if defining such games and finding a mechanism with optimal behaviour has any sense. Of course in the case of three facility the information we ask from agents is not to much but is seems that as the k increases the information we need will be much more. Therefore an other question is if there is a way to extent these ideas to more facilities without making the new games unuseful by asking a lot of information from the agents.

Chapter 3

Sufficient Conditions for Truthful Implementation [9]

Every function f can be implemented by an appropriately strong verification scheme combined with payments (see also Section 5.1). So, the problem now is to come up with a meaningful verification M, which is either inherent in or naturally enforceable for some interesting domains and allows for a few non-implementable functions to be M-truthfully implementable. To this end, previous work has considered two kinds of verification, namely *symmetric* and *asymmetric* verification.

Symmetric verification naturally applies to convex domains (e.g., Combinatorial Auctions) and to domains with an inherent notion of distance (e.g., Facility Location, Voting). The idea is that every type x can declare any type y not far from x. A typical example is M^{ε} verification, naturally applicable to convex domains and to Facility Location. In M^{ε} verification, each type x can declare any type y in a ball of radius ε around x. Another typical example is M^{swap} verification, naturally applicable to Voting and to ordinal preference domains. In M^{swap} verification, each type x is as a linear order on O and can declare any type y obtained from x by swapping two adjacent outcomes. Rather surprisingly, previous work provides strong evidence that symmetric verification does not give any benefit to the principal, as far as truthful implementation is concerned. In particular, the strong and elegant result of Archer and Kleinberg [2] and its extension by Berger, Müller, and Naeemi [5] imply that M^{ε} verification does not help in convex domains. Formally, the results of [2, 5, 6] imply that for any convex domain, a function is truthfully implementable with money if and only if it is M^{ε}-truthfully implementable with money. Similarly, Caragiannis, Elkind, Szegedy, and Yu [6] proved that M^{swap} verification does not help in the domain of Voting .

As far as implementation without money is concerned, the research on the power of symmetric verification is closely related to the research about sufficient and necessary conditions under which weaker properties are equivalent to global truthfulness. Even though the motivation for studying weaker properties may be more general (see e.g., [17, 2, 7, 18]), in the absence of money, local truthfulness is essentially a special case of symmetric verification. In this research agenda, Sato [18] considered M^{swap} verification (under the name of adjacent manipulation truthfulness) for ordinal preference domains, and proved that if $G_{M^{\text{swap}}}$ is strongly connected by paths satisfying the no-restoration property, then truthful implementation and M^{swap}-truthful implementation are equivalent. He also proved that the universal domain, that includes all linear orders on O, and single-peaked domains have the no-restoration property, and thus, for these domains, truthful implementation is equivalent to M^{swap}-truthful implementation. Independently, Carroll [7] obtained similar results for convex domains, for the universal domain, and for single-peaked and single-crossing domains, which also extend to randomized mechanisms. Carroll also gave a necessary condition for the equivalence of local and global truthfulness in a specific domain with cardinal preferences.

On the other hand, asymmetric verification is "one-sided". Given a social choice function f, a typical example of asymmetric verification is when the agent can only lie either by overstating or by understating her utility. E.g., for Scheduling on related machines, the machine can only lie by overstating its speed [4], for Combinatorial Auctions, the agent can only underbid on her preferred sets [12], and for Facility Location, the agent can only understate her distance to the nearest facility [8]. The use of asymmetric verification has led to strong positive results about the truthful implementation of natural social choice functions in several important domains (see e.g., [4, 12, 8] and the references there in). The intuition is that the social choice function is monotonic and discourages one direction of lying (e.g., underbidding for Combinatorial Auctions), while the other direction of lying is forbidden by the verification.

Motivation and Contribution. Our work is motivated by the general observation, stated explicitly and justified in [6], that even very strict symmetric verification schemes do not help in truthful implementation, while strong positive results are possible with the use of simple asymmetric verification. So, we seek a deeper understanding of the reasons that make symmetric verification essentially useless in virtually all domains, and some formal justification behind the success of asymmetric verification.

Departing from previous work, we do not restrict ourselves to any particular domain or to any particular kind of verification. To the contrary, we consider any possible (finite or infinite) domain D and very general classes of partial verification. To formalize the notions of symmetric and asymmetric verification, we say that a verification M is symmetric if the presence of a directed edge (x, y) in G_M implies the presence of the reverse edge (y, x), and asymmetric if G_M is an acyclic tournament.

Our main result is a general and unified explanation about the weakness of symmetric verification. In Section 3.1, we identify a natural property, namely that the correspondence graph G_M is strongly connected by finite paths along which the preferences are consistent with the preferences at the endpoints. In fact, we define three versions of this property depending on whether we consider implementation by deterministic truthful mechanisms (strict order-preserving property), by deterministic mechanisms that use payments (strict difference-preserving property), and by randomized truthful-in-expectation mechanisms (difference-convex property). Despite the slightly different definitions, the essence of the property is the same, but stronger versions of it are required as the mechanisms become more powerful. We show that for any (finite or infinite) domain D and any symmetric verification M that satisfies the corresponding version of the property, deterministic / randomized truthful implementation (resp. with money) is equivalent to deterministic / randomized M-truthful implementation (resp. with money). In all cases, the proof is simple and elegant, and only exploits an elementary combinatorial argument on the paths of G_M. With this general sufficient condition for the equivalence of truthfulness and M-truthfulness, we simplify, unify, and strengthen several known results about symmetric verification and local truthfulness without money. E.g., we obtain, as simple corollaries, the equivalence of truthful and M^{ε}-truthful implementation for any convex domain (even with money) and for Facility Location, and the equivalence of truthfulness and M^{swap}-truthfulness for Voting.

In Section 3.2, we identify necessary conditions for the equivalence of truthfulness and M-truthfulness, for any symmetric verification M. These are relaxed versions of the sufficient conditions, and require that the correspondence graph G_M is strongly connected by finite preference preserving paths. Otherwise, we show how to find a separator of G_M, which in turn, leads to the definition of a function that is M-truthfully implementable, but not implementable. We also observe that the necessary condition is violated by the domain of 2-Facility Location. To conclude the discussion about symmetric verification, we close the small gap between the sufficient and necessary properties, and present the first known condition that is both sufficient and necessary for the equivalence of truthful and M-truthful implementation. Overall, our conditions provide a generic and convenient way of checking whether truthful implementation can take advantage of any symmetric verification scheme in any domain.

Finally, in Section 5.1, we consider asymmetric verification, and prove that a social choice function f is M-truthfully implementable by some asymmetric verification M if and only if the subgraph of G_f consisting of negative edges is acyclic (Theorem 15). This result provides strong formal evidence about the power of asymmetric verification, since, as we discuss in Section 5.1, any reasonable social choice function f should not have a cycle in G_f that entirely consists of negative edges. Moreover, we prove that given any function f truthfully implementable by payments p, an asymmetric verification that truthfully implements f can be directly obtained by p (Proposition 9).

Comparison to Previous Work. The strict order-reserving property, which we employ as a sufficient condition for deterministic truthful implementation without money, is similar to the no-restoration property of [18]. However, the results of [18] are restricted to finite domains with ordinal preferences and to M^{swap} verification. Our results are far more general, since we manage, in Theorem 7, to extend the equivalence of truthful and M-truthful implementation, under the strict order-preserving property, to any (even infinite) domain and to any symmetric verification. Moreover, our necessary property generalizes and unifies the necessary conditions of both [7, 18].

We also note that our results in case of deterministic implementation with money are not directly comparable to the strong and elegant results about local truthfulness with money in convex domains (see e.g., [2, 1]). For instance, if we restrict Theorem 9 to convex domains and compare it to [2, Theorem 3.8], our result is significantly weaker, since it starts from a much stronger hypothesis (see also the discussion in Section 3.1.2). On the other hand, Theorem 9 is more general, in the sense that it applies to any symmetric strict difference-preserving verification and to arbitrary (even non-convex) domains.

3.1 Sufficient Conditions for Truthful Implementation

Without any additional assumptions on the domain, symmetric verification is not sufficient for the equivalence of truthfulness and M-truthfulness and we give now such an example in the 2-Facility Location setting.

Example : *A Social Choice Function that is M^ε-Truthful but not Truthful*
We next present a social choice function g that is M^ε-truthfully implementable but not truthfully implementable. The function g is defined in the 2-Facility Location domain from the perspective of a single agent. Specifically, the social choice function $g : \mathbb{R} \to \mathbb{R}^2$ is defined as:

$$g^1(x) = \begin{cases} -2 & \text{if } x \in (-\infty, 3/4 + \varepsilon] \cup [1, \infty) \\ -1.5 & \text{if } x \in (3/4 + \varepsilon, 1) \end{cases}$$

$$g^2(x) = \begin{cases} 3 & \text{if } x \in (-\infty, 0] \cup [1/4 - \varepsilon, \infty) \\ 2.5 & \text{if } x \in (0, 1/4 - \varepsilon) \end{cases}$$

To see that g is not truthful, we let the agent be at $x \in [0, 1/2]$ and use the facility on the left. Then, the agent has an incentive to declare any $y \in (3/4 + \varepsilon, 1)$, so that the facility on the left moves from -2 to -1.5. On the other hand, for every $x \in \mathbb{R}$, the agent has no incentive to declare any false location $y \in M^\varepsilon(x)$. Therefore, g is M^ε-truthfully implementable. Hence, g is a social choice function that is M^ε-truthfully implementable, but not truthfully implementable, which holds despite the fact that M^ε verification is symmetric and that the correspondence graph $G_{M^\varepsilon, g}$ is strongly connected (albeit not with order-preserving paths).

Moreover, G_g contains the negative cycle $(0.1, 0.3, 0.1)$, with:

$$\begin{aligned} u(0.1, g(0.1)) - u(0.1, g(0.3)) + u(0.3, g(0.3)) - u(0.3, g(0.1)) = \\ = -2.1 - (-2.1) + (-2.3) + (-2.2) = -0.1\,, \end{aligned}$$

where $u(x, g(y)) = -\min\{|x - g^1(y)|, |x - g^2(y)|\}$ is the utility of the agent at location x from the outcome of g if she declares location y, which for 2-Facility Location is equal to minus the distance of x to the nearest facility in $g(y)$. Due to the negative cycle above, we conclude that g is not truthfully implementable even if we use money. □

In this section, we assume that the correspondence graph G_M is symmetric and strongly connected by finite paths along which the preferences are consistent with the preferences at the endpoints. We prove that this property suffices for the equivalence of truthful and M-truthful implementation, even for infinite domains. To demonstrate that our result is applicable to infinite domains, we give two such examples where $G_{M^{\varepsilon}}$ is strongly connected by finite preference preserving paths at the end of the section.

3.1.1 Deterministic Mechanisms

We start with a sufficient condition for a symmetric verification M (and its correspondence graph) under which any deterministic M-truthful mechanism is also truthful.

Definition 28 (Order-Preserving Path). *Given a verification M, an $x - y$ path p in G_M is* order-preserving *if for all outcomes $a, b \in O$, with $x(a) > x(b)$ and $y(a) \geq y(b)$, and for any intermediate type w in p, $w(a) > w(b)$. A $x - y$ path p in G_M is* strict order-preserving *if for every type w in p, the subpath of p from x to w is order-preserving.*

Intuitively, if the endpoints x and y of an order-preserving path p agree that outcome a is preferable to outcome b, any intermediate type w in p should also agree on this. Following Definition 28, we say that a verification M is *symmetric* (resp. *strict*) *order-preserving* if M is symmetric and for any types $x, y \in D$, there is a *finite* (resp. strict) order-preserving $x - y$ path in the correspondence graph G_M. Next, we show that:

Theorem 7. *Let M be a symmetric strict order-preserving verification. Then, truthfulness is equivalent to M-truthfulness.*

Proof. If a social function is truthfully implementable, it is also M-truthfully implementable. The converse is proven by induction on the length of the strict order-preserving paths in G_M. Technically, for sake of contradiction, we assume that there is a function f that is M-truthfully implementable, but not implementable. Therefore, all edges in $G_{M,f}$ are non-negative, but there is a negative edge $(x, z) \in E(G_f)$.

Since M is symmetric strict order-preserving, there is a finite strict order-preserving $x - z$ path p in $G_{M,f}$. In particular, we let $p = (x = v_0, v_1, v_2, \dots, v_k = z)$, and let i, $2 \leq i \leq k$, be the smallest index such that the edge $(x, v_i) \in E(G_f)$ is negative. For convenience, we let $y = v_i$ and $w = v_{i-1}$. We note that by the definition of i, the edge $(x, w) \in E(G_f)$ is non-negative, and also since f is M-truthfully implementable, the edges $(w, y), (y, w) \in E(G_{M,f})$ are non-negative (see also Fig. 3.1.i).

For convenience, we let $a = f(x)$, $b = f(w)$, $c = f(y)$ denote the outcome of f at x, y, and w, respectively. Since the edge (x, y) is negative, $a \neq c$. Moreover, by the definition of i (and of y), $b \neq c$. By the discussion above, we have that $x(c) > x(a) \geq x(b)$ and $y(c) \geq y(b)$. Therefore, since the $x - z$ path is strict order-preserving, and thus its $x - y$ subpath is order-preserving, we obtain that $w(c) > w(b)$, a contradiction to the hypothesis that the edge $(w, y) \in E(G_{M,f})$ is non-negative. Therefore there is no negative edge in G_f, which implies that f is truthfully implementable. □

If the domain D is finite, we next show that for a symmetric verification, the strict order-preserving property is equivalent to the order-preserving property.

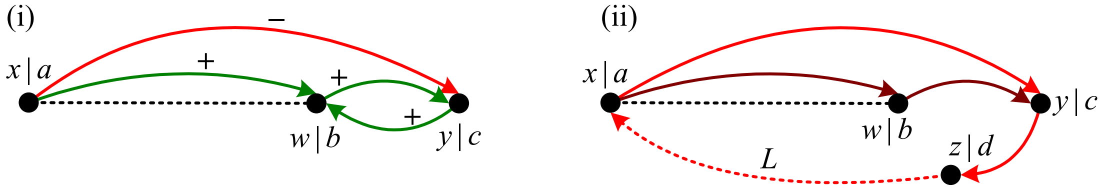


Figure 3.1: (i) The part of G_f considered in the proof of Theorem 7. (ii) The part of G_f considered in the proof of Theorem 9. The label of each node consists of the type and the outcome of f.

Lemma 6. *In a finite domain D, a symmetric verification M is order-preserving if and only if M is strict order-preserving.*

Proof. Clearly, if M is strict order-preserving, it is also order-preserving. The proof of the converse is by induction on the length of order-preserving paths in G_M. Technically, for sake of contradiction, we assume that M is order-preserving, but not strict order-preserving. Hence, there are types x, z so that there is no strict order-preserving $x - z$ path. Since M is order-preserving, there is an order-preserving $x - z$ path in G_M.

For any order-preserving $x - z$ path $p = (x = v_{k+1}, v_k, v_{k-1}, \dots, v_1, v_0 = z)$, we let $l(p)$ be the largest index such that every subpath $(x, v_k, \dots, v_j)$ with $j < l(p)$ is order preserving. Namely, $v_{l(p)}$ is the first node of p (as we move from z to x) where the order-preserving property fails. We note that the index $l(p)$ is well defined and satisfies $0 < l(p) \leq k - 1$, because p is not strict order-preserving. We now let $q = (x, v_k, \dots, v_{l(q)}, \dots, v_1, z)$ be the $x - z$ order-preserving path with the maximum $l(q)$. The path q is well defined because D is finite and $l(q) \leq |D|$. Since M is order-preserving, there is an order-preserving $x - v_{l(q)}$ path $r = (x, u_\ell, \dots, u_1, v_{l(q)})$. Next we prove that the subpath $(v_{l(q)}, \dots, v_1, z)$ of q and the subpath $(x, u_\ell, \dots, u_1)$ of r do not have any nodes in common.

Claim. *The subpath $(v_{l(q)}, \dots, v_1, z)$ of q and the subpath $(x, u_\ell, \dots, u_1)$ of r do not have any nodes in common.*

Proof. For sake of contradiction, we assume that the subpath $q' = (v_{l(q)}, \dots, v_1, z)$ of q and the subpath $r' = (x, u_\ell, \dots, u_1)$ of r have a node $u_i = v_j$ (v_j could be z) in common. We recall that $j < l(q) \leq k - 1$, and let $y = u_i = v_j$, for convenience.

Since the path r is order preserving, we have that:

$$\forall a, b \in O,\ (x(a) > x(b) \wedge v_{l(q)}(a) \geq v_{l(q)}(b) \Rightarrow y(a) > y(b)) \tag{3.1}$$

Moreover, by the definition of $l(q)$, the path $q'' = (x, v_k, \dots, v_{l(q)}, \dots, v_j = y)$ is order-preserving. Hence, for any $i \in \{j, \dots, k\}$,

$$\forall a, b \in O,\ (x(a) > x(b) \wedge y(a) > y(b) \Rightarrow v_i(a) > v_i(b)) \tag{3.2}$$

Combining (3.1) and (3.2), we obtain that for any $i \in \{j, \dots, l(q)\}$,

$$\forall a, b \in O,\ (x(a) > x(b) \wedge v_{l(q)}(a) \geq v_{l(q)}(b) \Rightarrow v_i(a) > v_i(b))$$

Therefore, the path $(x, v_k, \dots, v_{l(q)})$ is order-preserving, a contradiction to the definition of $l(q)$. □

The $x-z$ path $p = (x, u_\ell, \ldots, u_1, v_{l(q)}, \ldots, v_1, z)$ is order-preserving because the $x - v_{l(q)}$ path r and the $x - z$ path q are both order-preserving. Moreover, by the definition of l and since r is order-preserving, $l(p) > l(q)$, which contradicts the assumption that q is the order-preserving $x - z$ path with maximum $l(q)$. □

Combining Theorem 7 and Lemma 6, we obtain that:

Theorem 8. *Let M be a symmetric order-preserving verification in a finite domain D. Then, truthfulness is equivalent to M-truthfulness.*

Applications

Theorems 7 and 8 provide a generic and convenient way of checking whether truthful implementation can take any advantage of symmetric verification. E.g., it is not hard to verify that for any convex domain D, M^ε verification is (symmetric and) strict order-preserving, and that for Strategic Voting, M^{swap} verification is (symmetric and) order-preserving (Lemmas 10 and 7). Thus, we obtain alternative (and very simple) proofs of [6, Theorems 3.1 and 3.3]. Moreover, our corollary about M^{swap} verification implies the main result of [18]. Similarly, we show, in Lemma 8, that for the Facility Location domain, which is non-convex, M^ε verification is strict order-preserving. Thus, for Facility Location, a mechanism is truthful iff it is M^ε-truthful. This is similar to [18, Prop. 4.2] and to [7, Prop. 3]. However, we consider here a very restricted setting, where the set of outcomes is $\mathbb{R}$ and the preferences are given by a linear function of the agent's distance to the facility. Thus, [18, Prop. 4.2] and [7, Prop. 3] do not imply our result for Facility Location with M^ε verification.

Convex Domains

Convex domains have an stronger property which we call strict difference-preserving property which implies directly the strict order-preserving property we prove in the next section that convex domains have the strict difference-preserving property 10.

Strategic Voting

Lemma 7. *The M^{swap} verification is symmetric and strict order-preserving.*

Proof. It is easy to see that M^{swap} verification is symmetric, since applying the same swap two times, we get the initial linear order. Now we want to prove that M^{swap} is order-preserving. For this, suppose that we take two arbitrary types x, z and two outcomes a, b with $a \succ_x b$ and $a \succ_z b$. We have to prove that there is a sequence $y_1, y_2, \ldots, y_n$ with $y_1 = x, y_n = z$, $y_{i+1} \in M^{\text{swap}}(y_i)$ and also $a \succ_{y_i} b$ for every i. One way to find such a sequence is to bring, by consecutive swaps, the outcome which is first in z to the first position from its initial position to x and the same for the second outcome of z, etc. Therefore for every x, z there is an order-preserving path and because the domain is finite. Using Lemma 6, we can conclude that M^{swap} verification is symmetric strict order-preserving. □

Facility Locations
For the purposes of the following proof, we have to find a strict order-preserving path between any two types x, y. Thus we have to find a finite path with the strict difference preserving property. Note that although the domain is infinite if we consider a sequence of types with distance ε we can get a finite $x - y$ path. Through the proof we use this kid of finite paths to prove the strict difference preserving property.

Lemma 8. *For any $\varepsilon > 0$, the M^ε verification is symmetric strict order-preserving for the Facility Location domain.*

Proof. Again it is obvious that the M^ε verification is symmetric and we have to prove that it is also strict order-preserving. To do this we prove that for any $x, z \in \mathbb{R}$, any $a, b \in O$ and any $y \in [x, z]$ we take cases for the position of $a, b \in O$.

- $a, b \in (-\infty, x] :$ $x(a) > x(b)$ implies $a > b$ which implies $z(a) > z(b)$ and $y(a) > y(b)$
- $a \in (-\infty, x], b \in (x, y] :$ of course in this case $z(b) > z(a)$ and $y(b) > y(a)$ and so we don't care about x
- $a \in (-\infty, x], b \in (y, z] :$ of course in this case $z(b) > z(a)$ and $x(b) > x(a)$ implies $x > (a + b)/2$ which implies $y > (a + b)/2$ and so $y(b) > y(a)$
- $a \in (-\infty, x], b \in (z, \infty)$: $x(a) > x(b)$ implies $z(a) > z(b)$ and $y(a) > y(b)$ also $x(b) > x(a)$ and $z(b) \geq z(a)$ implies $y(b) > y(a)$
- $a \in (x, y], b \in (x, y] :$ $x(a) > x(b)$ implies $a < b$ which implies $z(b) > z(a)$
- $a \in (x, y], b \in (y, z] :$ of course in this case $x(a) > x(b)$ and $z(b) > z(a)$
- $a \in (x, y], b \in (z, \infty) :$ of course in this case $x(a) > x(b)$ and if $z(a) > z(b)$ implies $z < (a + b)/2$ which implies $y < (a + b)/2$ and so $y(a) > y(b)$
- $a \in (y, z], b \in (y, z] :$ $x(a) > x(b)$ implies $a < b$ which implies $z(b) > z(a)$
- $a \in (y, z], b \in (z, \infty) :$ of course in this case $x(a) > x(b)$ and if $z(a) > z(b)$ implies $z < (a + b)/2$ which implies $y < (a + b)/2$ and so $y(a) > y(b)$
- $a, b \in (z, \infty) :$ $x(a) > x(b)$ implies $a < b$ which implies $z(a) > z(b)$ and $y(a) > y(b)$

From the above analysis of cases we have that in every case the order-preserving property is satisfied. We then take a path p the sequence of points from x with step ε on the line segment and so after a finite number of steps we reach z. This path belongs to G_{M^ε} since every consecutive points have distance ε. Therefore p is order-preserving and since all the points are on the same line segment it is obvious that this path is also strict order-preserving. Since $x, z \in D$ are arbitrary we conclude that M^ε is symmetric strict order-preserving. □

3.1.2 Deterministic Mechanisms with Money

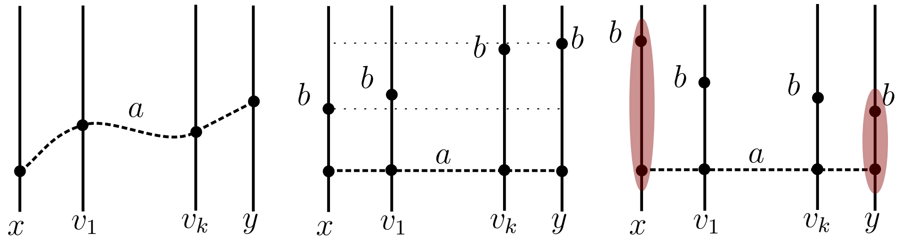
Next, we extend the notion of order-preserving paths to mechanisms with money. Since utilities are not ordinal anymore, we use the notion of difference-preserving paths, which takes into account the difference between the utility of different outcomes. Formally, given a verification M, an $x - y$ path p in G_M is *difference-preserving* if for any intermediate type w in p and for all outcomes $a, b \in O$, if $x(a) - x(b) \neq y(a) - y(b)$,

$$w(a) - w(b) \in (\min\{x(a) - x(b), y(a) - y(b)\}, \max\{x(a) - x(b), y(a) - y(b)\})$$

As for order-preserving paths, if both endpoints x and y of a difference-preserving path p prefer a to b, any type w in p should also prefer a to b. Moreover, the strength of w's reference for a, i.e., $w(a) - w(b)$, should lie between the strength of x's and of y's preference for a. We now give an intuitive interpretation of the difference-preserving property which also give a really simple way of thinking the proofs of the next theorems.

Interpretation of the Difference-Preserving Property Using Path Diagrams

There is an intuitive way of thinking about the difference-preserving property using the *path diagrams* defined here. Let $p = (x, v_1, v_2, \ldots, v_k, y)$ be a difference-preserving $x - y$ path in G_M. Then, the *$x - y$ path diagram* is the plot having the sequence $x, v_1, v_2, \ldots, v_k, y$ on the horizontal axis and the utility of each type in p for some possible outcomes on the vertical axis. E.g., in the first path diagram below, we have depicted the utility of each type in p for an outcome a.



To simplify a path diagram, we use the fact that truthfulness is not affected if a function g that depends only on the real type of the agent is added to the utility function. Namely, if the agent has real type x and reports type y, we let her modified utility be $u'(x, f(y)) = u(x, f(y)) + g(x)$, instead of $u(x, f(y))$. Then, it not hard to verify that f is truthful with utility function u iff f is truthful with utility function u'.

Using this argument, we can simplify a path diagram as follows. We choose an outcome a, set the modified utility of each type in p for a equal to 0, by letting $g(x) = -u(x, a)$, and set the modified utility of each type for the remaining outcomes accordingly. The important observation is that the difference of the utility of each type x for two outcomes a and b remains the same in both u and u'. So, as far as the difference-preserving property is concerned, we can have the same conclusion from both the original and the modified path diagrams. Applying this transformation, the path diagram allows one to directly conclude where a path p is difference-preserving or not, as it is shown by the second diagram above.

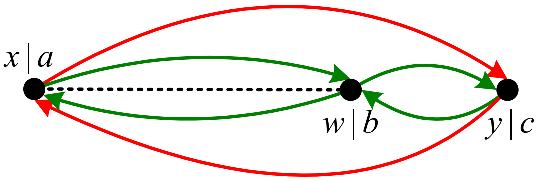


Figure 3.2: The part of G_f considered in the proof of Lemma 9. The label of each node includes the type and the outcome of f. The 2-cycles (x, w, x) and (w, y, w) are non-negative. For sake of contradiction, we assume that the 2-cycle (x, y, x) is negative.

Path diagrams also provide an intuitive way of thinking about negative 2-cycles. E.g., the third diagram depicts a negative cycle (x, y, x) where $f(x) = a$ and $f(y) = b$.

In fact, the difference-preserving property is a stronger version of the increasing difference property in [5, Definition 5].

Similarly, an $x - y$ path p in G_M is *strict difference-preserving* if for every type w in p, the subpath of p from x to w is also difference-preserving. A verification M is *symmetric* (resp. *strict*) *difference-preserving* if M is symmetric and for any $x, y \in D$, there is a *finite* (resp. strict) difference-preserving $x - y$ path in G_M.

We proceed to show that the symmetric strict difference-preserving property is sufficient for the equivalence between M-truthfulness with money and truthfulness with money. The proof is based on the equivalence between cycle monotonicity and truthful implementation with money. As a first step, we show that under the symmetric strict difference-preserving property, for any social choice function f, $G_{M,f}$ does not have any negative 2-cycles iff G_f does not have any negative 2-cycles. The proof is similar to the proof of Theorem 7.

Lemma 9. *Let M be a symmetric strict difference-preserving verification. Then for any social choice function f, f is M-weakly monotone if and only if f is weakly-monotone.*

Proof. If f is weakly-monotone, it is also M-weakly monotone. The proof of the converse is essentially an extension of the proof of Theorem 7 to mechanisms with money and to difference-preserving paths. For sake of contradiction, let us assume that f is M-weakly monotone, but not weakly-monotone. Therefore, every 2-cycle in $G_{M,f}$ is non-negative, while there is a negative 2-cycle (x, z, x) in G_f. Since M is symmetric strict difference-preserving, there is a finite strict difference-preserving $x - z$ path $p = (x, v_1, v_2, \dots, z)$ in G_M. We let $i \geq 2$ be the smallest index such that the 2-cycle (x, v_i, x) in G_f is negative. For convenience, we let $y = v_i$ and $w = v_{i-1}$. We note that by the definition of i, the 2-cycle (x, w, x) in G_f is non-negative, and also since f is M-weakly monotone, the 2-cycle (w, y, w) is non-negative (see also Fig. 3.2).

For convenience, we let $a = f(x)$, $b = f(w)$, $c = f(y)$ denote the outcome of f at x, y, and w, respectively. Assuming that the 2-cycle (x, y, x) is negative, i.e., that $x(c) - x(a) > y(c) - y(a)$, we reach a contradiction by considering the following cases:

- $x(b) - x(a) > y(b) - y(a)$. Since w belongs to a difference-preserving $x - y$ path, we have that $x(b) - x(a) > w(b) - w(a)$, i.e., that the 2-cycle (x, w, x) is negative.
- $y(b) - y(a) > x(b) - x(a)$. Then, $y(b) - y(a) > w(b) - w(a)$ and $w(c) - w(a) >$

$y(c) - y(a)$, both because w belongs to a difference-preserving $x - y$ path. Thus, $w(c) - w(b) > y(c) - y(b)$, i.e., the 2-cycle (w, y, w) is negative.

- $x(b) - x(a) = y(b) - y(a)$. Then since $x(c) - x(a) > y(c) - y(a)$ we have that $x(c) - x(b) > y(c) - y(b)$. Thus $w(c) - w(b) > y(c) - y(b)$, i.e., the 2-cycle (w, y, w) is negative.

□

Using Lemma 9, we next show that under the symmetric strict difference-preserving property, M-cycle monotonicity is equivalent to cycle monotonicity, and thus, M-truthful implementation with money is equivalent to truthful implementation with money.

Theorem 9. *Let M be a symmetric strict difference-preserving verification. Then for any social choice function f, f is M-truthfully implementable with money if and only if f is truthfully implementable with money.*

Proof. If f is truthfully implementable with money, it is also M-truthfully implementable with money. For the converse, we show that if $G_{M,f}$ does not have any negative cycles, then G_f does not have any negative cycles as well. In what follows, we assume that G_f does not have any negative 2-cycles, since otherwise, by Lemma 9, f is not M-weakly monotone, and thus, not truthfully implementable with money.

For sake of contradiction, we assume that G_f includes some negative cycle with more than 2 (and a finite number of) edges. In particular, we let $C = (x, y, z, \dots, x)$ be any such cycle. The existence of such a cycle C is guaranteed by Rochet's theorem. Moreover, C contains at least one edge $(x, y) \in E(G_f) \backslash E(G_{M,f})$, because C is not present in $G_{M,f}$. Since M is a symmetric strict difference-preserving verification, there is a finite strict difference-preserving $x - y$ path $p = (v_0 = x, v_1, \dots, v_k = y)$. For convenience, we let $w = v_{k-1}$ be the last node before y in p, let $a = f(x)$, $b = f(w)$, $c = f(y)$, and $d = f(z)$ be the outcome of f at x, w, y, and z, respectively, and let L be the total length of the $z - x$ path used by C (see also Fig. 3.1.ii).

Since the cycle C is negative, $x(a) - x(c) + y(c) - y(d) + L < 0$. Moreover, since G_f does not contain any negative 2-cycles, $x(c) - x(b) \leq y(c) - y(b)$. Otherwise, since w belongs to a difference-preserving $x - y$ path, we would have that $y(c) - y(b) < w(c) - w(b)$, which implies that the 2-cycle (w, y, w) is negative. Hence, since w belongs to a difference-preserving $x - y$ path, $x(c) - x(b) \leq w(c) - w(b)$. Therefore,

$$x(a) - x(b) + w(b) - w(c) + y(c) - y(d) + L \leq x(a) - x(c) + y(c) - y(d) + L < 0$$

So, we have that the cycle $C_1 = (x, w = v_{k-1}, y, \dots, z)$ is also negative.

Since p is strict difference-preserving, the path $p' = (x = v_0, v_1, \dots, v_{k-1} = w)$ is also difference-preserving. Therefore, using the same argument, we can prove that the cycle $C_2 = (x, v_{k-2}, v_{k-1}, y, \dots, z)$ is also negative. Repeating the same process $k - 1$ times, we obtain that the cycle $C_{k-1} = (x = v_0, v_1, \dots, v_{k-1}, y, \dots, z)$ is also negative.

However, all the edges (v_i, v_{i+1}), $i = 0, \ldots, k-1$, of the strict difference-preserving $x-y$ path p belong to G_M. Hence, the edge $(x, y) \in E(G_f) \setminus E(G_{M,f})$ in C is replaced by k edges of $E(G_{M,f})$ in C_{k-1}. Therefore, the negative cycle C_{k-1} has one edge not in $E(G_{M,f})$ less than the original negative cycle C. Repeating the same process for every edge of C not in $E(G_{M,f})$, we obtain a negative cycle C' with all edges in $E(G_{M,f})$. This is a contradiction, since it implies that f is not M-truthfully implementable with money. □

Application

In order to apply Theorem 9 we now prove that M^ε verification is symmetric and strict difference-preserving for any convex domain.

Convex Domains

To establish the strict difference-preserving property of M^ε, we have to find a strict difference-preserving path between any two types x and y. Thus we have to find a finite path with the strict difference-preserving property. Note that although the domain is infinite, if we consider a sequence of types with distance ε from each other, we can get a finite $x-y$ path. In the proof, we use this kind of finite paths to establish the strict difference preserving property.

Lemma 10. *Let D be any convex domain. Then, M^ε verification in D is symmetric and strict difference-preserving.*

Proof. It is obvious that the M^ε verification is symmetric and so we want to prove that it is strict difference-preserving. Let $x, z \in D$ and let $y = \lambda x + (1-\lambda)z$, also let $a, b \in O$ we have that $y(a) - y(b) = \lambda(x(a) - x(b)) + (1-\lambda)(z(a) - z(b))$ and since $\lambda \in [0, 1]$ we have that

$$y(a) - y(b) \in [\max(x(a) - x(b), z(a) - z(b)), \min(x(a) - x(b), z(a) - z(b))]$$

Therefore every path between x and z with points on the line segment between x and z is an difference-preserving path. We then take a path p the sequence of points from x with step ε on the line segment and so after a finite number of steps we reach z. This path belongs to G_{M^ε} since every consecutive points have distance ε. Therefore p is difference-preserving and since all the points are on the same line segment it is obvious that this path is also strict difference-preserving. Since $x, z \in D$ are arbitrary we conclude that M^ε is symmetric strict difference-preserving. □

As we observed in Section 3.1.2, strict difference-preserving property implies strict order-preserving property and therefore the above lemma also proves that M^ε verification on a convex domain D is symmetric order-preserving.

Since M^ε verification is symmetric and strict difference-preserving for any convex domain, Theorem 9 implies that for convex domains, M^ε-truthful implementation with money is equivalent to truthful implementation with money. This result is also a corollary of [2, Theorem 3.8], but here we obtain it through a completely different approach (and give a very simple proof using elementary combinatorial tools). In particular, Archer and Kleinberg [2] proved that if there is no "local" negative cycle C in G_f, where "local" means that C can fit in a small area of the convex domain D, then G_f does not contain any negative cycles, and thus, f is truthfully implementable with money. On the other hand, we prove here that if G_f does not contain any negative cycles consisting of "short" edges, where "short" means that the endpoints of the edge are very close to each other in D, then G_f does not contain any negative cycles. So, in our case, the hypothesis is much stronger, since it excludes the existence of negative cycles that consist of "short" edges, but may cover an arbitrarily large area of the convex domain D. In this sense, if we restrict Theorem 9 to convex domains, our result is different in nature and weaker than [2, Theorem 3.8]. Nevertheless, Theorem 9 is quite more general, in the sense that it applies to any symmetric strict difference-preserving verification and to arbitrary (even non-convex) domains.

3.1.3 Randomized Truthful-in-Expectation Mechanisms

A general condition is sufficient and/or necessary for the equivalence between universal truthfulness and M-universal truthfulness in randomized mechanisms, iff it is sufficient and/or necessary for the equivalence between truthfulness and M-truthfulness in deterministic mechanisms. Hence, all the results of Sections 3.1.1, 3.1.2, and 3.2 directly apply to randomized universally-truthful mechanisms (also with money).

A similar, but more interesting, correspondence holds for the case of randomized truthful-in-expectation mechanisms. For simplicity, we assume here that the set of outcomes $O = \{o_1, \dots, o_m\}$ is finite. With each type $x : D \mapsto \mathbb{R}$, we associate a new type $X : \Delta(O) \mapsto \mathbb{R}$, such that for each probability distribution $\vec{q}$ over outcomes, the utility $X(\vec{q})$ is the expected utility of x wrt. $\vec{q}$. Formally, $X(\vec{q}) = \sum_{i=1}^{m} q_i x(o_i)$. We let D' be the set of these new types. By definition, there is an one-to-one correspondence between types in D and types in D'. Hence, a social choice function $f : D \to \Delta(O)$ corresponds to a (deterministic) social choice function $f' : D' \to \Delta(O)$. Moreover, (resp. given a verification M) f is (resp. M-)truthful-in-expectation iff f' is (resp. M-)truthful.

As before, we seek a general condition under which truthfulness-in-expectation is equivalent to M-truthfulness-in-expectation. Given a verification M, we let M' be the verification corresponding to M in the new domain D'. Formally, for each type $X \in D'$, corresponding to type $x \in D$, $M'(X) = \{Y \in D' : y \in M(x)\}$. Now, the results of Sections 3.1.1, 3.1.2, and 3.2 directly apply to the new domain D' with verification M'. We note that if M is symmetric, then M' is symmetric as well. Hence, for a result that directly applies to the original verification M and domain D, we need a property of the paths in G_M that guarantees that the corresponding paths in $G_{M'}$ are order-preserving.

An $x - y$ path p in G_M is *difference-convex* if for any type w in p, there is a $\lambda \in (0, 1)$, such that for all $a, b \in O$, $w(a) - w(b) = \lambda(x(a) - x(b)) + (1 - \lambda)(y(a) - y(b))$. Similarly, an $x - y$ path p in G_M is *strict difference-convex* if for every type w in p, the subpath of p from x to w is also difference-convex. A verification M is called *symmetric* (resp. *strict*) *difference-convex* if M is symmetric and for any $x, y \in D$, there is a *finite* (resp.

strict) difference-convex $x - y$ path in G_M. For truthfulness-in-expectation, we quantify the utility of each type x for each outcome. Hence, the difference-convex property is a stronger version of the difference-preserving property, which in turn, is a stronger version of the order-preserving property. We now prove that:

Lemma 11. *If an $x - y$ path p in G_M is (resp. strict) difference-convex, then the corresponding $X - Y$ path p' in $G_{M'}$ is (resp. strict) difference-preserving, and thus, (resp. strict) order-preserving.*

Proof. Let p be a difference-convex $x - y$ path in G_M, and let w be any node / type in p. Since p is difference-convex, there exists a $\lambda \in (0, 1)$, such that for all possible outcomes $a, b \in O$,

$$w(a) - w(b) = \lambda(x(a) - x(b)) + (1 - \lambda)(y(a) - y(b))$$

We next show that the corresponding $X - Y$ path p' in $G_{M'}$ is difference-preserving. Namely, that for all probability distributions $\vec{q}, \vec{r} \in \Delta(O)$, and all nodes W in p', if $X(\vec{q}) - X(\vec{r}) \neq Y(\vec{q}) - Y(\vec{r})$, then

$$W(\vec{q}) - W(\vec{r}) \in (\min\{X(\vec{q}) - X(\vec{r}), Y(\vec{q}) - Y(\vec{r})\}, \max\{X(\vec{q}) - X(\vec{r}), Y(\vec{q}) - Y(\vec{r})\}) ,$$

while if $X(\vec{q}) - X(\vec{r}) = Y(\vec{q}) - Y(\vec{r})$, then $W(\vec{q}) - W(\vec{r}) = X(\vec{q}) - X(\vec{r})$.

We first show that for any $\vec{q} \in \Delta(O)$, $W(\vec{q}) = \lambda X(\vec{q}) + (1 - \lambda)Y(\vec{q})$. Using that $q_m = 1 - q_1 - \cdots - q_{m-1}$ we obtain that:

$$\begin{aligned} X(\vec{q}) &= \sum_{i=1}^{m-1} q_i(x(o_i) - x(o_m)) , \quad \text{and} \\ Y(\vec{q}) &= \sum_{i=1}^{m-1} q_i(y(o_i) - y(o_m)) \end{aligned}$$

Multiplying the first equality by λ and the second by $(1 - \lambda)$, summing up, and using the difference-convex property, we obtain that:

$$\begin{aligned} \lambda X(\vec{q}) + (1 - \lambda)Y(\vec{q}) &= \sum_{i=1}^{m-1} q_i[\lambda(x(o_i) - x(o_m)) + (1 - \lambda)(y(o_i) - y(o_m))] \\ &= \sum_{i=1}^{m-1} q_i(w(o_i) - w(o_m)) \\ &= \sum_{i=1}^{m} q_i w(o_i) = W(\vec{q}) \end{aligned}$$

Now, let $\vec{q}$, $\vec{r}$ be any two probability distributions over O. Expressing $W(\vec{q})$ and $W(\vec{r})$ as a convex combination of $X(\vec{q})$ and $Y(\vec{q})$, and $X(\vec{r})$ and $Y(\vec{r})$, respectively, we obtain that

$$W(\vec{q}) - W(\vec{r}) = \lambda(X(\vec{q}) - X(\vec{r})) + (1 - \lambda)(Y(\vec{q}) - Y(\vec{r}))$$

This immediately implies that the $X - Y$ path p' in $G_{M'}$ is difference-preserving. Moreover, the argument above implies that if the $x - y$ path p in G_M is strict difference-convex, the corresponding $X - Y$ path p' in $G_{M'}$ is strict difference-preserving.

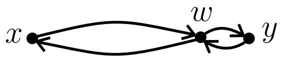
Since (strict) difference-preserving is a stronger property than (strict) order-preserving, we obtain that the $X - Y$ path p' in $G_{M'}$ is also (strict) order-preserving. □

Although the difference-convex property seems quite strong, there present an example where a slight deviation from it results in paths in $G_{M'}$ that are not difference-preserving. In this sense, the difference-convex property and Lemma 11 are tight.

Example : *An Example Showing that Lemma 11 is Tight*

We next present a simple example where a slight deviation from the difference-convex property results in paths in $G_{M'}$ that are not difference / order-preserving. In this sense, the difference-convex property and Lemma 11 are tight.

Let the domain $D = \{x, y, w\}$, let the outcome set $O = \{o_1, o_2, o_3\}$, and let the correspondence graph G_M be:



The types x, y, and w are defined as follows:

	x	w	y
o_1	$1+\varepsilon$	$2-\varepsilon$	$3+\varepsilon$
o_2	0	0	0
o_3	-1	-2	-3

We observe that the path $p = (x, w, y)$ is order-preserving and difference preserving, but it is ε-away from being difference-convex, in the sense that

$$w(o_1) - w(o_2) = (0.5 + \varepsilon)(x(o_1) - x(o_2)) + (0.5 - \varepsilon)(y(o_1) - y(o_2)) \, ,$$

while $w(o_3) - w(o_2) = 0.5(x(o_3) - x(o_2)) + 0.5(y(o_3) - y(o_2))$.

We next show that the path p' corresponding to p in $G_{M'}$ is neither order-preserving nor difference-preserving. To this end, we consider the probability distributions $\vec{q} = (0.5, 0, 0.5)$ and $\vec{r} = (0, 1, 0)$ over O. The expected utility of each type wrt. $\vec{q}$ and $\vec{r}$ is:

	X	W	Y
$\vec{q}$	$\varepsilon/2$	$-\varepsilon/2$	$\varepsilon/2$
$\vec{r}$	0	0	0

Therefore, the corresponding path $p' = (X, W, Y)$ in $G_{M'}$ is not order-preserving, because $X(\vec{q}) > X(\vec{r})$ and $Y(\vec{q}) > Y(\vec{r})$, while $W(\vec{q}) < W(\vec{r})$. Moreover, the path $p' = (X, W, Y)$ in $G_{M'}$ is not difference-preserving, because $X(\vec{q}) - X(\vec{r}) = Y(\vec{q}) - Y(\vec{r}) = \varepsilon/2$, while

$W(\vec{q}) - W(\vec{r}) = -\varepsilon/2$.

□

By the discussion above, Lemma 11, Theorem 7, and Theorem 9 imply that:

Theorem 10. *Let M be a symmetric strict difference-convex verification. Then, truthfulness-in-expectation (resp. with money) is equivalent to M-truthfulness-in-expectation (resp. with money).*

Similarly, we can combine Lemma 11 and Theorem 8, and obtain the equivalent of Theorem 10 for finite domains D, under the symmetric difference-convex property.

3.2 Necessary Conditions for Truthful Implementation

Next, we study relaxed versions of the sufficient conditions in Section 3.1, and show that they are necessary conditions for the equivalence of truthfulness and M-truthfulness.

Deterministic Mechanisms. Given an outcome $a \in O$, we say that an $x - y$ path p in G_M is *a-preserving* if for all outcomes $b \in O$, with $x(a) > x(b)$ and $y(a) \geq y(b)$, and for any intermediate type w in p, $w(a) > w(b)$. Namely, if the endpoints x and y of p agree that a is preferable to b, any intermediate type w in p should also prefer a to b. A verification M is called *symmetric outcome-preserving* if M is symmetric and for all types $x, y \in D$ and all outcomes $a \in O$, there is a *finite* a-preserving $x - y$ path p in G_M. Though quite close to each other, the order-preserving property implies the outcome-preserving property, but not vice versa. Specifically, an a-preserving path p may not be order-preserving, because the relative preference order of some outcomes, other than a, may change in the intermediate nodes of p.

Theorem 11. *Let M be a symmetric verification that is not outcome-preserving. Then, there exists a function g which is M-truthfully implementable, but not implementable.*

Proof. Since M is not outcome-preserving, there exists a pair of types $x, y \in D$ and an outcome $a \in O$, such that any finite $x - y$ path in G_M violates the a-preserving property. Thus, all $x - y$ paths in G_M consist of at least 2 edges (a single edge is trivially order-preserving). Then, we construct a certificate that M is not outcome-preserving, which is a separator of x and y in G_M, and based on this, we define a function g that is M-truthfully implementable, but not truthfully implementable.

For every finite $x - y$ path p in G_M, we let t_p denote the first intermediate type in p and o_p denote an outcome, such that $x(a) > x(o_p) \wedge y(a) \geq y(o_p) \wedge t_p(o_p) \geq t_p(a)$. Namely, for every finite $x - y$ path p, t_p and o_p provide a certificate that p violates the a-preserving property. We let $O_{xy} = \{o_p \in O : p \text{ is a finite } x - y \text{ path}\}$ be the set of outcomes in these certificates, and let $C_{xy} = \{z \in D \setminus \{y\} : \exists b \in O_{xy} \text{ with } z(b) \geq z(a)\}$ be a set of types that can be used as certificates along with the outcomes in O_{xy}.

For convenience, we simply use C instead of C_{xy}. The crucial observation is that for every finite $x - y$ path p in G_M, $t_p \in C$, and thus, C is a separator of x and y in G_M.

Let A be the set of types in the connected component [1] that contains x, obtained from G_M after we remove C, and let $B = D \setminus (A \cup C)$.

Since $y \notin C$, by definition, and for every finite $x - y$ path p, $t_p \in C$, y is in B.

We consider the following function:

$$g(z) = \begin{cases} \arg\max_{b \in O_{xy}} \{z(b)\} & z \in A \cup C \\ a & z \in B \end{cases}$$

By the definition of C, every type in $A \cup B$ prefers a to any outcome in O_{xy}. However, by the definition of A and B, no type $z \in A$ has a neighbor in B, since otherwise, we could find a finite path from x to G_M^y. Therefore, for any $z \in A$, all z's neighbors G_M are in $A \cup C$, and thus $g(z)$ is z's best outcome in its G_M neighborhood. Similarly, every type $z \in C$ prefers any type in O_{xy} to a, and every type $z \in B$ prefers a to any outcome in O_{xy}, by the definition of C. Hence, g is M-truthfully implementable. On the other hand, g is not truthfully implementable, because x prefers a to any outcome in O_{xy}, and thus has an incentive to misreport y, if we do not have any verification. □

Applications

Theorem 11 provides a convenient way of checking whether truthful implementation cannot take any advantage of symmetric verification. E.g., we show that for the domain of 2-Facility Location, M^ε verification is not outcome-preserving, and thus, there are such social choice functions that become truthful with M^ε verification.

Lemma 12. *For any $\varepsilon > 0$, M^ε verification is symmetric but not outcome-preserving for the domain of 2-Facility Location.*

Proof. As before, we know that M^ε verification is symmetric. Therefore, we have only to prove that it is not outcome-preserving. So we have to prove that there are $x, y \in D = \mathbb{R}$ and a pair $a, b \in \mathbb{R}^2$ such that there is a set C which separates x and y in G_{M^ε} and for every $w \in C$:

$$x(a) > x(b) \wedge y(a) > y(b) \wedge w(a) < w(b)$$

for this purpose we set $x = 0$, $y = 5\varepsilon$, $C = [2\varepsilon, 3\varepsilon]$ and $a = [0, 5\varepsilon]^T$, $b = [2\varepsilon, 3\varepsilon]^T$. Clearly C is a separator of x and y and $x(a) = y(a) = 0$, $x(b) = y(b) = -2\varepsilon$ also if we take an arbitrary $w \in C$ $w(a) \leq -2\varepsilon$ and $w(b) \geq -\varepsilon$ and therefore $w(a) < w(b)$ and so we found the separator we need to show that there is no a-preserving path between x and y and so M^ε is not outcome-preserving. □

Deterministic Mechanisms with Money. We obtain a necessary condition for the equivalence of weak and M-weak monotonicity. Given a verification M and $a, b \in O$, an

[1] If D is finite, we use the standard graph-theoretic definition of connected components. If D is infinite, A includes x and all types $w \in D$ reachable from x through a finite path.

$x - y$ path p in G_M is *difference* a, b*-preserving* if for any type w in p, if $x(a) - x(b) \neq y(a) - y(b)$,

$$w(a) - w(b) \in (\min\{x(a) - x(b), y(a) - y(b)\}, \max\{x(a) - x(b), y(a) - y(b)\})$$

A verification M is *symmetric difference outcome-preserving* if M is symmetric and for any types $x, y \in D$ and all outcomes $a \in O$, there is a *finite* difference a-preserving $x - y$ path p in G_M. As before, the difference-preserving property implies the difference outcome-preserving property, but not vice versa. The proof of the following is conceptually similar to the proof of Theorem 11.

Theorem 12. *Let M be a symmetric verification which is not difference outcome-preserving. Then, there is a social choice function g which is M-weakly monotone, but not weakly monotone.*

Proof. Without loss of generality we assume that since the verification is not difference outcome preserving there exists $x, y \in D$ and $a, b \in O$ such that every path p from x to y with $x(b) - x(a) > y(b) - y(a)$ has a type t_p

$$y(b) - y(a) \geq t_p(b) - t_p(a) \vee t_p(b) - t_p(a) \geq x(b) - x(a)$$

Let $C^1_{xy} = \{z \in D \setminus \{y\} : y(b) - y(a) \geq z(b) - z(a)\}$ and $C^2_{xy} = \{z \in D \setminus \{x\} : z(b) - z(a) \geq x(b) - x(a)\}$. For convenience, we simply use C^1 instead of C^1_{xy} and C^2 instead of C^2_{xy}, let also $C = C^1 \cup C^2$. The crucial observation is that for every finite $x - y$ path p, $t_p \in C$ (since $t_p \in C^1$ or $t_p \in C^2$) and thus C is a separator of x and y in G_M. Let A be the set of types in the connected component of G_M which contains x when we remove C from D, let $B = D \setminus A \cup C$. Since for every finite $x - y$ path p $t_p \in C$, y cannot belong to A and by the definition of C, y cannot belong to C therefore $y \in B$. We define

$$g(z) = \begin{cases} a & z \in C^1 \cup A \\ b & z \in C^2 \cup B \end{cases}$$

By the definitions of A and B any point in A cannot have a neighbor in B because then we could find a finite path from x to a type in B which contradicts to the definition of A. Therefore if there exists a negative 2-cycle (u, v, u) in $G_{M,f}$ for some $u, v \in D$ it cannot be the case that $u \in A$ and $v \in B$. Of course by the definition of g we cannot have $u, v \in A \cup C^1$ or $u, v \in B \cup C^2$. So we have to examine the following cases :

- $u \in A, v \in C^2$. By the definition of C^2 and by the fact that $u \notin C^2$ we have $v(b = f(v)) - v(a = f(u)) \geq x(b) - x(a) \geq u(a) - u(b)$ and therefore the 2-cycle (u, v, u) is positive.

- $u \in C^1, v \in B$. By the definition of C^1 and by the fact that $v \notin C^1$ we have $v(b = f(v)) - v(a = f(u)) \geq y(b) - y(a) \geq u(a) - u(b)$ and therefore the 2-cycle (u, v, u) is positive.

Hence g is M-weakly monotone. By since $x(b) - x(a) > y(b) - y(a)$ g is not weakly monotone. □

Sufficient and Necessary Condition. Closing the small gap between the order-preserving and outcome-preserving properties, we present a condition that is both sufficient and necessary for the equivalence of truthful and M-truthful implementation. Given a social choice function f, a $x - y$ path $p = (x = v_0, v_1, \dots, v_k, v_{k+1} = y)$ in G_M is *f-preserving* if for any type v_i, $1 \leq i \leq k+1$ in p, and for all outcomes $a \in O$, with $x(f(v_i)) > x(a)$ and $v_i(f(v_i)) \geq v_i(a)$, $v_{i-1}(f(v_i)) > v_{i-1}(a)$. A verification M is *symmetric function-preserving* if M is symmetric and for any M-truthfully implementable function f and all types $x, y \in D$, there is a *finite* f-preserving $x - y$ path in G_M. We now combine the techniques used in the proofs of Theorem 7 and Theorem 11, and show that:

Theorem 13. *Let M be a symmetric verification. Then, truthful implementation is equivalent to M-truthful implementation if and only if M is function-preserving.*

Theorem 13 is an immediate consequence of the following lemmas:

Lemma 13. *Let M be a symmetric function-preserving verification. Then truthfulness is equivalent to M-truthfulness.*

Proof. Clearly, if a social function is truthfully implementable, it is also M-truthfully implementable. So, we can focus on establishing the converse. The proof is by induction on the length of the paths in G_M. Technically, for sake of contradiction, we assume that there is a social choice function f that is M-truthfully implementable, but not truthfully implementable. Therefore, all edges in the correspondence graph $G_{M,f}$ are non-negative, but there is a negative edge $(x, z) \in E(G_f)$.

Since M is symmetric function-preserving, there is a finite f-preserving $x - z$ path p in $G_{M,f}$. In particular, we let $p = (x = v_0, v_1, v_2, \dots, v_k = z)$, and let i, $2 \leq i \leq k$, be the smallest index such that the edge $(x, v_i) \in E(G_f)$ is negative. For convenience, we let $y = v_i$ and $w = v_{i-1}$. We note that by the definition of i, the edge $(x, w) \in E(G_f)$ is non-negative, and also since f is M-truthfully implementable, the edges $(w, y), (y, w) \in E(G_{M,f})$ are non-negative (see also Fig. 3.1.i).

For convenience, we let $a = f(x)$, $b = f(w)$, $c = f(y)$ denote the outcome of f at x, y, and w, respectively. Since the edge (x, y) is negative, $a \neq c$. Moreover, by the definition of i (and of y), $b \neq c$.

By the discussion above, we have that $x(c) > x(a) \geq x(b)$ and $y(c) \geq y(b)$. Therefore, since the $x - z$ path is f-preserving, and thus its $x - y$ subpath is also f-preserving, we obtain that $w(c) > w(b)$, a contradiction to the hypothesis that the edge $(w, y) \in E(G_{M,f})$ is non-negative. Therefore there is no negative edge in G_f, which implies that f is truthfully implementable. □

Lemma 14. *Let M be a symmetric verification which is not function-preserving. Then there is a social choice function g which is M-truthfully implementable but not truthfully implementable.*

Proof. Since the verification is not function preserving there exists a social choice function f which is M truthfully implementable and $x, y \in D$ such that every path $p = (x = v_0, v_1, v_2, \ldots, v_{k-1}, v_k = y)$ from x to y there exists $i > 0$ and an outcome $a \in O$ such that

$$x(f(v_i)) > x(a) \wedge v_i(f(v_i)) \geq v_i(a) \wedge v_{i-1}(f(v_i)) \leq v_{i-1}(a)$$

We say that a point $v \in D$ is a *separation point* if and only if every path $p = (x = v_0, v_1, \ldots, v_{i-1} = v, \ldots, v_k = y)$ such that the path $p' = (v_i, \ldots, v_k)$ is f preserving part of a $x - y$ path there exists $o_{v_i} \in O$ such that

$$x(f(v_i)) > x(o_{v_i}) \wedge v_i(f(v_i)) \geq v_i(o_{v_i}) \wedge v_{i-1}(f(v_i)) \leq v_{i-1}(o_{v_i})$$

o_{v_i} is called a *separation outcome.* We define C be the set of separation points and O_{xy} be the set of separation outcomes.

Claim. *The set C is a separator of G_M with x and y be in different connected components.*

Proof of the Claim. Suppose we remove C from G_M and we get a new graph G'_M. Suppose that there is a path $p = (x = v_0, v_1, \ldots, v_{k-1}, v_k = y)$ from x to y. We call a vertex v on p *contagious* if there exists an f preserving part of a $x - y$ path from v to y, otherwise we call the vertex *free.*

Then since we suppose that the verification is not f-preserving we know that there is a v_{l-1} and an outcome $a \in O$ such that

$$x(f(v_1)) > x(a) \wedge v_i(f(v_l)) \geq v_l(a) \wedge v_{l-1}(f(v_l)) \leq v_{l-1}(a)$$

let L be the set of such vertices and we take the v_{l-1} which is nearest to y and it is free i.e. has maximum l and it's free. Let's call v_{i-1} and this vertex. For the definition of v_{i-1} we have that every vertex in L which is after v_{i-1} in the path p is contagious which means that the vertex u which is in L and it is the next of v_{i-1} is contagious and this implies that v_i is contagious since the path $(v_i, \ldots, u)$ is by the definition of u f preserving. This proves that v_i is contagious. This means that there is a path $p' = (v_i, \ldots, y)$ which is f preserving. But v_{i-1} is not in C and also is free by its definition. So satisfies the conditions in G'_M to be in C and therefore the only reason to don't be in C is the that there is a point $t \in C$ which is contagious but this contradicts the definition of C. This implies that v_{i-1} have to be in C and the means that the path p shouldn't exist in the G'_M graph. Therefore there cannot be a path from x to y in G'_M which implies that C is a separator of G_M and in G'_M x and y are in different connected components. □

Let B be the set of vertices in the connected component of y in the G'_M graph and A the set $D \setminus C \cup B$. We now define the social choice function

$$g(z) = \begin{cases} f(z) & z \in B \\ \arg\max_{a \in O_{xy}} z(a) & z \in A \cup C \end{cases}$$

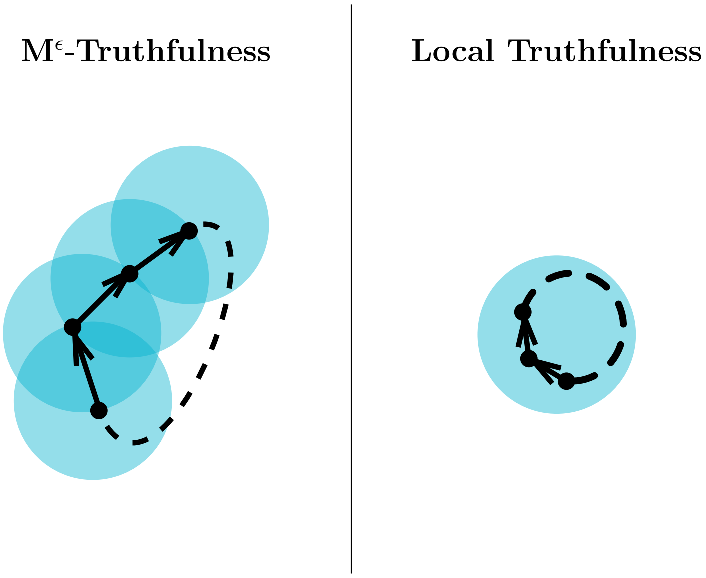
Using the definition of C and the fact that C is a separator of G_M we can easily prove that g is M truthfully implementable. And by the definition of O_{xy} we can see that there exists a type $w \in B$ which x prefers to report. This means that g is not truthfully implementable. □

Therefore the function preserving property gives us the tight property for proving that $M-$truthfulness is equivalent with truthfulness but it is far more complicated than the order preserving and the outcome preserving properties which are still more important in practical applications.

Chapter 4

Local to Global Truthfulness in convex domains [2]

As we saw in the previous chapter there is a general technique for proving that $M-$truthfulness is equivalent with truthfulness. We also notice that in the case of implementability without money and in the case of weak monotonicity property, proving that $M^{\epsilon}-$truthfulness without money (resp. $M^{\epsilon}-$weak monotonicity) implies truthfulness without money (resp. weak monotonicity) is the same as proving that local truthfulness without money (resp. local weak monotonicity) implies truthfulness without money (resp. weak monotonicity). The situation changes when we consider truthfulness with money. In this case $M^{\epsilon}-$truthfulness and truthfulness are different properties. This could be easily understood if we look at the interprentation of these properties using the graph representation. In this framework $M^{\epsilon}-$truthfulness implies that the $G_{M^{\epsilon},f}$ graph has no negative cycles, which means that on the G_f graph there is no cycle consisting only from "small" edges, where with the term "small" we mean that the belong to the graph $G_{M^{\epsilon}}$. On the other hand local truthfulness implies that the G_f graph contains no "small" cycle, where with the term "small" we mean that the entire cycle is in an $\epsilon-$neighborhood of the domain D. This difference becomes more clear in the next picture.



In this chapter we present a proof that local truthfulness also implies truthfulness in convex domains. This proof is the main result of the work of Archer and Kleinberg [2]. In order to prove this Archer and Kleinberg give a characterization of truthfulness using path integrals on the convex domain D. Afterwards the local to global truthfulness comes as a result of the properties of closed integrals on the domain D.

4.1 The characterization Theorem

For simplicity of the analysis in this section we are going to denote $x(f(y))$ as $x \cdot f(y)$. This notation makes some arguments and proof simpler because we can use the $x(f(y))$ as a product $x \cdot f(y)$, but it is clear that all the results could be easily derived even if we use the previous notation. Also we are going to use a sightly different graph representation from the G_f which we denote as H_f.

Definition 29. *For a given social choice function f we define the weighted graph G_f*

$$H_f = (D, D^2, w) \text{ where } w((x, y)) = y(f(y)) - x(f(y))$$

As we can see the deference between G_f and H_f is the weights of the edges and in order to distinguish between them we are going to use the notation $w_G(x, y)$ to refer to the weight of the edge in the G_f graph and $w_H(x, y)$ to refer to the weight of the edge in the H_f graph. Therefore using the notation we have introduced we have that

$$w_G(x, y) = x \cdot (f(x)) - f(y))$$
$$w_H(x, y) = (y - x) \cdot f(y)$$

It is easy to verify that the conditions for cycle monotonicity and weak monotonicity doesn't change in the H_f graph representations. Namely a social choice function f satisfies weak monotonicity if and only if H_f has no negative $2-$cycles and cycle monotonicity if and only if H_f has no negative cycle.

All of the statements and proofs in this section hold in the setting of abstract outcome sets and infinite-dimensional type spaces, provided the notation is interpreted correctly. However, it is conceptually helpful to assume for this discussion that $D, O \subseteq \mathbb{R}^n$. For example, this ensures that dot products have a well defined meaning. As we have already said dot product such as $x \cdot a$ should be interpreted to mean the valuation $x(a)$. This also applies to expressions that need to be expanded using the distributive law, e.g., $(x-y)\cdot(a-b)$ denotes $x(a)-x(b)-y(a)+y(b)$.

Having these in mind we can now give an intuition of what it will be the meaning of the line integrals that we are going to use for the characterization theorem in this section. The interpretation of line integrals in the infinite-dimensional case is as follows. Since every line integral considered in this chapter is defined over a polygonal path consisting of one or more line segments in D, it suffices to define the line integral over a single line segment. If f is a social choice function from D to O, x_0, x_1 are any two types, L is the line segment from x_0 to x_1, and $x_t = (1-t)x_0 + tx_1$, then $\int_L f(x)\cdot dx$ denotes the integral $\int_0^1 (x_1 - x_0)\cdot f(x_t)dt$.

We are now ready to present some useful definitions.

Definition 30. *An allocation function $f : D \mapsto O$ satisfies local weak monotonicity if and only if for every $x \in D$ and every line L through x, there exists an open neighborhood U about x such that*

$$(x-y)\cdot(f(x)-f(y)) \geq 0$$

for all $y \in L \cap U$.

Notice that the definition of local weak monotonicity is equivalent with the definition of $M^{\epsilon}-$weak monotonicity in convex domains. Therefore the only reason for which we present it again here is to become familiar with the notation we are using in this chapter.

Definition 31. *A social choice function $f : D \mapsto O$ is segment integrable if and only if for every closed line segment $L \subseteq D$, the line integral $\int_L f(x)\cdot dx$ is well-defined and finite.*

Although the definition of segment integrable function looks like it restricts the set of functions for which the characterization is going to apply we can prove that every function satisfying the local weak monotonicity property also satisfies the segment integrable property and therefore since we analyze locally truthful functions we lose nothing by assuming that they are also segment integrable.

Lemma 15. *Every social choice function f that satisfies local weak monotonicity is segment integrable.*

Proof. Let L be an arbitrary closed line segment in D with endpoints x_0, x_1. For the purposes of this lemma it is obvious that we can assume that $L = D$. Since D is convex using Lemma 9 and Lemma 10 we can conclude that f satisfies weak monotonicity. At this point we have to prove a technical claim.

Claim. *If a social choice function f is weakly monotone then the for every $x \in D$ and every vector h the function $g(t) = f(x + th) \cdot h$ is non-decreasing on the subset of $\mathbb{R}$ on which it is defined.*

Proof of the Claim. Let $s < t$ be any two real numbers such that the vectors $y = x + sh$, $z = x + th$ belong to D. We have that

$$g(t) - g(s) = (f(z) - f(y)) \cdot h = (f(z) - f(y)) \cdot \frac{(z - y)}{t - s}$$

and therefore since f satisfies weak monotonicity we have that $g(t) \geq g(s)$ for all g, s, t. □

Applying the claim with $x = x_0$ and $h = x_1 - x_0$, we conclude that the function $g(t)$ is monotonically non-decreasing on $[0, 1]$. Observe that $\int_L f(x)dx$ is defined as the integral $\int_0^1 g(t)dt$. The lemma follows because every non-decreasing function on a closed interval is Riemann integrable and has a finite integral. □

Now we are ready to define the property based on line integrals which will us the characterization of truthfulness.

Definition 32. *A segment integrable social choice function $f : D \mapsto O$ is vortex-free if for every $x_0 \in D$ and every $2-$dimensional plane Π through x, there exists an open neighborhood U about x_0 such that the path integral $\oint_\Delta f(x)\Delta dx$ vanishes for every triangle Δ in $\Pi \cap U$ with one corner at x_0.*

The definition of vortex-free implies a seemingly stronger condition :
for every $x_0 \in D$ and every $2-$dimensional plane Π through it, there exists an open neighborhood U about x_0 such that the path integral $\oint_\Delta f(x)\Delta dx$ vanishes for every triangle Δ in $\Pi \cap U$.
To see this, take U as in Definition 32, let x_1, x_2, x_3 be the corners of Δ. Now for $0 \leq i, j \leq 3$, define L_{ij} to be a line segment directed from x_i to x_j and let $W_{ij} = \int_L f(x)dx$. From the definition of vortex-freeness, we know that the loop integral $\oint_{\Delta'} f(x) \cdot dx$ vanishes when Δ' is a triangle contained in $\Pi \cap U$ with one corner at x_0. Thus

$$W_{01} + W_{12} - W_{02} = 0$$

$$W_{02} + W_{23} - W_{03} = 0$$

$$W_{03} + W_{31} - W_{01} = 0$$

where we used the fact that reversing a path negates its line integral. Summing these three equations, we obtain $\oint_\Delta f(x)\Delta dx$, as desired.

We are now ready to explain the characterization of truthfulness using line integrals.

Theorem 14. *Let D be a convex domain and $f : D \mapsto O$ be a social choice function. Then f is truthful if and only if it is vortex-free and satisfies local weak monotonicity.*

We now express the idea behind the proof of the characterization theorem. Since we have already show in the previous chapter that local weak monotonicity implies weak monotonicity in convex domains we have only to prove that vortex-freeness implies that there is no negative cycle between three or more types in D.

From the observation we have make after defining the vortex-free property it is easy to show that vortex-freeness implies that the line integral on every triangle in D vanishes. Using again the same argument we can prove that assuming vortex-freeness is equivalent with the hypothesis that the line integral over any closed polygon in D vanishes.

Now using the same machinery as in the proof of the Theorem 9 we can have that the total weight of a cycle C is greater that or equal to the total weight of a cycle C' where C' contains the vertices of C and some more vertices on the difference preserving paths between consequtive points of C. Since in convex domains the difference preserving path between to points is the line segments between these points we can have that the value of the line integral on the polygon defined by the points of C has value less than or equal to the total weight of C. Therefore if there is a negative cycle in H_f there must also be a line integral on a polygon which is not zero. This implies that vortex-freeness is violated. Therefore vortex-freeness implies truthfulness.

For the other direction we suppose that there is no negative cycle in H_f graph and that the line integral on a polygon is non-zero then by taking the one or the other direction to the line integral we can find a polygon on which the line integral is negative. But since the line integral is the limit of Riemann sums there must be a finite number of types in D such that the weight of their cycle in H_F is negative which contradicts to our hypothesis. Therefore truthfulness implies that the integral on every polygon in D vanishes which is equivalent as we have said with the vortex-free property. So truthfulness implies vortex-freeness.

Having in mind this proof sketch of Theorem 14 we prove some useful lemmas and then we give the entire proof of Theorem 14.

Lemma 16. *Let $g : [0, 1] \mapsto \mathbb{R}$ be an increasing function, and let $0 = x_0 < x_1 < \cdots < x_N = 1$ and $0 = y_0 < y_1 < \cdots < y_M = 1$ be two increasing sequences such that $(y_j)_{j=0}^M$ refines $(x_i)_{i=0}^N$, i.e., (x_i) is a subsequence of (y_j). Then the right-hand Riemann sums of g with respect to (x_i), (y_j) satisfy the inequality*

$$\sum_{i=1}^{N}(x_i - x_{i-1})g(x_i) \geq \sum_{j=1}^{M}(y_j - y_{j-1})g(y_j)$$

Proof. It suffices to prove the lemma in the case $M = N + 1$ because the general case then follows by induction. So assume that for some r we have $x_{r-2} = y_{r-1}$ and $x_r = y_r$. We will use the notation Δ_i^x (resp. Δ_i^y) to denote $x_i - x_{i-1}$ (resp. $y_i - y_{i-1}$). In the sum on the left side of the wanted relation the $i = q$ term on the left side matches the $j = q$ term on the right side for $q \leq r$ and it matches the $j = q + 1$ term on the right side for $q \geq r + 2$. Hence

$$\begin{aligned}
\sum_{i=1}^{N} \Delta_i^x g(x_i) - \sum_{j=1}^{M} \Delta_j^y g(y_j) = \\
&= \Delta_r^x g(x_r) - \Delta_r^y g(y_r) - \Delta_{r-1}^y g(y_{r-1}) \\
&= (y_r - y_{r-2})g(y_r) - \Delta_r^y g(y_r) - \Delta_{r-1}^y g(y_{r-1}) \\
&= \Delta_{r-1}^y (g(y_r) - g(y_{r-1})) \geq 0
\end{aligned}$$

□

From the above general lemma we consider a specific case which we are going to use to the proof of the Theorem 14.

Corollary 4. *If $f : D \mapsto O$ is a social choice function that satisfies weak monotonicity, and L is a line segment in D with endpoints x, y then*

$$(y - x) \cdot f(y) \geq \int_L f(z)dz$$

Proof. We will apply the previous lemma to the function $g(t) = f(x + t(y - x)) \cdot (y - x)$, which we have already proved that it is increasing by Lemma 15. For non-negative integers k and $i \leq 2^k$, let $x_k = i/2^k$. Note that for each k, $0 = x_0^k < x_1^k < \cdots < x_k^k = 1$, and that $(x_j^{k+1})_{j=0}^{2^{k+1}}$ refines $(x_i^k)_{i=1}^{2^k}$. By the previous lemma, the sequence of Riemann sums

$$S_k = \sum_{i=1}^{2^k} (x_i^k - x_{i-1}^k) g(x_i^k)$$

is decreasing. Moreover, by the definition of the Riemann integral, $\int_L f(z)dz = \lim_{k\to\infty} S_k$. Hence

$$(y - x) \cdot f(y) = S_0 \geq \lim_{k\to\infty} S_k = \int_L f(z)dz$$

□

We are now ready to prove the characterization Theorem 14.

Lemma 17. *If $f : D \to O$ is vortex-free, then for every triangle Δ contained in D, the path integral $\oint_\Delta f(x)dx$ vanishes.*

Proof. For clarity, in this proof we will distinguish between triangles (sets consisting of three points and the three line segments joining them) and $2-$*simplices* (the convex hull of three points). We will use the following geometric fact : if σ_1, σ_2 are $2-$simplices with disjoint interiors which share a side in common, and the boundaries of σ_1, σ_2 are triangles Δ_1, Δ_2 oriented consistently, then Δ_1 and Δ_2 traverse the common side of σ_1, σ_2 in opposite directions.

Let V be the $2-$simplex consisting of and its interior. The definition of vortex-free implies that V has an open covering $\{U_i \mid i \in I\}$ such that for every i and every triangle contained in U_i , the integral $\oint_{\Delta'} f(x)dx$ vanishes. Because V is compact, we can apply the Lebesgue number lemma [11] to deduce that there is a $\delta > 0$ such that every set of diameter less than δ is contained in one of the sets U_i. We can subdivide V into $2-$simplices $\sigma_1, \sigma_2, \dots, \sigma_N$ of diameter less than δ and let Δ_i be a closed curve tracing out the boundary of σ_i; assume every Δ_i is oriented consistently with a single, fixed orientation of D . If we write

$$0 = \sum_{i=1}^{N} \oint_{\Delta_i} f(x)dx$$

and break each loop integral on the right side into a sum of three integrals along line segments forming the boundary of σ_i, then each such line segment appears either

- twice with opposite orientations, if it is on the common boundary between two $2-$simplices σ_i, σ_j,
- once, if it is a subset of Δ.

Terms of the first type cancel each other out, while those of the second type sum up to $\oint_\Delta f(x)dx$. Thus $\oint f(x)dx = 0$, as claimed. □

Proof of Theorem 14. First assume f is truthful. Hence it satisfies cycle monotonicity. This immediately implies weak monotonicity and therefore local weak monotonicity. To see that f is vortex-free, we argue by contradiction, i.e., we will show that if f is not vortex-free then it fails to satisfy cycle monotonicity. Assuming f is not vortex-free, there is a triangle Δ such that $\oint_\Delta f(x)dx \neq 0$. Resersing the orientation of Δ if necessary, we may assume that $\oint_\Delta f(x)dx < 0$. Since the integral is the limit of Riemann sums, there mist be a negative sum, i.e., a sequence of points $x_1, x_2, \dots, x_N$ in D such that

$$\sum_{i=1}^{N} f(x_i) \cdot (x_i - x_{i-1}) < 0$$

where the indices are interpreted modulo N. This sequence constitutes a negative cycle.

Conversely, suppose f is vortex-free and satisfies local weak monotonicity. We will prove that f satisfies cycle monotonicity, from which it follows that f is truthful. For any sequence of type vectors $x_0, x_1, \dots, x_N = x_0$, let $L_{ij} : 0 \leq i < j \leq N$ denote the set of paths $L_{ij}(t) = (1-t)x_i + tx_j$, i.e. L_{ij} traces out a line segment from x_i to x_j. If P is the polygonal closed curve formed by concatenating $L_{01}, L_{12}, \dots, L_{(N-1)N}$ then Corollary 4 implies

$$\begin{aligned}\oint_P f(x)dx &= \sum_{i=1}^{N} \int_{L_{(i-1)i}} f(x)dx \\ &\leq \sum_{i=1}^{N} f(x_i) \cdot (x_i - x_{i-1})\end{aligned}$$

so to prove cycle monotonicity (i.e. that the sum on the right side of the above equation is non-negative) it suffices to prove that $\int_P f(x)dx = 0$. For $0 \leq i < j \leq N$ let

$$W_{ij} = \int_{L_{ij}} f(x)dx$$

Since integrating along a curve in the opposite direction negates the value of the integral, we have $W_{ij} = -W_{ji}$. For $i = 1, 2, \dots, N-2$ let T_i denote the triangle formed from L_{0i}, $L_{i(i+1)}$, $L_{(i+1)0}$. Lemma 17 implies that

$$0 = \oint_{T_i} f(x)dx = W_{0i} + W_{i(i+1)} + W_{(i+1)0}$$

Interpreting the subscripts mod N and summing previous equation as i runs from 1 to $N-2$ yields

$$\begin{aligned}0 &= \sum_{i=1}^{N-2} W_{0i} + W_{i(i+1)} + W_{(i+1)0} \\ &= \sum_{i=1}^{N} W_{i(i+1)} + \sum_{i=2}^{N-2} W_{0i} + W_{i0} \\ &= \sum_{i=1}^{N} W_{i(i+1)} = \oint_P f(x)dx\end{aligned}$$

□

Now we are ready to present the main consequence of the characterization theorem which also could be read in terms of verification.

Corollary 5. *If D is a convex domain then every social choice function f is truthful if and only if it is locally truthful.*

Another important consequence is based on the observation that vortex-freeness in a property on triangles in D. This means that truthfulness is a property which takes into account only the restriction of f on the $2-$dimensional subsets of D.

Corollary 6. *If D is convex and the restriction of f to $\Pi \cap D$ is truthful for every $2-$dimensional affine subspace Π, then f is truthful.*

This means we can tell that truthfulness with money is a $2-$dimensional property. It is easy to see that truthfulness without money and weak monotonicity are $1-$D properties. Basically this is the reason for which the techniques described in the previous chapter work very well for proving local to global conditions for truthfulness without money and weak monotonicity but didn't work well for truthfulness with money. Therefore if we want to extend our techniques of the previous chapter in order to capture also truthfulness with money, we have to extend the previous techniques in a $2-$dimensional sense.

Chapter 5

Positive results using asymmetric verification

5.1 On the Power of Asymmetric Verification

Intuitively, one should expect that asymmetric verification is powerful due to requirement that the correspondence graph should be acyclic. In fact, if we consider any asymmetric verification M, since G_M does not have any negative cycles, Rochet's theorem implies that any social choice function f is M-truthfully implementable with money. We next show a natural characterization of the social choice functions that can be M-truthfully implemented (without money), for some asymmetric verification M.

Theorem 15. *Let f be any social choice function. There is an asymmetric verification M such that f is M-truthfully implementable iff G_f^- is a directed acyclic graph.*

Proof. Let M be an asymmetric verification that truthfully implements f. Hence, G_M is an acyclic tournament and $G_{M,f}$ does not contain any negative edges, i.e., any edges of G_f^-. Therefore, if we arrange the vertices of G_f (i.e., the types of D) on the line according to the (unique) topological ordering of $G_{M,f}$, all edges of G_f not included in $G_{M,f}$ are directed from right to left. Therefore, the edges of G_f^- cannot form a cycle.

For the converse, let f be a social choice function with an acyclic G_f^-. We consider a topological ordering of G_f^- and remove any edge of G_f directed from left to right. This removes all edges of G_f^- and leaves an acyclic correspondence subgraph G'_f, since all its edges are directed from right to left. Moreover, for every pair of types x, y, we remove one of the edges (x, y) and (y, x). Hence, G'_f is an acyclic tournament without any negative edges. Therefore, f is M-truthfully implementable for the asymmetric verification M corresponding to G'_f. □ □

Reasonable social choice functions should have an acyclic G_f^-. This is true for all functions maximizing the social welfare and all functions truthfully implementable with money. Although one may construct examples of functions f where G_f^- contains cycles, such functions (and such cycles) are hardly natural. For instance, a 2-cycle (x, y, x) in G_f^-

indicates that type x prefers outcome $f(y)$ to $f(x)$, while type y prefers outcome $f(x)$ to $f(y)$. But then, one may change f to f', with $f'(x) = f(y)$, $f'(y) = f(x)$, and $f'(z) = f(z)$ for any other type z. Thus, one eliminates the cycle (x, y, x) and the social welfare is strictly greater using f' allocation.

To further demonstrate the power of asymmetric verification, we extend, in the Appendix, the construction in the proof of Theorem 15 to a *universal asymmetric verification*, which can truthfully implement any social choice function with acyclic G_f^-.

5.1.1 Implementation using Asymmetric Verification for Facility Location

In this section, we apply the idea behind universal asymmetric verification, and show that in the Facility Location domain, the function $F_{\max}(\vec{x}) = (\min \vec{x} + \max \vec{x})/2$, that minimizes the maximum distance of the agents to the facility, can be truthfully implemented with verification $M_{\max}(x_i) = \{y : |y - F_{\max}(\vec{x}_{-i})| \leq |x_i - F_{\max}(\vec{x}_{-i})|\}$.

$$F_{\max}(\mathbf{x}) = \frac{\min(\mathbf{x}) + \max(\mathbf{x})}{2}$$

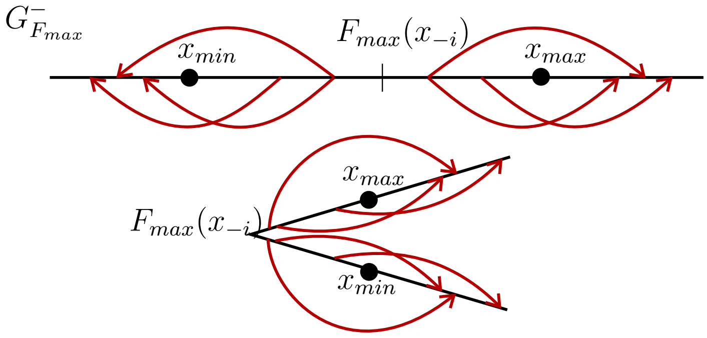
We firstly have to found the graph $G_{F_{\max}}^-$ for an arbitrary agent i which means that all the other players have fixed positions $\mathbf{x}_{-i}$. Let $x_{min} = \min \mathbf{x}_{-i}$ and $x_{max} = \max \mathbf{x}_{-i}$ then we have the following cases.

- $x_i < F_{\max}(\mathbf{x}_{-i})$: In this case, for every $x \in [2x_i - x_{max}, \min(x_i, x_{min})]$, the edge (x_i, x) in the $G_{F_{\max}}$ graph has negative weight and every other edge starting from x_i has positive weight.

- $x_i > F_{\max}(\mathbf{x}_{-i})$: In this case, for every $x \in [\max(x_i, x_{max}), 2x_i - x_{min}]$, the edge (x_i, x) in the $G_{F_{\max}}$ graph has negative weight and every other edge starting from x_i has positive weight.

- $x_i = F_{\max}(\mathbf{x}_{-i})$: In this case, every edge starting for x_i has positive weight.

The above cases describe completely the graph $G_{F_{\max}}^-$ and it is easy to see that this graph is acyclic. Therefore, we can apply Theorem 15, and define an asymmetric verification which implements $F_{\max}$. To do this, we have first to find a topological ordering of $G_{F_{\max}}^-$, with $F_{\max}(\mathbf{x}_{-i})$ as the source. The topological ordering can be viewed in the following figures:



Therefore the verification which implements the $F_{\max}$ is the following:

$$M_{max}(x_i) = \{y \in \mathbb{R} : |y - F_{max}(\mathbf{x}_{-i})| \leq |x_i - F_{max}(\mathbf{x}_{-i})|\}$$

Clearly if the agent i hasn't the minimum or the maximum position of all the agents then $F_{max}(\mathbf{x}_{-i}) = F_{max}(\mathbf{x})$. In the case agent i has the maximum position on the line then the set of $y \in \mathbb{R}$ with $y > x_i$ are the only point agent i has an incentive to misreport and is a subset of the point for which $|y - F_{max}(\mathbf{x})| \leq |x_i - F_{max}(\mathbf{x})|$. This implies that the following verification also implements $F_{\max}$:

$$M_{max}(x_i) = \{y \in \mathbb{R} : |y - F_{max}(\mathbf{x})| \leq |x_i - F_{max}(\mathbf{x})|\}$$

We note that $F_{\max}$ cannot be approximated within a factor less than 2 (resp. 3/2) by deterministic (resp. randomized) mechanisms without money [15].

5.1.2 Implementation using Asymmetric Verification for Strategic Voting

We now show that, in the domain of Strategic Voting, Plurality can be truthfully implemented by an asymmetric verification where the voters are not allowed to misreport a higher preference for the winner of the election. Similarly, we show that Borda Count can be truthfully implemented by an asymmetric verification where the voters are not allowed to misreport either a higher preference for the winner of the election or a lower preference for some of the remaining candidates.

In this domain we are interested in implementing the following social choice functions.

- **Plurality function $\mathbf{F_{pl}}$** : this function for every candidate–outcome a counts the number of voters–agents having a first in their order and sets winner the candidate with the maximum such number. So let $O_a = \{i \mid i \in V,\ v_i(a) = 1\}$ and $n_a = |O_a|$ then $F_{pl}(v) = \arg\max_{a \in O}\ n_a$

- **Borda count function $\mathbf{F_{brd}}$** : this function sets winner the candidate a which minimizes the $\sum_i v_i(a)$, i.e. $F_{brd}(v) = \arg\min_{a \in O} \sum_{i \in V} v_i(a)$

Implementation of $\mathbf{F_{pl}}$. It is easy to see that a voter has incentive to misreport that a candidate b is her first option if she prefers b from the real winner a. Therefore the asymmetric verification which implements F_{pl} is the verification which forbids the overbid of the winner candidates. This means that after the voting the verification has to check if the voters have overbid the winner candidate. Let w be the winner of the voting, i.e. $F_{pl}(v) = w$

$$M_{pl}(i) = \{j : j \in V,\ v_j(w) \leq v_i(w)\}$$

Implementation of $\mathbf{F_{brd}}$. This a more difficult case of social choice function because a voter has an incentive to misreport both the winner of the voting $w = F_{brd}(v)$ and the winner without her $w_{-i} = F_{brd}(v_{-i})$. Therefore we have to forbid both the overbid of w and the underbid of w_{-i}.

$$M_{brd}(i) = \{j : j \in V,\ v_j(w_{-i}) \geq v_i(w_{-i}),\ v_j(w) \leq v_i(w)\}$$

5.1.3 Asymmetric Verification and Payments

The absence of negative cycles in G_f implies the absence of cycles in G_f^-. Thus, Theorem 15, combined with Rochet's theorem, shows that for any function f truthfully implementable with money, there is an asymmetric verification M that truthfully implements f. Next, we show that such an asymmetric verification M can be directly obtained from any payment scheme that implements f. The proof is an extension of the proof of Theorem 15.

Proposition 9. *Let f be a social choice function truthfully implementable by payments $p : D \mapsto \mathbb{R}$. Then, removing all edges $(x, y) \in E(G_f)$ with $p(f(x)) > p(f(y))$ results in an asymmetric verification M that truthfully implements f (without money).*

Proof. Let $p : D \mapsto \mathbb{R}$ be any set of payments that truthfully implements f. We arrange all types on the line in decreasing order of the corresponding payments by p. Precisely, type x precedes (i.e., is on the left of) type y if $p(f(x)) > p(f(y))$, with ties broken in an arbitrary (but fixed) way. Then, we observe that any negative edge $(x, y) \in E(G_f^-)$ is directed from left to right. Indeed, since the edge (x, y) is negative, we have that $x(f(y)) > x(f(x))$. Moreover, since the p truthfully implements f, $x(f(x)) + p(f(x)) \geq x(f(y)) + p(f(y))$. Therefore, $p(f(x)) > p(f(y))$, which indeed implies that (x, y) is directed from left to right in the linear ordering of D. Thus, as in the proof of Theorem 15, removing any edge of G_f directed from left to right, i.e., all edges (x, y) with $p(f(x)) > p(f(y))$, eliminates all edges of G_f^- and leaves an acyclic tournament that corresponds to an asymmetric verification M that truthfully implements f. □ □

Chapter 6

Probabilistic Verification [6]

Motivated by the negative results presented in Chapter 3, Caragiannis et al. proceed to study a broader class of verification settings. The starting point is the observation that in the partial verification model of Green and Laffont [10] lie detection is fully deterministic: either an agent of type v can declare a type $v = v'$ without any risk of being caught, or he simply cannot declare v' as his type. However, in many real-life scenarios lie detection is probabilistic: an agent can report any type v' that differs from his true type v, and is caught with a certain probability. In this model the probability may depend on both v' and v. If a lie is detected, the lying agent is usually punished: if the center was supposed to pay the agent, the payment may be withheld, and the agent may have to pay a *fine*; again, the fine may depend on the agent's true type, the declared type, or both. Because of the existence of the fine all the mechanisms discussed in this chapter refer to implementation with money.

For instance, consider a member of a decision-making body (let us call him Mr. X) who is supposed to vote by submitting his ranking of several alternatives, such as budget proposals or nominees for an administrative post. Let us denote the available alternatives by A, B, and C, and suppose that Mr. X's true ranking of the alternatives is A ¿ B ¿ C. Moreover, he once wrote a private e-mail in which he argued that A is preferable to B, and on another occasion he told a group of supporters that he prefers A to C. Now, if Mr. X votes B ¿ A ¿ C for strategic reasons, his reputation may be damaged if that private e-mail of his is leaked, Thus, when he weighs the cost and the benefits of the strategic vote, he must take into account the leakage probability. Voting B ¿ C ¿ A is even more dangerous, as there is an additional risk that the position he expressed when talking to his supporters becomes publicly known.

In this chapter we provide a formal model for such scenarios by explaining the framework introduced here [6] of mechanism design with *probabilistic verification*. This model allows for probabilistic lie detection and fines, and can be shown to generalize the partial verification model. We characterize the set of social choice functions that can be truthfully implemented in this model; the proof is based on a modification of the graph representation of mechanism design. These results indicate that probabilistic verification can be very powerful. In particular, whenever all lie detection probabilities are strictly positive, any social choice function can be implemented: intuitively, if payments are large enough, even a small chance of not receiving them makes a player reluctant to lie.

6.1 Model and Characterization

We assume that for each pair of types $u, v \in D$ we are given a pair of numbers $\lambda[u, v] \in [0, 1]$ and $\psi[u, v] \in \mathbb{R}^+ \cup \{0, +\infty\}$, $\lambda[u, v]$ is the probability that a player with type u can report type v and not get caught, and $\psi[u, v]$ is the fine that a player of type u has to pay when he is caught reporting v. We require $\lambda[u, u] = 1$ for all $u \in D$, and write $\Lambda = \{\lambda[u, v]\}_{u,v\in D}$, $\Psi = \psi[u, v]_{u,v\in D}$. We refer to Λ as the verification probability matrix and to Ψ as the fine matrix. We assume that the outcome is chosen according to the declared type and the agent enjoys the utility associated with this outcome, but he only gets paid/does not get fined if the lie is not detected. This is motivated by applications such as scheduling, where lie detection typically takes place after an assignment of jobs to machines has been determined, but before the payments are distributed. That is, under a mechanism (g, p) the expected utility of an agent with type u who reports v is

$$U_{(g,p)}(u, v) = u(g(v)) + \lambda[u, v]p(v) - (1 - \lambda[u, v])\psi[u, v]$$

Definition 33. *Given an outcome space O, a domain D, a verification probability matrix $\Lambda = \{\lambda[u, v]\}_{u,v\in D}$, a fine matrix $\Psi = \psi[u, v]_{u,v\in D}$, and a social choice function $f : D \mapsto O$, a mechanism (g, p) is said to $(\Lambda, \Psi)-$implement f if for every $u \in D$ there exists a $v \in D$ such that*

$$g(v) = f(u)$$

$$U_{(g,p)}(u, v) \geq U_{(g,p)}(u, u) foreachu \in D.$$

If Ψ is the all-zero matrix, we omit it from the notation, and say that (g, p) $\Lambda-$implements f. In words, the expected utility of the agent when declaring a type v with $g(v) = f(u)$ must be at least as high as for any other declaration. A $(\Lambda, \Psi)-$implementation is said to be truthful if $f = g$ and condition i.e.,

$$u(f(u)) + p(u) \geq u(f(u)) + \lambda[u, u']p(u) - (1 - \lambda[u, u'])\psi[u, u'] forallu, u' \in D$$

This condition can be rewritten as

$$\lambda[u, v]p(v) - p(u) \leq c[u, v] forallu, v \in D$$

where $c[u, v] = u(f(u)) - u(f(v)) + (1 - \lambda[u, v])\psi[u, v]$

Our probabilistic model generalizes the partial verification model (and, hence, also the classic model): a misreport graph G_M can be simulated by setting $\lambda[u, v] = 1$, $\psi[u, v] = 0$ if $(u, v) \in E[G_M]$ and $\lambda[u, v] = 0$, $\psi[u, v] = +\infty$ otherwise. We denote the resulting verification probability matrix by Λ^M and the fine matrix by Ψ^M. Note that in this construction the fine matrix Ψ^M cannot be replaced by the all-zero matrix, i.e. a mechanism that $M-$implements f does not necessarily λ^M-implement it. Indeed, in the absence of fines, if $u(g(v)) > u(g(u')) + p(u')$ for some $v \in M(u)$ and all $u' \in M(u)$, an agent of type u would prefer to report v, even if he knows for sure that he will be denied payment. However, there is an interesting special case of the probabilistic verification model where fines are not necessary. Namely, suppose that all outcomes are associated with tasks, so that the agent incurs a cost for each outcome, i.e., $v(o) \leq 0$ for all $o \in O$ and all $v \in D$. Let M be a misreport correspondence, and let (g, p) be an individually rational mechanism that $M-$implements some function f; we can assume that $p(v) \geq 0$ for all $v \in D$. Then (g, p) is an individually rational λ^M-implementation of f. Indeed, if an agent of type u reports a type $v \in M(u)$, he will be detected and his utility will be $u(g(v)) \leq 0$, whereas if he reports v, the individual rationality of (g, p) guarantees him a non-negative utility.

We will now show that a similar characterization can be obtained for probabilistic verification under some mild conditions on the domain D and the matrix Λ. Namely, we will assume that D is bounded, i.e., $-C < v(o) < C$ for some sufficiently large constant $C > 0$ and all $v \in D$, $o \in O$, and all elements of Λ are either equal to 1 or are bounded away from 1, i.e., there exists an $\epsilon > 0$ such that for all $u, v \in D$ the inequality $\lambda[u, v] < 1$ implies $\lambda[u, v] \leq 1 - \epsilon$. Note that both of these restrictions trivially hold if the domain D is finite. We will modify Rochet's construction as follows. Given a social choice function f and a probability verification matrix $\Lambda = \{\lambda[u, v]\}_{u,v \in D}$, we construct a graph G_f^Λ from G_f by setting edge weight $c^\Lambda[u, v] = +\infty$ for all edges (u, v) with $\lambda[u, v] < 1$ and $c^\Lambda[u, v] = c[u, v]$ otherwise. Now, the set of social choice functions that are $(\Lambda, \Psi)-$implementable can be characterized as follows.

Theorem 16. *Suppose that D is bounded and all entries of Λ are either equal to 1 or are bounded away from 1. Then a social choice function f is truthfully $(\Lambda, \Psi)-$implementable if and only if G^λ has no negative cycle.*

Bibliography

[1] A. Archer and R. Kleinberg. Characterizing truthful mechanisms with convex type spaces. *ACM SIGecom Exchanges*, 7(3), 2008.

[2] A. Archer and R. Kleinberg. Truthful germs are contagious: A local-to-global characterization of truthfulness. In *Proc. of the 9th ACM Conference on Electronic Commerce (EC '08)*, pp. 21-30, 2008.

[3] V. Auletta, P. Penna, G. Persiano, and C. Ventre. Alternatives to truthfulness are hard to recognize. *Autonomous Agents and Multi-Agent Systems*, 22(1):200-216, 2011.

[4] V. Auletta, R. De Prisco, P. Penna, and G. Persiano. The power of verification for one-parameter agents. *Journal of Computer and System Sciences*, 75:190-211, 2009.

[5] A. Berger, R. Müller, and S.H. Naeemi. Characterizing incentive compatibility for convex valuations. In *Proc. of the 2nd International Symposium on Algorithmic Game Theory (SAGT '09)*, LNCS 5814, pp. 24-35, 2009. Updated version as Research Memoranda 035, Maastricht Research School of Economics of Technology and Organization.

[6] I. Caragiannis, E. Elkind, M. Szegedy, and L. Yu. Mechanism design: from partial to probabilistic verification. In *Proc. of the 13th ACM Conference on Electronic Commerce (EC '12)*, pp. 266-283, 2012.

[7] G. Carroll. When are local incentive constraints sufficient? *Econometrica*, 80(2):661-686, 2012.

[8] D. Fotakis and C. Tzamos. Winner-imposing strategyproof mechanisms for multiple Facility Location games. *Theoretical Computer Science*, 472:90-103, 2013.

[9] D. Fotakis and E. Zampetakis Truthfulness Flooded Domains and the Power of Verification for Mechanism Design. *Conference on Web and Internet Economics*, 2013

[10] J. Green and J. Laffont. Partially verifiable information and mechanism design. *Review of Economic Studies*, 53(3):447-456, 1986.

[11] H. Gui, R. Müller, and R.V. Vohra. Dominant strategy mechanisms with multidimensional types. *Discussion Paper 1392*, Northwestern University, Center for Mathematical Studies in Economics and Management Science, 2004.

[12] P. Krysta and C. Ventre. Combinatorial auctions with verification are tractable. In *Proc. of the 18th European Symposium on Algorithms (ESA '10)*, LNCS 6347, pp. 39-50, 2010.

[13] N. Nisan. Introduction to Mechanism Design (for Computer Scientists). *Algorithmic Game Theory*, Chapter 9:209-241, 2007.

[14] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166-196, 2001.

[15] A.D. Procaccia and M. Tennenholtz. Approximate mechanism design without money. In *Proc. of the 10th ACM Conference on Electronic Commerce (EC '09)*, pp. 177-186, 2009.

[16] J.C. Rochet. A necessary and sufficient condition for rationalizability in a quasi-linear context. *Journal of Mathematical Economics*, 16(2):191-200, 1987.

[17] M.E. Saks and L. Yu. Weak monotonicity suffices for truthfulness on convex domains. In *Proc. of the 6th ACM Conference on Electronic Commerce (EC '05)*, pp. 286-293, 2005.

[18] S. Sato. A sufficient condition for the equivalence of strategy-proofness and nonmanipulability by preferences adjacent to the sincere one. *Journal of Economic Theory*, 148:259-278, 2013.

[19] R.V. Vohra. *Mechanism Design: A Linear Programming Approach.* Cambridge University Press, 2011.