UNIVERSITÀ
DEGLI STUDI
FIRENZE

Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Tesi di Laurea

TEORIA DEI GIOCHI E MULTI-PARTY COMPUTATION

GAME THEORY AND MULTI-PARTY COMPUTATION

FRANCESCO MUCCI

Relatore: *Michele Boreale*

Anno Accademico 2014-2015

A Giulia,
che ogni giorno mi regala il sorriso.

"Quando uno legge uno scritto di cui vuole conoscere il senso, non ne disprezza i segni e le lettere, né li chiama illusione, accidente e involucro senza valore, bensì li decifra, li studia e li ama, lettera per lettera." Hermann Hesse, "Siddharta", 1922 [1].

INDICE

1

INTRODUZIONE

La *Teoria dei Giochi* studia i modelli matematici che descrivono l'interazione, conflittuale o cooperativa, tra più soggetti intelligenti e razionali [2]; tali modelli, chiamati *giochi*, rappresentano classi di situazioni reali in modo altamente astratto, caratteristica che ci consente di sfruttarli per studiare un'ampia gamma di fenomeni [3]: sono, infatti, utilizzati in economia, scienze politiche, psicologia, logica, informatica e biologia [4].

La Teoria dei Giochi è un campo di studio relativamente giovane: alcune sue idee possono essere ricondotte al diciottesimo secolo, ma il maggiore sviluppo della teoria è iniziato a partire dal 1920 con l'opera del matematico Emile Borel (1871-1956) e del poliedrico John von Neumann (1903-1957). Un evento decisivo nello sviluppo della teoria è stato la pubblicazione nel 1944 di *"Theory of games and economic behavior"* di von Neumann e Oskar Morgenstern [5]; difatti, a partire dal 1950, modelli della Teoria dei Giochi iniziano ad essere utilizzati nei campi dell'economia teorica, delle scienze politiche e della psicologia. Sempre nel 1950, il matematico John F. Nash (1928-2015) formula la prima versione del concetto di *equilibrio di Nash*: informalmente, Nash dimostra l'esistenza di una situazione di "equilibrio" in un'interazione strategica non cooperativa, cioè una situazione in cui nessuno dei partecipanti a tale interazione ha interesse ad essere l'unico a cambiare la propria scelta strategica; il riconoscimento definitivo per questi studi giunge nel 1994, quando, insieme ai colleghi John C. Harsanyi (1920-2000) e Reinhard Selten (1930-), riceve il premio Nobel per l'economia. A partire dal 1970, i modelli della Teoria dei Giochi vengono utilizzati come strumento per lo studio della biologia evolutiva e successivamente cominciano ad essere studiati ed applicati a molteplici campi del sapere [6].

L'informatica ha iniziato a sfruttare i principi della Teoria dei Giochi solamente di recente. Il campo della *Teoria dei Giochi Algoritmica (Algorithmic Game Theory)*, introdotto da Nisan e Ronen in [7], si pone l'obiettivo di progettare algoritmi in ambienti strategici: ovvero, in un ambiente di calcolo distribuito in cui i partecipanti, piuttosto che seguire ciecamente il dato algoritmo distribuito, seguiranno il proprio interesse personale. Possiamo, ad esempio, considerare il caso in cui l'input per un dato algoritmo sia distribuito tra molti utenti che hanno un interesse personale nell'output; in tale situazione, i partecipanti alla computazione, spinti dai propri interessi, potrebbero non condividere i propri input in modo corretto: sarà compito del progettista dell'algoritmo trovare il modo di incentivare i partecipanti a comportarsi in modo corretto. È possibile

approcciarsi alla Teoria dei Giochi Algoritmica secondo due diversi punti di vista:

- *Analisi.* Analizzare algoritmi esistenti utilizzando strumenti della Teoria dei Giochi.
- *Progettazione.* Progettare *giochi* per i partecipanti ad un dato algoritmo e strutturarli in modo che tutti i partecipanti siano motivati ad agire come il progettista dell'algoritmo desidera. Tale area prende il nome di *progettazione di meccanismi algoritmici (algorithmic mechanism design)* [8].

La *Multi-Party Computation (MPC, computazione a parti multiple)* sicura può essere definita come il problema in cui un insieme di soggetti, ognuno avente un input segreto, desidera calcolare una certa funzione dei loro input in modo sicuro, dove per sicuro si intende, come minimo, che: (a) l'output della funzione deve essere corretto e (b) la segretezza degli input deve essere preservata anche in uno scenario in cui una parte dei giocatori è corrotta.
Il primo problema di Multi-Party Computation sicura è stato descritto da Andrew Yao nel 1982 nell'articolo *"Protocols for secure computations"* [9]; il particolare problema introdotto da Yao è ricordato con il nome di *"Yao's millionaires's problem" (Il problema dei milionari di Yao)*: due milionari desiderano sapere chi tra di loro è più ricco, ma senza rivelare all'altro l'esatto ammontare del proprio patrimonio. In questo contesto il patrimonio di ciascun giocatore costituisce l'input segreto e la funzione da computare è definita come segue:

$$f(x_1, x_2) = \begin{cases} 1, & \text{se } x_1 < x_2, \\ 0, & \text{altrimenti.} \end{cases}$$

Se il risultato fosse che il primo milionario è più ricco del secondo, questa dovrebbe essere l'unica informazione che il secondo apprende dopo aver eseguito il protocollo per il calcolo di f [10]. Tale problema, storicamente e teoricamente importante per la crittografia, assume un nuovo significato nel contesto dell'e-commerce: ad esempio, consideriamo un'asta digitale, in cui l'input di ogni partecipante corrisponde all'offerta fatta per aggiudicarsi un dato oggetto e l'output è del tipo $f(x_1, \ldots, x_n) = (x_j, j)$, dove $1 \leqslant j \leqslant n$ è l'indice per cui $x_j = max(x_1, \ldots, x_n)$ (in caso di parità si può scegliere, attraverso una funzione randomizzata, un giocatore a caso tra quelli con l'offerta più alta). Un altro contesto di rilevanza pratica in cui possiamo possiamo rivedere il problema MPC è quello degli schemi di voto elettronico [10]: ciascun partecipante alla votazione possiede un input segreto corrispondente all'identificativo del candidato che desidera votare; l'output della funzione sarà il candidato che ha ricevuto il più alto numero di voti e tale valore sarà l'unico a dover essere reso pubblico. Nel caso in cui ci fossero unicamente due candidati da votare, dunque, l'input

segreto di ogni giocatore sarà $x_i \in \{0, 1\}$ (dove 0 esprime una preferenza per il primo candidato, mentre 1 esprime una preferenza per il secondo) e la funzione da computare sarà del tipo

$$f(x_1, \ldots, x_n) = \begin{cases} 1, & \text{se } \sum_{i=1}^{n} x_i > \frac{n}{2}, \\ 0, & \text{altrimenti.} \end{cases}$$

L'obbiettivo di questo lavoro di tesi è illustrare il problema della *Multi-Party Computation (MPC)* e mostrare come sia possibile modellare i partecipanti ad un *protocollo MPC* in modo razionale sfruttando le nozioni della *Teoria dei Giochi*. La struttura della tesi è la seguente:

- *Capitolo 2: Introduzione alla Teoria dei Giochi.* In questo capitolo introduciamo le nozioni fondamentali della Teoria dei Giochi focalizzandoci sulla categoria dei *giochi "one-shot simultaneous moves"*. Nella prima sezione, dopo aver definito in modo informale i concetti base, definiamo formalmente i *giochi in forma normale* e la nozione di *strategia* in tali giochi. La trattazione prosegue enunciando alcuni *concetti di soluzione* quali l'*ottimo paretiano*, l'*equilibrio di Nash* e la *dominanza*. Il capitolo si conclude analizzando tramite esempi le seguenti classi di giochi: *giochi con payoff comune*, *giochi a somma costante*, *giochi con equilibri a strategie dominanti*.

- *Capitolo 3: Multi-Party Computation.* Il capitolo si pone l'obbiettivo di introdurre ed analizzare il *problema MPC*, illustrare le nozioni di sicurezza che un *protocollo MPC* dovrebbe rispettare e mostrare la struttura di tali *protocolli MPC sicuri*. Nella prima parte, introduciamo il *problema MPC* e caratterizziamo la figura dell'*avversario* all'interno della computazioni a parti multiple. Proseguendo nella trattazione, vengono illustrati i *requisiti di sicurezza* che un protocollo MPC dovrebbe garantire e formalizzato il concetto di protocollo MPC che computa una funzione in modo sicuro. Il capitolo si chiude, dopo una breve panoramica dei risultati più significativi ottenuti in letteratura per la MPC sicura, illustrando la struttura di un protocollo MPC che garantisce unicamente i requisiti di sicurezza minimi: *input privacy* e *correttezza della computazione*.

- *Capitolo 4: Multi-Party Computation modellata secondo la Teoria dei Giochi.* Il capitolo si occupa della costruzione di un modello di MPC con partecipanti razionali. Inizialmente, si definisce un *gioco di valutazione di funzione*, in cui l'obbiettivo dei partecipanti è stimare correttamente il valore di una funzione dei propri input, modellato sfruttando il concetto di *gioco ad informazione incompleta*. In seguito, utilizzando le nozioni di *equilibrio correlato* e di *gioco mediato ad informazione incompleta*, definiamo un *gioco mediato di valutazione di funzione*, in cui i partecipanti sfruttano un mediatore per

computare la funzione che desiderano calcolare. Infine, definiamo un *Gioco di Multi-Party Computation*, appartenente alla classe dei *giochi estesi tramite cheap-talk*, in cui a tutti gli effetti i partecipanti computano una funzione eseguendo un protocollo MPC. Nella trattazione ci occupiamo inoltre di caratterizzare la classe di funzioni, le *funzioni* K*-NCC*, per le quali, anche in coalizioni di al massimo K giocatori, questi sono incentivati a condividere i propri input segreti in modo onesto e corretto, e di definire un appropriato concetto di equilibrio resistente a tali coalizioni, *equilibrio* K*-resistente*.

2

INTRODUZIONE ALLA TEORIA DEI GIOCHI

2.1 CONCETTI DELLA TEORIA DEI GIOCHI

La *Teoria dei Giochi* è lo studio dei modelli matematici che descrivono il "processo decisionale" [11]: è quindi la scienza matematica che studia e analizza le interazioni strategiche tra più soggetti, spesso detti "giocatori", intelligenti, razionali e tra loro indipendenti, in situazioni di conflitto e/o cooperazione [4].

Per comprendere meglio di cosa si occupa la teoria dei giochi è bene introdurre alcuni concetti, per il momento in modo informale:

Gioco. Un *gioco*, o *gioco strategico*, è costituito da un insieme di partecipanti e rappresenta un modello di interazione tra quest'ultimi [12]. In sostanza, è una descrizione dell'interazione strategica che include vincoli sulle azioni che i partecipanti possono compiere e sugli "interessi" di quest'ultimi, ma non specifica quali azioni compieranno [3]. Ogni partecipante (giocatore) viene inizializzato con informazioni riguardo gli altri partecipanti, obbiettivi (risultati del gioco desiderati) e un insieme di possibili azioni da compiere. Di norma, un gioco può essere costituito da una singola mossa per ogni giocatore, oppure da vari turni in cui i giocatori agiscono in modo interattivo [13].

Giocatore. I *giocatori* sono i partecipanti al gioco, anche chiamati *agenti* o *decisori* (dall'inglese *"Decision-makers"*) ; interagiscono tra di loro e cercano di raggiungere i propri obbiettivi scegliendo l'azione più indicata dall'insieme di azioni possibili. Le decisioni prese da un giocatore possono influire sui risultati conseguibili dagli altri e viceversa, secondo un meccanismo di retroazione [14]. Si ipotizza che i partecipanti al gioco siano totalmente "egoisti", cioè che ognuno agisca unicamente per il proprio interesse; ciò significa che ogni giocatore non cambierà il proprio comportamento per aiutare o intralciare altri, ma agirà solamente per raggiungere il proprio obbiettivo [2]. Si suppone che i giocatori siano *intelligenti* e *razionali*: in senso stretto, per *intelligente* intendiamo che, dati due risultati, un giocatore sarà in grado di decidere quale dei due sia migliore per lui o se siano equivalenti, mentre per *razionale* si intende semplicemente la capacità logica di saper riconoscere le azioni necessarie per massimizzare i propri "guadagni" [12].

Azioni. Le *azioni* rappresentano sostanzialmente le "mosse" a disposizione del giocatore. L'insieme delle possibili azioni definisce in modo informale le *regole del gioco*, cioè cosa i giocatori possono fare (e di conseguenza anche cosa i giocatori non possono fare, cioè tutto il resto) . Si suppone che i partecipanti al gioco siano a conoscenza delle regole del gioco e consapevoli delle conseguenze di ogni singola mossa [13].

Strategia. La *strategia* in un gioco è la sequenza delle decisioni, o azioni, compiute da un giocatore: è quindi un piano completo di azione per qualsiasi situazione in cui il giocatore potrebbe trovarsi, il che significa che andrà a determinare l'azione o le azioni che quest'ultimo compirà in ogni stadio del gioco e per ogni possibile "storia" del gioco. Dunque la nozione di strategia della teoria dei giochi si può equiparare a quella di algoritmo dell'informatica [13].

Profilo e Risultato. Avendo n giocatori, se ogni giocatore sceglie un'azione dal proprio insieme di azioni possibili, avremo un *profilo* per n giocatori.
Un *profilo* è quindi una n-upla di azioni ed ad esso corrisponde un *risultato*, ossia delle conseguenze che si vengono a produrre nel momento in cui ciascun giocatore compie una determinata azione [12].
I concetti di profilo e risultato sono estendibile anche alle strategie.

Funzione Utilità e Payoff. Dato l'insieme dei possibili *risultati*, ogni giocatore è in grado di ordinare tale insieme secondo un personale valore di preferenza (*giocatori intelligenti*) . Per esprimere la relazione di preferenza di ciascun giocatore sull'insieme dei possibili risultati, utilizziamo una *funzione utilità*, associata ad ognuno di essi, che fa corrispondere valori più elevati a risultati più graditi. Quindi, dato un profilo ed un giocatore, la funzione utilità ci dirà quant'è l'"utilità", in inglese *"payoff"* (letteralmente "ricompensa") , derivante dal dato profilo per il dato giocatore; tale valore potrà essere accompagnato da un'unità di misura e potrà essere positivo, negativo o nullo.
Il valore di *payoff* dipende direttamente dal singolo giocatore e dal vettore di azioni, ovvero un profilo, di tutti i giocatori; questo è un dato importante poiché distingue un gioco da un semplice problema decisionale: non è infatti possibile trascurare l'interazione con gli altri giocatori [12]. I vari giocatori interagiranno tra di loro, attraverso la scelta di una strategia, con l'obbiettivo di massimizzare il proprio *payoff* [13].

Soluzione. La *soluzione* di un gioco è una descrizione sistematica dei possibili risultati che possono emergere in un determinato tipo di gioco [3].

Equilibrio. Un *equilibrio* in un gioco è rappresentato da un insieme di strategie per i partecipanti al gioco, tale che nessun giocatore, o insieme di giocatori, sia incentivato a cambiare la propria strategia, a meno che qualcun altro non la

cambi [13].

La Teoria dei Giochi si pone l'obbiettivo di studiare le connessioni tra lo stato iniziale del gioco ed i possibili *risultati*: quali passi e decisioni debbano essere prese in una data situazione, quali altri risultati siano possibili o desiderabili, come quest'ultimi siano legati alle "regole" dell'interazione e come sia possibile modificare e costruire "regole" per ottenere un risultato desiderato [13]. Quindi, in primis, la teoria dei giochi svolge un ruolo *predittivo*: analizza il processo decisionale generato dall'interazione cooperativa, e non, di più giocatori, interpretando così un'approssimazione della realtà con il fine di spiegare il perché delle scelte dei partecipanti [12]. In secondo luogo svolge due funzioni più *prescrittive*: suggerire *soluzioni* per classi di giochi esaminandone le proprietà [3] (nello specifico cercare di determinare quali situazioni di equilibrio possono, o non possono, verificarsi come risultato dell'interazione di due o più soggetti [12]) e progettare giochi al fine di condurre i giocatori, "egoisti" e razionali, verso interazioni "benefiche". Quest'ultimo aspetto della teoria dei giochi prende il nome di *"progettazione di meccanismi"*, dall'inglese *"mechanism design"*; in tal contesto il gioco progettato ad hoc (*"designed game*") prenderà il nome di *meccanismo* [13].

2.2 GIOCO STRATEGICO

2.2.1 *Definizione di gioco strategico*

Iniziamo definendo giochi strategici che verranno giocati un'unica volta (*"one-shot"*) ed in cui tutti i giocatori selezioneranno simultaneamente un'azione dal proprio insieme di strategie, una e una sola volta, prima dell'inizio del gioco (*"simultaneous move"*) . Dunque i giochi che appartengono a questa categoria non sono basati su turni e non sono nemmeno sequenziali, e in inglese vengono chiamati *"one-shot simultaneous move games"* [15].

I *giochi strategici* sono anche detti *giochi in forma matriciale* e nella terminologia introdotta da Von Neumann e Morgenstern in [5] *giochi in forma normale* [3]. La *forma normale* è la più comune rappresentazione di un'interazione strategica nella teoria dei giochi e, dato che la maggior parte delle altre rappresentazioni possono essere ridotte a essa, possiamo senza dubbio dire che quest'ultima è la più fondamentale [2].

Definizione 2.1 (*Gioco in forma normale*)**.** Un gioco in forma normale, NFG, è costituito da una tupla $\mathrm{NFG} = (N, A, u)$ dove:

- N è un insieme finito di $n \in \mathbb{N}$ *giocatori* indicizzati con i;

		Giocatore$_2$	
		a_{21}	a_{22}
Giocatore$_1$	a_{11}	$u_2(a_{11}, a_{21}), u_1(a_{11}, a_{21})$	$u_2(a_{11}, a_{21}), u_1(a_{11}, a_{21})$
	a_{12}	$u_2(a_{11}, a_{21}), u_1(a_{11}, a_{21})$	$u_2(a_{11}, a_{21}), u_1(a_{11}, a_{21})$

Tabella 1: *Matrice dei payoff* in forma generalizzata ($\{a_{11}, a_{12}\} = A_1$ e $\{a_{21}, a_{22}\} = A_2$ azioni disponibile, rispettivamente, per il giocatore$_1$ e per il giocatore$_2$)

- $A = A_1 \times \ldots \times A_n$, prodotto cartesiano, è detto insieme degli stati del gioco; A_i è l'insieme di tutte le possibili *azioni* del giocatore i-esimo. Ogni vettore $a = (a_1, \ldots, a_n)$ rappresenta un *profilo di azioni*, con $a_i \in A_i$ un'azione per il giocatore i-esimo;
- $u = (u_1, \ldots, u_n)$, con $u_i \colon A \to \mathbb{R}$ *funzione utilità* per il giocatore i-esimo [2];

Il *risultato* di un gioco NFG è dato dal vettore $a = (a_1, \ldots, a_n)$ di azioni, cioè il profilo di azioni, effettivamente scelte dai giocatori. Dato il risultato di un gioco i valori di *payoff* dei giocatori sono calcolati utilizzando la funzione u_i [13]. È inoltre importante notare che ogni giocatore conoscerà le proprie azioni disponibili, quelle disponibili agli altri giocatori ed i valori di payoff di tutti i giocatori per ogni possibile risultato: i giochi di questo tipo sono detti *giochi a informazione completa* [16].

Un modo naturale per rappresentare i giochi è tramite una matrice n-dimensionale, spesso detta *matrice dei payoff* o *matrice dei costi*, dove $n = |N|$ è il numero di giocatori. Nel caso di due giocatori, per rappresentare un dato NFG, avremo una matrice bidimensionale M. Ogni riga di M corrisponderà ad una possibile azione per il giocatore$_1$, ogni colonna corrisponderà ad una possibile azione per il giocatore$_2$ ed ogni elemento della matrice sarà associato ad uno dei possibili risultati (quindi ad un dato profilo di azioni: (a_i, a_j) con azione $a_i \in A_1$ vettore di azioni del giocatore$_1$ e azione $a_j \in A_2$ vettore di azioni del giocatore$_2$) . Per la precisione, l'elemento $m_{ij} \in M$ sarà costituito dai valori di payoff, del primo e del secondo giocatore, associati al dato risultato ($m_{ij} = (u_1(a_i, a_j), u_2(a_i, a_j))$) [2].

La tabella 1 ci mostra una rappresentazione generalizzata della matrice dei payoff. È conveniente utilizzare questo tipo di rappresentazione solamente nel caso di due giocatori e solo se quest'ultimi hanno poche azioni selezionabili. Ad esempio, se consideriamo un gioco nel quale abbiamo n giocatori con due possibili azioni l'uno, allora, per rappresentarlo nella sua forma matriciale, avremo bisogno di assegnare un valore ad ognuno dei possibili 2^n vettori di azioni. Rappresentare esplicitamente il gioco diventa complicato e risulta più semplice

		Sospettato$_2$	
		Confessa	Non confessa
Sospettato$_1$	Confessa	-3, -3	0, -4
	Non confessa	-4, 0	-1, -1

Tabella 2: *Dilemma del prigioniero* (Esempio 2.1)

darne una rappresentazione puramente descrittiva [15].

Esempio 2.1 (*Dilemma del prigioniero*)**.** Due sospettati per un crimine, dopo essere stati portati in celle separate, in modo da evitare ogni forma di comunicazione reciproca, vengono interrogati dagli investigatori. I due sospettati vengono posti davanti ad una scelta: confessare il crimine oppure non confessare. Viene anche loro spiegato che se solo uno dei due confessa, chi ha confessato eviterà la pena per aver collaborato e sarà utilizzato come testimone durante il processo contro l'altro, che riceverà un pena detentiva pari a quattro anni. Se nessuno dei due confessa, entrambi verranno condannati ad un anno per i reati minori evidenziati dalle indagini svolte. Infine, se entrambi confessano, riceveranno una condanna ridotta di tre anni per la collaborazione. Il gioco è rappresentato tramite la tabella 2, i valori di payoff sono tutti non positivi e il loro valore assoluto può essere interpretato come il numero di anni che ciascuno sospettato, una volta concluso il gioco e quindi stabilita la pena, trascorrerà in prigione [3]. L'obbiettivo dei sospettati dovrebbe essere quello di cercare di minimizzare la propria condanna. Il risultato migliore per entrambi è ottenuto dallo scenario in cui nessuno dei due confessa (la strategia "non confessa", come vedremo meglio in seguito, sarà detta *strategia dominante*), tuttavia, essendo il gioco *non cooperativo*, cioè un gioco in cui ogni giocatore non è a conoscenza delle scelte fatte dall'altro, la miglior strategia di questo gioco è "confessa": anche supponendo che i due si siano promessi di non confessare in caso di arresto, entrambi non potranno avere la certezza che la promessa venga mantenuta dal compagno. Ogni sospettato confessando potrebbe essere rilasciato oppure fare tre anni di galera (confessando: 0,-3) , non confessando rischia di farne uno oppure quattro (non confessando: -1, -4) : quindi, in ogni caso, per i sospettati è più conveniente confessare. La situazione di *equilibrio* è data dal profilo $(\text{confessa}, \text{confessa})$ [17].

2.2.2 *Strategie nei giochi in forma normale*

Per il momento abbiamo analizzato formalmente le *azioni* disponibili per ogni ogni giocatore, ma non abbia ancora definito l'insieme di *strategie* e le possibile scelte disponibili. Come detto precedentemente, una strategia sostanzialmente è un algoritmo che specifica, per ogni scenario che potrebbe verificarsi durante il

gioco, quale azione il giocatore dovrà selezionare: in pratica, è un algoritmo per giocare il gioco.
In generale, ogni giocatore avrà un insieme di strategie disponibili. L'esecuzione di un gioco sarà caratterizzata dalla scelta di una singola strategia per ogni partecipante, cioè risulterà in un determinato *profilo di strategie*, che dunque includerà una ed una sola strategia per ogni giocatore [18].

Una possibile strategia potrebbe essere costituita semplicemente dalla scelta di una specifica mossa o azione che il giocatore seguirà in ogni possibile situazione del gioco [19]: questo tipo di strategia viene chiamata *strategia pura* e per rappresentarla utilizzeremo la stessa notazione precedentemente utilizzata per le azioni. Nel caso in cui ogni giocatore opti di giocare scegliendo una strategia pura avremo come risultato del gioco un *profilo a strategia pura*.
Un'altra possibile tipologia di strategia è la seguente: ogni giocatore sceglierà un'azione dal proprio insieme di azioni disponibili in modo casuale e in accordo con una data distribuzione di probabilità. Una strategia di questo tipo è detta *strategia mista* e in sostanza ci dirà la frequenza con cui il giocatore gioca le varie azioni [2]. La scelta di una strategia del genere potrebbe essere dettata dalla necessità di evitare che gli avversari traggano benefici strategici dalla conoscenza della mossa successiva; in alternativa una strategia mista può essere adottata semplicemente quando la scelta di azioni distinte porta ad esiti indifferenti [20].

Andiamo adesso a definire in modo formale il concetto di strategia mista per i giochi in forma normale: tale definizione ci porterà a capire in modo più preciso anche il concetto di strategia pura.

Definizione 2.2 (*Strategia mista*)**.** Sia $NFG = (N, A, u)$ un gioco in forma normale e, per ogni insieme X, sia $\Delta(X)$ l'insieme di tutte le distribuzione di probabilità su X, allora l'insieme di strategie miste per il giocatore$_i$ sarà $S_i = \Delta(A_i)$.
La k-esima *strategia mista* per il giocatore$_i$ sarà $s_k \in S_i$, mentre con $s_k(a_j)$ denotiamo la probabilità che l'azione $a_j \in A_i$ venga giocata nella particolare strategia mista s_k.

Per comodità possiamo rappresentare una strategia mista $s_k \in S_i$ tramite un vettore di probabilità $s_k(A_i) = \{s_k(a_1), \ldots, s_k(a_m)\}$ sulle possibili m azioni che il giocatore$_i$ può intraprendere [12].

Definizione 2.3 (*Profilo a strategia mista*)**.** L'insieme di *profili a strategia mista*, per un gioco con $n = |N|$ giocatori, è il prodotto cartesiano dei singoli insiemi di strategie miste di ogni giocatore $S = S_1 \times \ldots \times S_n$. Ogni vettore $s = (s_1, \ldots, s_n)$ rappresenta un *profilo a strategia mista*, con $s_i \in S_i$ strategia mista per il giocatore i-esimo.

Definizione 2.4 (*Supporto*)**.** Il supporto per una data strategia mista $s_k \in S_i$, relativa all'i-esimo giocatore, è l'insieme di azioni $\{a_j | s_k(a_j) > 0\} = supp(s_k)$ (con $a_j \in A_i$ insieme di azioni per il giocatore$_i$).

Quindi un supporto è il sottoinsieme di azioni alle quali viene assegnata probabilità positiva dalla strategia mista s_k, cioè quelle azioni effettivamente giocabili dopo aver scelto la data strategia mista.

Si noti che sostanzialmente una strategia pura è un caso speciale di strategia mista in cui il supporto è costituito da una singola azione. Proviamo a definirlo in modo formale:

Definizione 2.5 (*Strategia pura*)**.** Sia $s_k \in S_i$ una strategia mista per il giocatore$_i$, se $|\{a_j | s_k(a_j) > 0\}| = 1$, allora s_k è una *strategia pura*. In tal caso possiamo identificare la strategia pura semplicemente con l'unica azione ($a_j \in supp(s_k)$) appartenente al suo supporto: $s_k = a_j$.

Nel caso in cui $s_k \in S_i$ sia una strategia pura avremo un'unica azione $a_j \in A_i$ tale che $s_k(a_j) = 1$; quindi la probabilità che il giocatore$_i$ scelga l'azione a_j equivale all'evento certo. Per comodità spesso ci riferiremo ad un'azione chiamandola semplicemente strategia pura [3].

Possiamo anche avere strategie miste in cui assegniamo un valore di probabilità diverso da zero ad ogni azione disponibile per il dato giocatore; in tal caso, diremo che abbiamo un *supporto totale* e una *strategia totalmente mista*.

Definizione 2.6 (*Strategia totalmente mista*)**.** Sia $s_k \in S_i$ una strategia mista per il giocatore$_i$, se $\{a_j | s_k(a_j) > 0\} = A_i$, allora s_k è una *strategia totalmente mista* ed il supporto per tale strategia sarà detto *supporto totale*.

Nel caso in cui $s_k \in S_i$ sia una strategia totalmente mista avremo che $s_k(a_j) = \frac{1}{m}$, $\forall a_j \in A_i$ con $|A_i| = m$.

Dato un particolare profilo a strategia mista, è necessario stabilire quali siano i valori di payoff dei giocatori; infatti, la matrice dei payoff li stabilisce direttamente solo nel caso particolare di profili a strategia pura. Definiamo a tale scopo la *funzione utilità attesa* per il giocatore$_i$: essa calcola la probabilità di raggiungere ogni risultato dato il profilo strategico e poi calcola la media dei payoff dei risultati, pesata con le probabilità di ogni risultato. (Il simbolo u_i è sovraccaricato: verrà utilizzato sia per la funzione utilità sia per la funzione utilità attesa)

Definizione 2.7 (*Funzione utilità attesa*)**.** Dato un gioco in forma normale $\mathsf{NFG} = (N, A, u)$ ed un profilo a strategia mista $s = (s_1, \ldots, s_n)$, la *funzione utilità attesa* per il giocatore$_i$ sarà

$$u_i(s) = \sum_{a \in A} u_i(a) \cdot \prod_{j=1}^{n} s_j(a_j) \tag{2.1}$$

con $a \in A$ profilo di azioni [2].

Teorema 2.1 (Utilità attesa per un'azione, in un gioco con due giocatori)**.** *Dato un gioco in forma normale* $\mathsf{NFG} = (N, A, u)$ *con due giocatori,* G *ed* F*, sia* $A_f = (a_1^f, \ldots, a_m^f)$ *l'insieme di azioni possibili per* F*. Inoltre, sia* $s_f(a_j^f) = p_j$ *la probabilità che il giocatore* G *assegna al fatto che il giocatore* F *scelga l'azione* a_j *(dunque* $s_f = (s_f(a_1^f) = p_1, \ldots, s_f(a_m^f) = p_m)$*), allora, se* G *sceglie di giocare l'azione* a_k^g*, la sua utilità attesa sarà*

$$U_g(a_k^g) = u_g(a_k^g, a_1^f) \cdot p_1 + u_g(a_k^g, a_2^f) \cdot p_2 + \cdots + u_g(a_k^g, a_m^f) \cdot p_m. \tag{2.2}$$

Dimostrazione. L'equazione (2.2) si ricava direttamente da (2.1) [12]. Il profilo a strategie miste per due giocatori G ed F sarà $s = (s_g, s_f)$, considerando le assunzioni di probabilità fatte da G e sapendo che quest'ultimo giocherà sicuramente l'azione a_k^g sappiamo che $s_g = \{s_g(a_j) | s_g(a_j^g) = 0 \; \forall_{j \neq k}, s_g(a_{j=k}^g) = 1\}$ e $s_f = (s_f(a_1^f) = p_1, \ldots, s_f(a_m^f) = p_m)$. Applichiamo la funzione utilità attesa per G ad s

$$u_g(s) = \sum_{a \in A} u_g(a) \cdot s_g(a_1) \cdot s_f(a_2)$$

con $a = (a_1, a_2)$ profilo di azioni, $a_1 \in A_g$ e $a_2 \in A_f$. La probabilità che il risultato del gioco sia un dato profilo di azioni a è data da $s_g(a_1)s_f(a_2)$; dunque, sapendo che $s_g(a_j^g) = 0 \; \forall_{j \neq k}$, i profili di azioni con probabilità diversa da zero saranno solamente quelli in cui $a_1 = a_g^k$, ergo

$$\begin{aligned} u_g(s) &= u_g(a_k^g, a_1^f) \cdot s_g(a_k^g) \cdot s_f(a_1^f) + \cdots + u_g(a_k^g, a_m^f) \cdot s_g(a_k^g) \cdot s_f(a_m^f) \\ &= u_g(a_k^g, a_1^f) \cdot s_f(a_1^f) + \cdots + u_g(a_k^g, a_m^f) \cdot s_f(a_m^f) \\ &= u_g(a_k^g, a_1^f) \cdot p_1 + \cdots + u_g(a_k^g, a_m^f) \cdot p_m \\ &= U_g(a_k^g). \end{aligned}$$

□

2.3 ALCUNI CONCETTI DI SOLUZIONE

I concetti utilizzati per analizzare i giochi, nell'ambito della teoria dei giochi, vengono detti *"concetti di soluzione"* [2].
Come detto precedentemente, una *soluzione* è una descrizione sistematica dei possibili *risultati* che possono emergere in una data famiglia di giochi. La teoria dei giochi pone tra i suoi obbiettivi quello di proporre *soluzioni* ragionevoli per classi di giochi e di esaminare le loro proprietà [3], quindi cerca di identificare particolari sottoinsiemi di *risultati*, detti *concetti di soluzione*, che sono interessanti per un verso o per un altro. Tali concetti di soluzione intendono descrivere quelle *strategie* che i giocatori, individualmente o congiuntamente, dovrebbero seguire come conseguenza delle ipotesi di razionalità [12] : sostanzialmente, possiamo pensare ad un *concetto di soluzione* come ad una regola formale per predire come il gioco verrà giocato [21].

2.3.1 *Ottimo paretiano*

In un problema decisionale con un singolo agente, una nozione chiave è quella di *strategia ottimale*: cioè una strategia che massimizza l'utilità attesa per una data situazione nella quale l'agente opera. In una situazione con più giocatori, il concetto di strategia ottimale perde di significato, in quanto la questione è molto più complessa: come facciamo a stabilire che un dato *risultato* di un gioco sia migliore di un altro?
Non possiamo decretare che gli interessi di un giocatore siano più importanti di quelli di un altro, ma, nonostante ciò, dato un gioco con due giocatori e dati due risultati s e s' tali che $u(s) = (u_1(s), u_2(s)) = (9, 3)$ e $u(s') = (u_1(s'), u_2(s')) = (10, 3)$, in modo intuitivo, possiamo dire con certezza che il risultato s sia migliore del risultato s'. Esprimiamo tale concetto in modo formale:

Definizione 2.8 (*Dominanza paretiana*)**.** Un profilo strategico s *domina in modo paretiano* un profilo strategico s' se, $\forall i \in N$, $u_i(s) \geqslant u_i(s')$ ed $\exists$ almeno un $j \in N$ per cui $u_j(s) > u_j(s')$.

In sostanza, se abbiamo profilo strategico dominato in modo paretiano, sappiamo che qualche giocatore potrebbe ottenere dei payoff migliori senza che nessuno degli altri giocatori ottenga dei payoff peggiori. La dominanza paretiana ci dà un ordinamento parziale su un insieme di profili strategici, ma non ci permette ancora di identificare un singolo risultato da etichettare come "il migliore".

Definizione 2.9 (*Ottimo paretiano*)**.** Un profilo strategico s è *pareto ottimale* se non esiste nessun altro profilo strategico $s' \in S$ che domina in modo paretiano s.

Ogni gioco avrà almeno un profilo strategico ottimo paretiano. Inoltre, deve sempre esistere almeno un profilo strategico ottimo paretiano in cui tutti i giocatori adottano strategie pure [2].

2.3.2 *Equilibrio di Nash*

Cerchiamo adesso di analizzare un gioco dal punto di vista di un singolo agente. Un giocatore che fosse a conoscenza delle strategie adottate dagli altri si troverebbe difronte ad un problema decisionale relativamente semplice: scegliere la strategia che massimizza il proprio payoff. Indichiamo con $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ un profilo strategico s privo della strategia s_i relativa all' $i-$esimo giocatore, dunque $s = (s_i, s_{-i})$.

Definizione 2.10 (*Miglior risposta*)**.** Se il giocatore$_i$ è a conoscenza del fatto che gli altri giocatori adottano un profilo strategico s_{-i}, la *miglio risposta* del giocatore$_i$ al profilo s_{-i} è una strategia mista $s_i^* \in S_i$ tale che $u_i(s_i^*, s_{-i}) \geqslant u_i(s_i, s_{-i})$ $\forall s_i \in S_i$.

La miglior risposta sarà unica solamente nel caso in cui sia costituita da una strategia pura [2].

In generale, un giocatore non conoscerà quali strategie saranno adottate dagli altri partecipanti, dunque non possiamo considerare la nozione di miglior risposta come un concetto di soluzione. Possiamo comunque sfruttare l'idea di miglior risposta per definire una delle principali nozioni della teoria dei giochi non cooperativi, l'*equilibrio di Nash*.

Definizione 2.11 (*Equilibrio di Nash*)**.** Un profilo strategico $s = (s_1, \ldots, s_n)$ è un *equilibrio di Nash* se, $\forall$ giocatore$_i$, s_i è la migliore risposta al profilo s_{-i}.

In modo intuitivo possiamo dire che un equilibrio di Nash è un profilo strategico stabile: nessun giocatore cambierebbe razionalmente la propria strategia anche se fosse a conoscenza delle scelte strategiche seguite dagli altri partecipanti [2].

Definizione 2.12 (*Equilibrio di Nash a strategie pure*)**.** Sia $s = (s_1, \ldots, s_n)$ un *equilibrio di Nash*. Se, $\forall$ giocatore$_i$, s_i è una *strategia pura*, dunque $\text{supp}(s_i) = a_i$, allora diremo che s è un *equilibrio di Nash a strategie pure*.

Definizione 2.13 (*Equilibrio di Nash a strategie miste*)**.** Sia $s = (s_1, \ldots, s_n)$ un *equilibrio di Nash*. Se, $\forall$ giocatore$_i$, s_i è una *strategia mista*, allora diremo che s è un *equilibrio di Nash a strategie miste* .

Possiamo dividere l'equilibrio di Nash in due categorie:

Definizione 2.14 (*Equilibrio di Nash forte*)**.** Un profilo strategico $s = (s_1, \ldots, s_n)$ è un *equilibrio di Nash forte* se, $\forall$ giocatore$_i$ e $\forall$ strategia $s_i' \neq s_i$, $u_i(s_i, s_{-i}) > u_i(s_i', s_{-i})$.

Un equilibrio di Nash si dice forte se la strategia selezionata da ogni giocatore costituisce l'unica miglior risposta alle strategie scelte dagli altri giocatori, in caso contrario diremo che l'equilibrio è debole.

Definizione 2.15 (*Equilibrio di Nash debole*)**.** Un profilo strategico $s = (s_1, \ldots, s_n)$ è un *equilibrio di Nash debole* se, $\forall$ giocatore$_i$ e $\forall$ strategia $s_i' \neq s_i$, $u_i(s_i, s_{-i}) \geqslant u_i(s_i', s_{-i})$ e s non è un equilibrio di Nash forte.

In una situazione di equilibrio di Nash debole, almeno un giocatore$_i$ possiede una miglior risposta alle strategie scelte dagli altri giocatori che non è la sua strategie di equilibrio ($s_i \in s$ equilibrio di Nash); tale equilibrio è meno stabile rispetto ad un equilibrio di Nash forte.
Equilibri di Nash a strategie miste sono sempre deboli, mentre ciò non è detto per equilibri di Nash a strategie pure [2].

Teorema 2.2 (Nash, 1951)**.** *Ogni gioco con un numero finito di giocatori e di profili di azioni ha almeno un equilibrio di Nash a strategie miste.*

Date due strategie qualunque nel supporto dell'equilibrio di Nash, l'utilità attesa per tali strategie sarà la stessa. Tale utilità attesa sarà maggiore o uguale di quella ottenuta giocando una strategia esterna al supporto.

Proposizione 2.2.1. *Un profilo strategico* $s = (s_1, \ldots, s_n)$ *è un equilibrio di Nash a strategie miste per un gioco in forma normale* $NFG = (N, A, u)$ *se e solo se:*

1. $u_i(a_i, s_{-i}) = u_i(a_i', s_{-i}) \qquad \forall a_i, a_i' \in supp(s_i)$
2. $u_i(a_i, s_{-i}) \geqslant u_i(a_i', s_{-i}) \qquad \forall a_i \in supp(s_i) \qquad \forall a_i' \notin supp(s_i)$

Dimostrazione. $\Rightarrow$ supponiamo che $s = (s_1, \ldots, s_n)$ sia un equilibrio di Nash (NE) e che (1) e (2) siano false. Se (1) è falsa, allora esisterà una coppia di azioni a_i, a_i' appartenenti a $supp(s_i)$ tali che $u_i(a_i', s_{-i}) > u_i(a_i, s_{-i})$, ma allora, per la definizione di NE, s non può essere un NE. Se (2) è falsa, allora esisterà una coppia di azioni $a_i \in supp(s_i)$ e $a_i' \notin supp(s_i)$ tali che $u_i(a_i', s_{-i}) > u_i(a_i, s_{-i})$: se ogni volta in cui dovremmo giocare a_i giochiamo invece a_i', otteniamo dei payoff strettamente migliori; dunque s, per definizione di NE, non può essere un NE.
$\Leftarrow$ supponiamo che (1) e (2) siano vere e che s non sia un NE, allora $\exists i \in N$ tale che $u_i(s_i', s_{-i}) > u_i(s_i, s_{-i})$. In particolare $\exists a_i' \in supp(s_i')$ tale che $u_i(a_i', s_{-i}') > u_i(s_i, s_{-i})$. Se (1) è vera allora $a_i' \notin supp(s_i)$ deve essere vera, ma ciò contraddice (2). □

Corollario 2.2.1. *Sia* $s = (s_1, \ldots, s_n)$ *un equilibrio di Nash. Dato* $supp(s_i)$ $\forall i \in N$*, allora è possibile calcolare tale equilibrio in tempo polinomiale [13].*

2.3.3 *Dominanza*

Con la definizione di dominanza paretiana abbiamo visto cosa intendiamo quando diciamo che un profilo strategico domina (in modo paretiano) un altro profilo strategico; spostiamo adesso il concetto di dominanza alle singole strategie.
Una strategia ne domina un'altra per un giocatore i se la prima comporta un payoff maggiore rispetto alla seconda, per ogni possibile profilo strategico adottato dagli altri giocatori.

Definizione 2.16 (*Dominanza*)**.** Siano s_i e s'_i due strategie per il giocatore$_i$ e sia S_{-i} l'insieme di tutte le strategie possibili per i restanti giocatori, allora

1. s_i *domina strettamente* s'_i se $u_i(s_i, s_{-i}) > u_i(s'_i, s_{-i})$, $\forall s_{-i} \in S_{-i}$.
2. s_i *domina debolmente* s'_i se $u_i(s_i, s_{-i}) \geqslant u_i(s'_i, s_{-i})$, $\forall s_{-i} \in S_{-i}$, e $u_i(s_i, s_{-i}) > u_i(s'_i, s_{-i})$, per almeno un $s_{-i} \in S_{-i}$.
3. s_i *domina molto debolmente* s'_i se $u_i(s_i, s_{-i}) \geqslant u_i(s'_i, s_{-i})$, $\forall s_{-i} \in S_{-i}$.

Definizione 2.17 (*Strategia dominante*)**.** Una strategia è (strettamente, debolmente, molto debolmente) *dominante* per un dato giocatore, se domina (strettamente, debolmente, molto debolmente) tutte le altre strategie di quel giocatore.

Definizione 2.18 (*Strategia dominata*)**.** Una strategia s_i è (strettamente, debolmente, molto debolmente) *dominata* per un dato giocatore$_i$, se qualche altra strategia s'_i di quel giocatore la domina.

Definizione 2.19 (*Equilibrio a strategie dominanti*)**.** Un profilo strategico $s = (s_1, \ldots, s_n)$ nel quale ogni strategia s_i è dominante per l'i-esimo giocatore sarà un equilibrio di Nash e sarà chiamato un *equilibrio a strategie dominanti*.

Un equilibrio a strategie strettamente dominante sarà l'unico equilibrio di Nash possibile [2] e in tal caso la miglior mossa di un giocatore non dipenderà dalle scelte compiute dagli altri partecipanti [22].

Le strategie dominanti svolgono un ruolo molto importante nella teoria dei giochi, in particolare nel contesto del *design di meccanismi*: è spesso preferibile progettare giochi nei quali la "miglior strategia" (per coloro che progettano il gioco) sia quella dominante [13].

		Amico$_2$	
		"Star wars"	"Star trek"
Amico$_1$	"Star wars"	2, 2	0, 0
	"Star trek"	0, 0	1, 1

Tabella 3: *Gioco di coordinazione puro* (Esempio 2.2)

2.4 ESEMPI DI GIOCHI STRATEGICI

Analizziamo adesso alcune famiglie di giochi in forma normale.

2.4.1 *Giochi con payoff comune*

Definizione 2.20 (*Gioco con payoff comune*)**.** Un *gioco con payoff comune* è un gioco nel quale, $\forall a \in A_1 \times \ldots \times A_n$, profilo di azioni, e per ogni coppia di giocatori (i, j), $u_i(a) = u_j(a)$.

Giochi di questo tipo sono anche detti *giochi di coordinazione pura*: infatti i partecipanti non sono apertamente in conflitto, il loro obbiettivo è invece quello di coordinare le proprie scelte strategiche al fine di conseguire l'ottimale payoff comune.

Esempio 2.2 (*"Star wars" o "Star trek"*)**.** Due amici desiderano uscire insieme e vorrebbero andare al cinema per vedere un film di fantascienza. Al cinema più vicino, in una sala proiettano "Star wars" e nell'altra "Star trek"; entrambi gli amici preferirebbero assistere al primo. Associamo alle preferenze un valore tramite la funzione utilità e rappresentiamo il gioco nella tabella 3.
Proviamo ad analizzare il gioco: se la scelta strategica dell'amico$_2$ è $s_{-1} =$ ("Star trek"), la *miglior risposta* dell'amico$_1$ al profilo strategico s_{-1} è unica (infatti siamo nel caso di profili a strategie pure) ed è $s_1 =$ ("Start trek"); se la scelta strategica dell'amico$_1$ è $s_{-2} =$ ("Star trek"), la *miglior risposta* dell'amico$_2$ al profilo strategico s_{-2} è $s_2 =$ ("Start trek") $\Rightarrow$ ("Star trek", "Star trek") è un equilibrio di Nash. Con un ragionamento analogo giungiamo alla conclusione che anche ("Star wars", Star wars) è un equilibrio di Nash. Quindi, il gioco possiede due *equilibri di Nash a strategie pure*, ma i due giocatori hanno un interesse reciproco a raggiungerne uno in particolare, cioè ("Star wars", Star wars) [13].

Normalmente i giochi tendono ad includere contemporaneamente elementi di coordinamento e di competizione.

Esempio 2.3 (*"Baustelle" o "Verdena" (Battaglia dei sessi)*)**.** Francesco (f) e Giulia (g) desiderano trascorrere la serata insieme andando a vedere un concerto. Francesco preferirebbe andare a vedere il concerto dei "Baustelle" (B), mentre Giulia quello

		Giulia	
		"Baustelle"	"Verdena"
Francesco	"Baustelle"	2, 1	0, 0
	"Verdena"	0, 0	1, 2

Tabella 4: *Battaglia dei sessi* (Esempio 2.3)

dei "Verdena" (V). Ambedue, comunque, preferiscono uscire insieme piuttosto che rimanere separati: la mancanza del compagno renderebbe non interessante anche lo spettacolo più gradito. Assegnando alle preferenze un valore tramite una funzione utilità, possiamo rappresentare il gioco con la tabella 4.
In modo abbastanza ovvio si vede che esistono due *equilibri di Nash a strategie pure*: ("Baustelle","Baustelle") e ("Verdena","Verdena"). Esiste inoltre un *equilibrio di Nash a strategie miste*, tuttavia calcolarlo non è un compito semplice, a meno che non si conosca (o non si possa supporre) il *supporto* delle strategie di equilibrio. Supponiamo che Francesco scelga B con probabilità uguale a p e V con probabilità uguale ad $(1-p)$, dunque $s_f = (s_f(B) = p, s_f(V) = (1-p))$. Quindi, affinché ci si trovi in una situazione di equilibrio, data la strategia scelta da Francesco, Giulia dovrà trovarsi in una situazione di indifferenza rispetto alla scelta di una qualunque delle sue azione 2.2.1: l'utilità attesa per Giulia dovrà essere la stessa per ogni azione lei decida di giocare. Fatta questa assunzione, sfruttando l'equazione (2.2) possiamo ricavare i valori p e $(1-p)$ che Francesco dovrà adottare affinché ci si possa trovare in una situazione di equilibrio:

$$\begin{aligned} U_g(B) &= U_g(V) \\ u_g(B,B)\cdot p + u_g(V,B)\cdot(1-p) &= u_g(B,V)\cdot p + u_g(V,V)\cdot(1-p) \\ 1\cdot p + 0\cdot(1-p) &= 0\cdot p + 2\cdot(1-p) \\ p &= \frac{2}{3}. \end{aligned}$$

Dunque $s_f = (s_f(B) = \frac{2}{3}, s_f(V) = \frac{1}{3})$ e con un ragionamento analogo possiamo concludere che $s_g = (s_g(B) = \frac{1}{3}, s_g(V) = \frac{2}{3})$. Quindi la situazione di equilibrio di Nash a strategie miste è data dal profilo strategico $s = (s_f, s_g)$ e l'utilità attesa (2.1) per entrambi i giocatori sarà $\frac{2}{3}$; verifichiamolo solamente per Francesco:

$$\begin{aligned} u_f(s) &= u_f(B,B)\cdot s_f(B)\cdot s_g(B) + u_g(V,V)\cdot s_f(V)\cdot s_g(V) \\ &= 2\cdot\frac{2}{3}\cdot\frac{1}{3} + 1\cdot\frac{2}{3}\cdot\frac{1}{3} \\ &= \frac{2}{3}. \end{aligned}$$

Si noti che, in questo esempio, ogni equilibrio di Nash a strategie pure *domina in modo paretiano* l'equilibrio di Nash a strategie miste [2].

		Giocatore$_2$	
		Testa	Croce
Giocatore$_1$	Testa	1, -1	-1, 1
	Croce	-1, 1	1, -1

Tabella 5: *Gioco a somma zero* (Esempio 2.4)

2.4.2 *Giochi a somma costante*

Definizione 2.21 (*Gioco a somma costante con due giocatori*)**.** Un gioco in forma normale con due giocatori si dice a *somma costante* se esiste una costante c tale che $u_1(a) + u_2(a) = c$ per ogni profilo strategico $a \in A_1 \times \ldots \times A_n$.

Un gioco a somma costante in cui $c = 0$ sarà detto *gioco a somma zero*; tale categoria di giochi rappresenta una situazione di pura competizione: il guadagno o la perdita di un partecipante è perfettamente bilanciato da una perdita o un guadagno di un altro partecipante, dunque qualunque risultato di un gioco di questo tipo sarà un ottimo paretiano [23]. Un qualunque gioco con n partecipanti può essere convertito in un gioco a somma zero aggiungendo un giocatore fittizio i cui valori di payoff sono scelti ad hoc per rendere la somma dei payoff in ogni risultato uguale a zero: per tale motivo consideriamo solamente giochi a somma zero con due giocatori [2].

Esempio 2.4 (*Abbinamento dei penny*)**.** Abbiamo due giocatori, ognuno possiede un penny e deve girarlo segretamente su testa (T) o croce (C). I giocatori rivelano contemporaneamente le proprie scelte e, se le scelte coincidono, il giocatore$_1$ si intasca il penny dell'avversario, altrimenti sarà il giocatore$_2$ ad ottenere la moneta del primo. Identificando i valori di payoff con i risultati monetari, possiamo rappresentare la situazione tramite la tabella 5. Questo gioco non ha equilibri di Nash a strategie pure, poiché non vi è una strategia pura che sia una miglior risposta ad una altrui miglior risposta. Tuttavia, ha un equilibrio di Nash a strategie miste identificato dal seguente profilo strategico $s = (s_1, s_2) = ((\frac{1}{2}, \frac{1}{2}), (\frac{1}{2}, \frac{1}{2}))$. È interessante osservare come l'utilità attesa (2.1) per entrambi i giocatori sia zero [2]; verifichiamolo solamente per il primo:

$$\begin{aligned} u_1(s) &= u_1(T,T) \cdot s_1(T) \cdot s_2(T) + u_1(T,C) \cdot s_1(T) \cdot s_2(C) + \\ &+ u_1(C,T) \cdot s_1(C) \cdot s_2(T) + u_1(C,C) \cdot s_1(C) \cdot s_2(C) = \\ &= \frac{1}{4} - \frac{1}{4} + \frac{1}{4} - \frac{1}{4} = 0. \end{aligned}$$

2.4.3 *Giochi con equilibri a strategie dominanti*

Tramite quest'ultima coppia di esempi vediamo un accenno di quello che è il *"design di meccanismi"*: introduciamo un gioco privo di strategie dominanti e vediamo come sia possibile modificarlo per ottenere un nuovo gioco con un equilibrio a strategie dominanti.

Esempio 2.5 (*Asta con offerta segreta*)**.** Ad un'asta con n partecipanti sta per essere venduto un oggetto. Ogni partecipante associa all'oggetto un valore v_i, la propria valutazione. Assumiamo che $v_1 > v_2 \cdots > v_n > 0$. L'asta si svolge nel seguente modo: i partecipanti inoltrano contemporaneamente le proprie offerte ($b_{i=1,\dots,n} > 0$) al banditore, l'oggetto è dato al giocatore con l'indice più basso tra coloro che presentano l'offerta più alta. Il vincitore dell'asta dovrà pagare un corrispettivo pari alla propria offerta b_i. Dato un profilo di azioni $b = (b_1, \dots, b_n)$, la funzione utilità sarà definita nel seguente modo:

$$u_i(b) = \begin{cases} v_i - b_i, & \text{se } max(b) = b_i, \\ 0, & \text{altrimenti.} \end{cases}$$

In tal gioco non ci sono strategie dominanti: il giocatore che ha fatto l'offerta più alta avrebbe potuto migliorare il proprio payoff facendo un'offerta più bassa; per esempio, se avesse offerto appena poco più della seconda offerta più alta, avrebbe vinto comunque, ma avrebbe pagato un prezzo più basso per lo stesso oggetto [22]. In sostanza, la miglior strategia di un giocatore dipende dalla assunzioni che fa al riguardo delle strategie degli altri partecipanti. Avendo modo di analizzare tale situazione più nel dettaglio, potremmo osservare come in tutti gli equilibri di Nash del gioco sia il giocatore uno ad ottenere l'oggetto [3].

Esempio 2.6 (*Asta di Vickrey*)**.** Questa tipologia di asta si svolge esattamente come la precedente, ma con un'unica differenza: il vincitore dovrà pagare un corrispettivo pari alla seconda offerta più alta. Dunque, dato un profilo di azioni $b = (b_1, \dots, b_n)$, la funzione utilità sarà ridefinita nel seguente modo:

$$u_i(b) = \begin{cases} v_i - b_j, & \text{se } max(b) = b_i \text{ e } b_i > b_j > b_{k \neq i,j}\ , \\ 0, & \text{altrimenti.} \end{cases}$$

Nell'asta di Vickrey la strategia dominante per ogni giocatore equivale a fare un'offerta pari al proprio valore di valutazione dell'oggetto, dunque il profilo strategico $s = (b_1 = v_1, b_2 = v_2, \dots, b_n = v_n)$ è un *equilibrio a strategie dominanti* [22]. Differentemente dall'esempio 2.5, il giocatore (g_i) che si è aggiudicato l'offerta non è incentivato a cambiare la propria strategia; infatti, supponendo che b_j sia la seconda offerta più alta, nel caso in cui g_i avesse offerto meno della

propria valutazione ($b_i < v_i$) sarebbe potuto incorrere in due situazioni: la sua offerta avrebbe potuto continuare ad essere quella più alta e avrebbe pagato comunque un valore pari a b_j oppure, nel caso peggiore, avrebbe potuto perdere ($b_i < b_j$). Allo stesso modo anche gli altri giocatori, nonostante non si siano aggiudicati l'oggetto, non sono incentivati a cambiare la propria strategia: se g_j avesse fatto un'offerta $b_j > b_i > v_j$, si sarebbe trovato a pagare una somma pari a b_i, dunque superiore alla propria valutazione dell'oggetto; ed infatti, in questo caso, il valore di payoff di g_j sarebbe stato negativo.

3

MULTI-PARTY COMPUTATION

3.1 INTRODUZIONE ALLA MULTI-PARTY COMPUTATION

3.1.1 *Problema MPC*

La computazione congiunta a più partecipanti (*Multi-Party Computation*, MPC), letteralmente computazione a parti multiple, è un campo della crittografia il cui obbiettivo è definire metodi che permettano a più partecipanti, detti anche giocatori, di calcolare in modo congiunto una funzione che riceve in ingresso un input da ognuno di essi. Formalmente il problema della computazione congiunta a più partecipanti, o problema MPC (introdotto per la prima volta, nella sua forma a due partecipanti, da Yao in [9]), può essere definito nel seguente modo: abbiamo $n \geqslant 2$ giocatori, $P_1, \ldots, P_n$; ogni giocatore P_i possiede un valore di input t_i, con $1 \leqslant i \leqslant n$. I partecipanti desiderano calcolare congiuntamente ed "in modo sicuro" una funzione $s = f(t_1, \ldots, t_n)$ che riceve in entrata i valori di input dei vari giocatori. L'obbiettivo è fare in modo che ogni giocatore apprenda l'output della funzione, s, senza però apprendere alcuna informazione riguardo gli input degli altri partecipanti, escludendo ovviamente quello che potrebbe dedurre dalla coppia (t_i, s). Per "in modo sicuro" si intende non solo la garanzia della *segretezza* degli input dei giocatori, ma anche la *correttezza* dell'output, anche in situazioni in cui alcuni giocatori "barino" [24].

Possiamo generalizzare la situazione sopra descritta per due categorie di funzioni da computare:

- *Funzioni probabilistiche.* In tal caso il valore della funzione dipende da una stringa casuale r generata secondo una qualche distribuzione: $s = f(t_1, \ldots, t_n, r)$. È di cruciale importanza che la stringa r non sia controllata da alcuno dei partecipanti, ma sia in qualche modo generata congiuntamente durante la computazione.

- *Funzioni multi-output.* Non è necessario che la funzione da computare ritorni un singolo valore; in generale potrebbe ritornare un valore di output unico per ogni partecipante alla computazione, $(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$: in questo caso ogni giocatore P_i apprenderebbe unicamente l'output s_i ed ovviamente anche tutto ciò che potrebbe dedurre dalla coppia (t_i, s_i) [25].

Cerchiamo di rendere più concreto il problema MPC tramite un esempio.

Esempio 3.1 (*Problema dei salari*)**.** Supponiamo di avere tre amici, Anna, Bernardo e Carlo. I tre hanno dei salari rispettivamente pari ad x, y e z, e vorrebbero scoprire quale dei tre sia quello maggiore senza però rivelare alcuna informazione riguardo ai proprio guadagni. Vogliono quindi computare la funzione $F(x, y, z) = max(x, y, z)$. Supponiamo inoltre che abbiano un amico in comune, fidato e capace di mantenere un segreto, Tommaso. I tre potrebbero confidare quanto guadagnano a Tommaso, il quale, una volta calcolato il massimo, potrebbe comunicar loro l'esito della computazione. L'obbiettivo dunque è progettare un protocollo che, unicamente tramite lo scambio di messaggi tra Anna, Bernardo e Carlo, permetta loro di apprendere il valore $F(x, y, z)$ senza rivelare chi guadagni quanto e senza fare affidamento su Tommaso. Tale protocollo dovrebbe permettere di conoscere niente di più e niente di meno di quanto i partecipanti avrebbero appreso se avessero interagito tramite un amico incorruttibile e perfettamente affidabile (trusted outside party). In particolare, ogni giocatore potrà apprendere tutto ciò che è possibile dedurre dall'output e dal proprio input. Dunque, se l'output fosse z, allora Carlo saprebbe che il suo z è il valore massimo, mentre Anna e Bernardo, nel caso in cui x, y e z fossero distinti, saprebbero che il loro input non è il massimo e che il massimo è equivalente a z.

Quindi, l'obbiettivo è definire un *protocollo MPC* che garantisca le due seguenti proprietà:

- *Input privacy.*
- *Correttezza della computazione [26].*

3.1.2 *Avversari e loro capacità*

È fondamentale sottolineare che un protocollo MPC deve permettere di calcolare il valore di una data funzione anche sotto l'assunzione che qualcuno dei partecipanti possa deviare dal protocollo stesso.

Partecipanti onesti o disonesti. Identificheremo come onesti (*honest*) i partecipanti che seguono il protocollo e come disonesti (*faulty*) coloro che non lo seguono. Ovviamente i giocatori onesti non saranno a conoscenza di quali siano i giocatori disonesti, almeno non inizialmente.

Avversario. Tipicamente, nel contesto della crittografia, viene concepita una singola entità, detta *avversario* o *attaccante*, che coordina le attività di tutti i partecipanti disonesti al fine di infrangere le proprietà garantite dal protocollo. Tale avversario può essere uno dei giocatori oppure un'entità esterna in grado di corrompere un loro sottoinsieme; nel secondo caso, possiamo pensare all'avversario come ad un hacker che cerca di irrompere nei computer dei giocatori. Una volta che un partecipante è corrotto, l'avversario avrà accesso a tutti i dati posseduti

dal giocatore, incluse informazioni complete sulle azione ed i messaggi che quest'ultimo ha ricevuto internamente al protocollo fino a quel momento.

Corruzione passiva o attiva. È possibile distinguere tra corruzione passiva ed attiva. Per *corruzione passiva* intendiamo il caso in cui un attaccante accede a tutte le informazioni relative ai giocatori corrotti, ma quest'ultimi continuano ad eseguire il protocollo in modo corretto. Mentre nel caso di *corruzione attiva* l'avversario avrà totale controllo sui giocatori corrotti.

Struttura dell'avversario e valore di soglia. Un avversario $\mathcal{A}$ controlla tutti i partecipanti corrotti e coordina le loro azioni. Tale "collaborazione" tra giocatori corrotti porta l'avversario ad avere una conoscenza maggiore che potrà essere utilizzata per influenzare la computazione. È dunque fondamentale notare che nessun protocollo potrà essere sicuro se ogni giocatore può essere corrotto: ciò ci fa capire che è necessario porre alcune limitazioni. Sia N l'insieme dei partecipanti al protocollo, definiamo la *struttura dell'avversario* Γ come una famiglia di sottoinsiemi di N. Definiamo l'avversario $\mathcal{A}$ in modo tale che possa corrompere un sottoinsieme di giocatori $A \subset N$ solo se $A \in \Gamma$. Per poter riuscire a raggiungere una soluzione al problema, sarà necessario limitare il numero di giocatori che possono essere corrotti. Tale valore limite, detto *valore di soglia* (threshold) k, ci indicherà che il protocollo rimane sicuro fintato che il numero di partecipanti corrotti è inferiore a k. Dunque la struttura dell'avversario sarà costituita da tutti i sottoinsiemi di N con cardinalità minore del valore di soglia ($\Gamma = \{A_i |\ A_i \subset N,\ |A_i| \leqslant k\ )$. È importante sottolineare che i protocolli e le soluzioni per i casi in cui $k \leqslant \frac{n}{2}$ (quindi nel caso in cui sia assunta una maggioranza di partecipanti onesti) sono molto diversi da quelli in cui tale assunzione non viene fatta. [1]

Strategia di corruzione. La strategia di corruzione ci indica quando e come un avversario corrompe i giocatori: ci mostra dunque in che modo si evolve l'insieme dei giocatori corrotti. Esistono due modelli principali:

- *Corruzione statica.* In questo caso l'avversario controlla un insieme fisso di giocatori, che quindi risulta essere *statico*, cioè selezionato una volte per tutte nel momento in cui il protocollo comincia.
- *Corruzione adattiva.* L'avversario possiede la capacità di scegliere e corrompere un nuovo giocatore in qualunque momento durante l'esecuzione del protocollo (fintanto che non si supera il valore di soglia k). La scelta di chi corrompere e quando corromperlo può essere presa dall'avversario in modo arbitrario e può dipendere dalla propria visione dell'esecuzione; per

1 per la precisione chiamiamo Γ, un sottoinsieme dell'insieme potenza 2^N di N, *struttura* per N se, per ogni $S \in \Gamma$ e per ogni $S' \subset S$, allora $S' \in \Gamma$.

tale ragione, questa tipologia di corruzione è detta adattiva. Una volta che un giocatore è corrotto lo sarà per sempre, dunque l'insieme dei giocatori corrotti è *dinamico* e cresce fino a raggiungere il valore di soglia.

Un modello addizionale, chiamato modello *proattivo*, considera la possibilità che i partecipanti siano corrotti solamente per un certo lasso di tempo. Dunque i giocatori onesti possono essere corrotti durante la computazione (esattamente come nel caso della corruzione adattiva), ma possono tornare ad essere nuovamente onesti [27].

Comportamento degli avversari. In un primo momento possiamo schematicamente dividere gli avversari in due categorie in base a quanto essi siano intenzionati a deviare dal protocollo. Ogni categoria darà luogo ad una diversa forma di sicurezza e richiederà protocolli diversi per trattarla [24] [26].

- *Avversario onesto-ma-curioso (corruzione passiva).* Spesso detto anche avversario semi-onesto. In questa situazione assumiamo che i giocatori corrotti semplicemente cooperino per raccogliere informazioni sensibili senza però deviare dalle specifiche del protocollo. Protocolli MPC progettati modellando gli avversari secondo tale categoria saranno detti *debolmente sicuri* e saranno unicamente in grado di prevenire la fuga involontaria di informazioni tra le parti (*sicurezza passiva*). Nonostante ciò, tali protocolli sono molto efficienti e spesso rappresentano il primo passo per raggiungere un livello di sicurezza più elevato.

- *Avversario malevolo (corruzione attiva).* In questa situazione assumiamo che i giocatori corrotti possano scostarsi arbitrariamente dall'esecuzione del protocollo nel tentativo di imbrogliare. Protocolli MPC modellati per contrastare avversari malevoli forniranno garanzie di sicurezza elevate e saranno detti *fortemente sicuri*. In presenza di tali protocolli, un avversario potrà ottenere dei risultati solamente quando il numero dei giocatori corrotti risulterà maggiore rispetto al numero dei giocatori onesti, ed anche in tal caso potrà unicamente forzare i giocatori onesti ad annullare l'esecuzione della computazione (infatti, quest'ultimi avranno rilevato la presenza dei giocatori corrotti ed i tentativi di infrangere il protocollo). Protocolli MPC fortemente sicuri garantiscono la segretezza degli input e la correttezza e segretezza dell'output nel caso in cui i giocatori onesti ne ricevano uno (*sicurezza attiva*).

È interessante sottolineare che è possibile modellare i comportamenti degli avversari anche in modo meno rigido e più realistico, dando così luogo a diverse e nuove forme di sicurezza:

- *Avversario sotto copertura.* Dal momento che la sicurezza contro gli avversari malevoli è spesso possibile solo a costo di ridurre l'efficienza, si è portati

a considerare una forma rilassata di sicurezza attiva, chiamata *sicurezza sotto copertura* [28]. Protocolli che garantiscono sicurezza sotto copertura sono modellati su avversari che possono deviare in modo arbitrario dal protocollo, ma che non desiderano essere "catturati" nel farlo. Tali avversari, detti *avversari sotto copertura*, in un certo senso sono avversari attivi costretti ad agire in modo passivo per motivazioni non crittografiche esterne; valutano il rischio di essere scoperti rispetto ai benefici apportati dal possibile comportamento scorretto, quindi in determinate situazioni potrebbero scegliere di correre tale rischio. Viene modellata una situazione più realistica dove giocatori disonesti spesso preferiscono seguire il protocollo per paura di danneggiare la propria reputazione, impedendo dunque una futura collaborazione con altri partecipanti onesti. Protocolli che forniscono sicurezza sotto copertura permettono ai partecipanti onesti di rilevare deviazioni con una probabilità minima stabilita a priori, scegliendola in base al caso specifico (più la reputazione dei partecipanti è rilevante e più basso potrà essere tale valore). Tali protocolli sono più efficienti rispetto a quelli che garantiscono sicurezza attiva e più sicuri di quelli che garantiscono sicurezza passiva.

- *Avversario razionale.* L'avversario razionale ha un certo obiettivo ben definito (specificato da una funzione di utilità) e sarà portato a deviare dal protocollo unicamente al fine di massimizzare la propria funzione di utilità. Tale tipologia di avversari è modellata unendo concetti della crittografia e concetti della teoria dei giochi ed ha prodotto interessanti risultati per applicazioni specifiche come ad esempio le aste elettroniche [29].

Potere computazionale degli avversari. Possiamo distinguere gli avversari in due categorie basandoci sul loro potere computazionale [10]:

- *Avversari limitati computazionalmente.* Avversari che possiedono un potere computazionale limitato. Per quantificare il limite computazionale di un avversario $\mathcal{A}$, faremo riferimento alla classe delle macchine di Turing probabilistiche a tempo polinomiale (*Probabilistic Polynomial Time, PPT*): dunque tali avversari saranno anche detti *Avversari PPT*. Avversari PPT usano un certo grado di randomicità come parte della propria logica (ovvero $\mathcal{A}$ è probabilistico) e, per ogni input $x \in \{0,1\}^*$ ($\{0,1\}^*$ è l'insieme di stringhe a lunghezza arbitraria), l'esecuzione di $\mathcal{A}(x)$ termina in un numero di passi n polinomiale nella lunghezza della stringa x.

- *Avversari illimitati computazionalmente.* Avversari non legati a nessuna classe di complessità, possiedono un potere computazionale illimitato.

3.2 SICUREZZA NEI PROTOCOLLI MPC

3.2.1 *Requisiti di sicurezza MPC*

Un protocollo MPC sicuro deve essere in grado di resistere a qualunque possibile attacco di un avversario. Al fine di dimostrare ed affermare formalmente che un protocollo MPC è sicuro, è necessaria una precisa definizione di sicurezza per la computazione congiunta a più giocatori: ne esistono varie, ma tutte hanno come obbiettivo principale quello di garantire una serie di requisiti fondamentali.

Per prima cosa vediamo quali sono i più importanti *requisiti di sicurezza MPC*:

- *Privacy.* Garantisce che i partecipanti apprendano unicamente il proprio valore di output. In particolare, le uniche informazioni che possono essere apprese da un giocatore sono quelle che è possibile derivare dal proprio input e dall'output ricevuto.
- *Correttezza della computazione.* Garantisce ad ogni giocatore che l'output ricevuto sia corretto. Nessun avversario potrà essere in grado di alterare il risultato della computazione.
- *Indipendenza degli input.* Garantisce che i giocatori corrotti scelgano i propri valori di input indipendentemente dagli input dei giocatori onesti. È importante notare che tale proprietà non è implicata dalla *privacy degli input*: ad esempio, nel contesto di un'asta on line, è possibile che un giocatore corrotto generi la puntata più alta senza però conoscere quale sia il precedente valore massimo di puntata (dato uno schema di cifratura e data la codifica secondo tale schema di un valore naturale $n \in \mathbb{N}$, è possibile generare una codifica valida di $n+1$ senza però conoscere il valore originale cifrato (n)).
- *Consegna dell'output garantita.* Garantisce che i partecipanti corrotti non siano in grado di impedire alle parti oneste di ricevere il loro output. In altre parole, l'avversario non dovrebbe essere in grado di interrompere la computazione effettuando un attacco "Denial of Service".
- *Equità (Fairness).* Garantisce che i valori di output siano ricevuti dai partecipanti corrotti se e solo se sono ricevuti anche dai partecipanti onesti. Non è consentito lo scenario in cui un giocatore corrotto ottiene il proprio output e lo nega ai giocatori onesti.

È importante sottolineare che la lista appena stilata non costituisce una definizione di sicurezza, ma bensì un insieme di requisiti che ogni protocollo sicuro dovrebbe garantire. Dunque, un possibile approccio per dare una definizione di sicurezza è quello di generare una lista di requisiti distinti, esattamente come

abbiamo fatto sopra, e poi affermare che un protocollo è sicuro se e solo se soddisfa ognuno di essi [24] [27].

3.2.2 *Paradigma di simulazione reale/ideale*

L'approccio descritto nella sottosezione precedente non è soddisfacente. In primo luogo, è possibile che un requisito importante venga tralasciato e non incluso nella lista. Ciò è particolarmente vero in quanto diverse applicazioni richiedono che vengano rispettate proprietà diverse; vorremmo quindi una definizione abbastanza generale da catturare i requisiti richiesti da tutte le applicazioni. In secondo luogo, per rendere banale constatare che tutti i possibili attacchi degli avversari sono impediti dalla definizione proposta, è necessario che essa sia sufficientemente semplice.

Per ottenere una definizione di protocollo sicuro che formalizzi il concetto di sicurezza in modo più generale, utilizziamo il così detto *paradigma di simulazione reale/ideale* (introdotto per la prima volta da Goldreich in [30]). Facendo un esperimento mentale, consideriamo un "mondo ideale" in cui esista una terza parte fidata ed incorruttibile (*trusted third party*, TTP) disposta ad aiutare i giocatori a portare a termine la propria computazione: dunque i giocatori possono semplicemente inviare i propri input al TTP che calcolerà la funzione desiderata ed in seguito passerà ad ogni partecipante l'output appropriato. In tale "mondo ideale" l'unica libertà che viene lasciata ai singoli giocatori è quella di inviare i propri input al TTP; di conseguenza, l'unica libertà lasciata all'avversario è quella di scegliere gli input dei giocatori corrotti.

È importante notare che in questa computazione ideale tutti i *requisiti di sicurezza MPC* enunciati nella sottosezione 3.2.1 (e molti altri) sono garantiti. Il TTP consegnerà ad ogni partecipante unicamente il proprio output, dunque la *privacy* è garantita. Il TTP non può essere corrotto e calcolerà sempre la funzione in modo corretto, quindi la *correttezza della computazione* è garantita. Infine *consegna degli output garantita* e *equità (fairness)* sono sempre assicurate dalla presenza del TTP che si occuperà di consegnare i valori di output a tutti i partecipanti.

Naturalmente, nel "mondo reale" non esisterà alcun TTP; piuttosto i partecipanti eseguiranno tra di loro, senza nessun aiuto esterno, un protocollo di scambio messaggi.

Quindi, abbiamo da un lato il "mondo reale" o *modello reale* (dove il protocollo e gli attacchi ad esso hanno luogo) e dall'altro un "mondo ideale" o *modello ideale* (da noi definito, che in sostanza specifica ciò che vorremmo che il protocollo facesse); possiamo dunque ritenere sicuro un protocollo se quello che

produce non può essere distinto da quello che si otterrebbe nel "mondo ideale". Riformulando il concetto, possiamo dire che un protocollo è sicuro se nessun avversario può compromettere la computazione nel "mondo reale" più di quanto non la possa compromettere nel "mondo ideale": per ogni avversario che porta a termine con successo un attacco ad un protocollo sicuro nel "modo reale", esisterà un avversario che porta a termine con successo lo stesso attacco nel "modo ideale"; tuttavia, nessun attacco può essere portato a termine con successo nel "mondo ideale", perciò possiamo concludere che tutti gli attacchi avversari ad un protocollo sicuro in esecuzione nel "mondo reale" dovranno fallire [27] [10] [24].

In modo più formale, la sicurezza di un protocollo è stabilita comparando i risultati dell'esecuzione di un protocollo reale con i risultati di una computazione ideale. È necessario capire cosa si intende in questo caso per risultati: formalmente l'interazione di un avversario con i partecipanti nel protocollo genera una *trascrizione*. Detta *trascrizione* è una variabile aleatoria che include gli output di tutti i giocatori onesti e l'output dell'avversario. L'output dell'avversario è costituito da tutti le informazioni che è riuscito ad apprendere: i propri input (cioè gli input dei giocatori corrotti)e tutti i messaggi che i giocatori onesti gli hanno inviato (in base al modello di comunicazione potrebbe anche contenere ulteriori informazioni, ad esempio i messaggi pubblici che i partecipanti onesti si sono scambiati). Dunque, una *trascrizione* è una variabile aleatoria la cui distribuzione di probabilità dipenderà dal particolare protocollo usato per la computazione, dal vettore di input su cui viene fatto partire il protocollo, dall'interazione dei partecipanti con l'avversario e da un particolare parametro di sicurezza (vedremo poi che tutte le caratteristiche di complessità dei nostri modelli saranno misurate in riferimento a tale parametro di sicurezza). Una trascrizione, definita in modo equivalente, viene generata anche nel "modo ideale" dall'interazione dei partecipanti e dell'avversario con un TTP. Formalmente:

Definizione 3.1 (*Trascrizione reale* $\mathrm{REAL}_{\mathcal{A},\pi}(t,\lambda)$)**.** Sia f una funzione che riceve come argomenti n valori in input, sia π un protocollo MPC per n partecipanti che calcola la funzione f e sia λ un parametro di sicurezza. Dato un avversario $\mathcal{A}$ che controlla un sottoinsieme dei partecipanti ($A \subset N$) e fatto partire il protocollo π con ingresso il vettore di input $t = (t_1, \dots t_n)$ sotto l'attacco di $\mathcal{A}$, l'output di $\mathcal{A}$ unito alla sequenza di output ottenuti dei giocatori onesti sarà detto *trascrizione reale* e verrà indicato con $\mathrm{REAL}_{\mathcal{A},\pi}(t,\lambda)$.

Definizione 3.2 (*Trascrizione ideale* $\mathrm{IDEAL}_{\mathcal{A}',f}(t,\lambda)$)**.** Sia f una funzione che riceve come argomenti n valori input, sia T un TTP nel "mondo ideale" in grado di calcolare la funzione f e sia λ un parametro di sicurezza. Dato un avversario $\mathcal{A}'$ che controlla un sottoinsieme dei partecipanti ($A \subset N$), definiamo $\mathrm{IDEAL}_{\mathcal{A}',f}(t,\lambda)$ come l'output di $\mathcal{A}'$ unito alla sequenza di output dei giocatori

onesti calcolati da T nel "mondo ideale" con input il vettore t; tale sequenza sarà detta *trascrizione ideale*.

Prima di poter proseguire è necessario fornire un'ulteriore definizione:

Definizione 3.3 (*Insieme di distribuzioni (distribution ensemble)*)**.** Data una variabile aleatoria $X(t, \lambda)$ indicizzata tramite t e λ , definiamo un *insieme di distribuzioni* $X = \{X(t, \lambda)\}_{t \in D, \lambda \in \mathbb{N}}$ come una sequenza infinita di distribuzioni di probabilità (formalmente è una collezione di variabili aleatorie), dove una distribuzione $X(t, \lambda)$ è associata ad ogni valore di $\lambda \in \mathbb{N}$ e $t \in D$, per qualche dominio D [31].

L'insieme di distribuzioni $\{REAL_{\mathcal{A},\pi}(t, \lambda)\}_{t \in \{0,1\}^*, \lambda \in \mathbb{N}}$ e l'insieme di distribuzioni $\{IDEAL_{\mathcal{A}',f}(t, \lambda)\}_{t \in \{0,1\}^*, \lambda \in \mathbb{N}}$ saranno indicati rispettivamente con $REAL_{\mathcal{A},\pi}$ e $IDEAL_{\mathcal{A}',f}$. Il parametro t corrisponde ai vari possibili vettori di input e λ è il parametro di sicurezza.

Dunque, se mostriamo che l'esecuzione di un protocollo che interagisce con un avversario nel "mondo reale" genera esattamente lo stesso insieme di distribuzioni che viene generato da un avversario equivalente che interagisce con un TTP nel "mondo ideale", allora possiamo garantire che nessuna informazione è trapelata dall'esecuzione del protocollo. Diamo una definizione rigorosa:

Definizione 3.4 (*Computazione sicura*)**.** Diremo che π, protocollo MPC per n giocatori, *calcola in modo sicuro* f se, per ogni avversario $\mathcal{A}$ che controlla un sottoinsieme dei partecipanti ($A \subset N$), esisterà un avversario $\mathcal{A}'$, che controlla gli stessi giocatori nel "mondo ideale", tale che l'insieme di distribuzioni $REAL_{\mathcal{A},\pi}$ sia "indistinguibile" dall'insieme di distribuzioni $IDEAL_{\mathcal{A}',f}$ (per il momento non approfondiamo il significato del termine "indistinguibile") [31].

In un protocollo π che *calcola in modo sicuro* f i *requisiti di sicurezza MPC* (vedere sottosezione 3.2.1) sono garantiti. Dunque, sia P_i un partecipante corrotto con valore di input t_i, gli unici comportamenti "scorretti" che può avere (tali comportamenti sono dunque ammissibili sia nel "mondo ideale" che nel "mondo reale") sono i seguenti:

- *Mentire sul proprio valore di input.* P_i corrotto potrebbe "mentire" ed usare un falso valore di input t_i' al posto di t_i. In tal caso né il protocollo π né il TTP potranno sapere che t_i' non è il vero valore di input di P_i , dunque t_i' risulterà come il valore di input di P_i nella *trascrizione reale* e *ideale*.

- *Non trasmettere alcun valore di input.* P_i potrebbe anche rifiutarsi di trasmettere il proprio valore t_i. In questo caso la situazione può essere gestita in due modi diversi: la computazione viene interrotta oppure viene assegnato un valore di default a t_i. Assumiamo che il dominio della funzione f contenga un simbolo speciale $\perp$ che indica l'assenza di input; in tal modo è possibile ben definire come computare f nel caso in cui alcuni input non

vengano forniti. Dunque, un protocollo, se rileva che un partecipante non ha fornito il proprio input, dovrà gestire la situazione nell'esatto modo in cui verrebbe gestita se un partecipane immettesse il valore di input $\perp$ nel modello ideale.

È possibile dare una definizione di sicurezza meno forte e richiedere che un protocollo soddisfi unicamente i requisiti di *privacy* e *correttezza della computazione*; in tal caso, i partecipanti corrotti potrebbero apprendere i propri output ed impedire ai partecipanti onesti di apprendere i loro [25].

3.2.3 *Variazioni del modello*

Le nozioni di sicurezza di base introdotte fino ad ora sono universali ed indipendenti dal modello adottato. Tuttavia, implementazioni specifiche dipendono in modo esplicito dalle caratteristiche del modello in cui verrà effettuato il calcolo. In particolare, devono essere specificati i seguenti aspetti: *tipologia di avversario*, *presupposti e modelli di comunicazione* e *assunzioni computazionali*.

Tipologia di avversario

La tipologia di avversario viene specificata dal *comportamento dell'avversario* (onesto-ma-curioso, malevolo, sotto copertura o razionale), dal *valore di soglia* (numero massimo k di partecipanti corrompibili dall'avversario), dalla *strategia di corruzione* (adattiva o statica) adottata dall'avversario.

Presupposti e modelli di comunicazione

Presupposti di comunicazione. Due presupposti di comunicazione comuni sono l'esistenza di un *canale sicuro* e l'esistenza di un *canale di broadcast*. I canali sicuri assicurano che ogni coppia di partecipanti P_i e P_j sia collegata tramite un canale di comunicazione privato e autenticato: dunque l'avversario non potrà intercettare messaggi o interferire con la comunicazione tra le parti oneste. Un canale di broadcast è un canale con le seguenti proprietà: se un partecipante P_i (onesto o disonesto) invia un messaggio m, allora m sarà correttamente ricevuto da tutti i partecipanti (che saranno anche certi che il messaggio provenga da P_i); perciò se un partecipante onesto riceve un messaggio m, allora saprà che ogni altro partecipante onesto avrà ricevuto m.

Un diverso presupposto di comunicazione è l'esistenza di *buste* (envelopes). Una busta, nella sua definizione più generale, garantisce le seguenti proprietà: un messaggio m può essere memorizzato all'interno della busta ed il suo valore sarà conservato in modo segreto (senza esporlo) per un dato periodo di tempo, allo scadere del quale il valore di m verrà rivelato senza alcuna modifica. Una *scatola di scheda* (ballot box) rappresenta un ampliamento del modello basato su

busta che fornisce anche un meccanismo di rimescolamento casuale delle buste.

Tali presupposti di comunicazione rappresentano ovviamente delle astrazioni idealizzate che, separando i problemi di comunicazione da quelli computazionali, ci permettono di descrivere un protocollo in maniera chiara e semplice. Queste assunzioni possono essere realizzate mediante meccanismi fisici, anche se in alcuni contesti tali meccanismi potrebbero non essere disponibili. Dunque, alle volte, sarà necessario rimuovere alcuni presupposti di comunicazione: ad esempio, un canale sicuro, introducendo dei prerequisiti computazionali e un'infrastruttura a chiave pubblica, potrebbe essere sostituito da un protocollo [25].

Modelli di comunicazione. In generale la comunicazione può essere *sincrona* oppure *asincrona*. Nel caso in cui la comunicazione sia sincrona, i partecipanti invieranno i messaggi in modo sincrono, dunque il protocollo procederà in turni: durante ogni turno, ogni giocatore invierà un messaggio ad un altro giocatore e tutti i messaggi saranno consegnati prima che il turno successivo cominci. L'avversario potrà vedere tutti i messaggi inviati dai giocatori onesti a quelli corrotti (o tutti i messaggi nel caso ne abbia facoltà), se adotta una strategia di corruzione adattivo potrà decidere di corrompere alcuni partecipanti onesti ed infine deciderà quali messaggi inviare in vece dei giocatori corrotti. Nel modello di comunicazione asincrona la consegna dei messaggi non è garantita e non ci sono certezze sul tempo di ricezione; nonostante ciò, il problema MPC è comunque risolvibile anche se in modo più debole (se i messaggi non sono consegnati, non è possibile garantire che un protocollo generi alcun output) [27].

Assunzioni computazionali

Come abbiamo precedentemente visto, gli avversari possono essere suddivisi in due categorie in base al loro potere computazionale, ovvero avversari limitati ed illimitati computazionalmente. Tale distinzione produce due differenti modelli per la computazione sicura: il *modello di teoria dell'informazione (information-theoretic model)* ed il *modello computazionale (computational model)*.

Modello di teoria dell'informazione. In tale modello assumiamo che l'avversario possieda potere computazionale illimitato. Di conseguenza è necessario assumere che i partecipanti alla computazione congiunta possano comunicare tramite canali sicuri: in tal modo l'avversario, nonostante il potere computazionale illimitato, non riceverà alcuna informazione sui messaggi scambiati tra i giocatori onesti e la sicurezza potrà quindi essere garantita. In questo contesto il termine "indistinguibile" usato nella definizione 3.3 può essere formalizzato richiedendo che i due insiemi di distribuzioni siano identicamente distribuiti, pertanto parleremo di *sicurezza perfetta (perfect security)*, oppure, almeno, statisticamente

"vicine", ed in questo caso parleremo di *sicurezza statistica (statistical security)*.

Vediamo in modo formale cosa intendiamo per *sicurezza perfetta*:

Definizione 3.5 (*Insiemi di distribuzioni identicamente distribuiti*)**.** Siano X e Y due insiemi di distribuzioni 3.3 . Diremo che X e Y sono *identicamente distribuiti* (e scriveremo $X \stackrel{d}{\equiv} Y$) se, per ogni t e per ogni λ, le distribuzioni $X(t,\lambda)$ e $Y(t,\lambda)$ sono identiche (sovraccarichiamo la notazione di significati scrivendo anche in questo caso $X(t,\lambda) \stackrel{d}{\equiv} Y(t,\lambda)$).

Definizione 3.6 (*Computazione perfettamente sicura*)**.** Diremo che π, protocollo MPC per n giocatori, *calcola in modo perfettamente sicuro* f se, per ogni avversario $\mathcal{A}$ che controlla un sottoinsieme dei partecipanti ($A \subset N$), esisterà un avversario $\mathcal{A}'$, che controlla gli stessi giocatori nel "mondo ideale", in modo tale che gli insiemi di distribuzioni $\mathrm{REAL}_{\mathcal{A},\pi}$ e $\mathrm{IDEAL}_{\mathcal{A}',f}$ siano identicamente distribuiti ($\mathrm{REAL}_{\mathcal{A},\pi} \stackrel{d}{\equiv} \mathrm{IDEAL}_{\mathcal{A}',f}$) [31] [32].

Formalizziamo anche il concetto di *sicurezza statistica*:

Definizione 3.7 (*Funzione trascurabile (negligible function)*)**.** Una funzione $\sigma\colon \mathbb{N} \to \mathbb{R}$ è *trascurabile*, indicato con $\sigma(n) = negl(n)$, se per ogni $c > 0$ esiste una costante n_0 tale che per ogni $n > n_0$ (informalmente diremo: per un n sufficientemente grande) si ha che $\sigma(n) < n^{-c}$ [10].

Ad esempio, le funzioni $2^{-\sqrt{n}}$ e $n^{-\log(2n)}$ sono trascurabili. Le funzioni trascurabili possiedono una proprietà interessante: per ogni funzione trascurabile σ e per ogni funzione polinomiale p, la funzione $\sigma'(n) = \sigma(n) \cdot p(n)$ è trascurabile ($negl(n) \cdot p(n) = negl(n)$). Da ciò segue che un evento che si verifica con probabilità trascurabile continuerà a verificarsi con probabilità trascurabile (molto improbabile che si verifichi) anche se l'esperimento verrà ripetuto un numero polinomiale di volte. In sostanza, un evento che accadrà con probabilità trascurabile accadrà così di rado che possiamo semplicemente non considerarlo [33].

Definizione 3.8 (*Distanza statistica*)**.** Date due variabili aleatorie X_0 e X_1 con valori in $\mathcal{X}$ insieme numerabile, la loro *distanza statistica* è

$$\Delta(X_0, X_1) = \frac{1}{2} \cdot \sum_{x \in \mathcal{X}} |\mathbb{P}[X_0 = x] - \mathbb{P}[X_1 = x]|^2 [10].$$

Definizione 3.9 (*Insiemi di distribuzioni statisticamente indistinguibili*)**.** Sia $\sigma\colon \mathbb{N} \to \mathbb{R}$ una funzione e siano X e Y due insiemi di distribuzioni. Diremo che X e Y hanno *distanza statistica* σ se per tutti i λ sufficientemente grandi e per tutti i t abbiamo che $\Delta(X(t,\lambda), Y(t,\lambda)) < \sigma(\lambda)$. Se σ è una funzione trascurabile diremo che X e Y sono *statisticamente indistinguibili* (e scriveremo $X \stackrel{s}{\approx} Y$) [31].

2 $\mathbb{P}[X = x] = P_X(x)$ rappresenta la probabilità che X assuma il valore x.

Definizione 3.10 (*Computazione statisticamente sicura*)**.** Diremo che π, protocollo MPC per n giocatori, *calcola in modo statisticamente sicuro* f se, per ogni avversario $\mathcal{A}$ che controlla un sottoinsieme dei partecipanti ($A \subset N$), esisterà un avversario $\mathcal{A}'$, che controlla gli stessi giocatori nel "mondo ideale", in modo tale che gli insieme di distribuzioni $\mathrm{REAL}_{\mathcal{A},\pi}$ e $\mathrm{IDEAL}_{\mathcal{A}',f}$ siano statisticamente indistinguibili ($\mathrm{REAL}_{\mathcal{A},\pi} \overset{s}{\approx} \mathrm{IDEAL}_{\mathcal{A}',f}$).

Modello computazionale. Nel modello computazionale, detto anche modello crittografico, assumiamo che l'avversario ed anche i partecipanti onesti posseggano potere computazionale limitato (avversario PPT). Più precisamente, assumiamo che il corrispondente problema MPC sia parametrizzato da un *parametro di sicurezza* λ; in tal caso: (a) tutti i calcoli e le comunicazioni dovranno essere realizzati in un numero di passi n polinomiale in λ (in tempo polinomiale in λ); e (b) le strategie di comportamento scorretto dei partecipanti corrotti sono anch'esse limitate ad essere eseguite in tempo polinomiale in λ. Inoltre, in questo contesto parliamo di *sicurezza computazionale (computationl security)* ed il termine "indistinguibile" usato nella definizione 3.3 può essere formalizzato utilizzando il concetto di indistinguibilità computazionale:

Definizione 3.11 (*Insiemi di distribuzioni computazionalmente indistinguibili*)**.** Sia $\sigma\colon \mathbb{N} \to \mathbb{R}$ una funzione e siano X e Y due insiemi di distribuzioni. Diremo che X e Y hanno *distanza computazionale* σ se per ogni attaccante $\mathcal{D}$ (per ogni algoritmo $\mathcal{D}$) PTT eseguibile in tempo n polinomiale in λ, per tutti i λ sufficientemente grandi e per tutti i t abbiamo che

$$|\mathbb{P}[\mathcal{D}(X(t,\lambda), 1^\lambda) = 1] - \mathbb{P}[\mathcal{D}(Y(t,\lambda), 1^\lambda) = 1]| < \sigma(\lambda).$$

Se σ è una funzione trascurabile diremo che X e Y sono *computazionalmente indistinguibili* (e scriveremo $X \overset{c}{\approx} Y$).

Osserviamo che la probabilità è presa sulle corrispondenti variabili aleatorie $X(t,\lambda)$ o $Y(t,\lambda)$ e sulla casualità interna dell'algoritmo probabilistico $\mathcal{D}$. L'avversario $\mathcal{D}$ restituisce un valore in $\{0,1\}$ come tentativo di distinguere $X(t,\lambda)$ da $Y(t,\lambda)$ e la stringa 1^λ gli viene data come input ausiliario [10] [33].

Definizione 3.12 (*Computazione computazionalmente sicura*)**.** Diremo che π, protocollo MPC per n giocatori, *calcola in modo computazionalmente sicuro* f se, per ogni avversario $\mathcal{A}$ che controlla un sottoinsieme dei partecipanti ($A \subset N$), esisterà un avversario $\mathcal{A}'$, che controlla gli stessi giocatori nel "mondo ideale", in modo tale che gli insieme di distribuzioni $\mathrm{REAL}_{\mathcal{A},\pi}$ e $\mathrm{IDEAL}_{\mathcal{A}',f}$ siano computazionalmente indistinguibili ($\mathrm{REAL}_{\mathcal{A},\pi} \overset{c}{\approx} \mathrm{IDEAL}_{\mathcal{A}',f}$).

Dunque, considerando il potere computazionale limitato e presupponendo che l'avversario non possa risolvere qualche problema computazionale, non sarà più necessario assumere che i partecipanti abbiano accesso a canali idealmente

Modello	Avversario	Condizione
computazionale	passivo	$k < n$
computazionale	attivo	$k < \frac{n}{2}$
teoria dell'informazione	passivo	$k < \frac{n}{2}$
teoria dell'informazione	attivo	$k < \frac{n}{3}$
teoria dell'informazione (con canale di broadcast)	attivo	$k < \frac{n}{2}$

Tabella 6: *Condizioni di soglia necessarie e sufficienti affinché MPC sicura sia possibile*

privati, che possono, infatti, essere implementati utilizzando la crittografia a chiave pubblica. Tuttavia, si presume che i canali di comunicazione siano autenticati; cioè, se due giocatori onesti comunicano, l'avversario sarà in grado di intercettare i messaggi scambiati, ma non potrà modificarli. Tale autenticazione può essere realizzata utilizzando firme digitali e un'infrastruttura a chiave pubblica. Ciò significa che la sicurezza può essere garantita solo in un senso crittografico. In particolare, è importante notare che il parametro di sicurezza λ determinerà la lunghezza delle chiavi crittografiche; più precisamente, individuerà la lunghezza dell'input necessario a qualche problema computazionalmente difficile affinché nessun avversario reale possa risolvere tale problema in una quantità di tempo ragionevole [27] [24].

3.3 PANORAMICA DEI RISULTATI PER MPC SICURA

Il problema MPC, fin dalla sua introduzione all'inizio degli anni ottanta [9], è stato studiato ampiamente; vediamo alcuni dei risultati più significativi senza però dimostrarli.

3.3.1 *Risultati classici nel caso di avversari a soglia*

I lavori di ricerca classici volti a risolvere il problema MPC si occupano di avversari a soglia (cioè in grado di corrompere al più $k < n$ giocatori, dove n è il numero totale di partecipanti alla computazione) categorizzati unicamente in base al tipo di corruzione (attiva o passiva).

Nel modello computazionale (caso della sicurezza computazionale), studiato per la prima volta da Goldreich, Micali e Wigderson [30] , la condizione necessaria e sufficiente per ottenere sicurezza computazionale è che si abbia: (a) $k < \frac{n}{2}$ nel caso di attaccanti attivi (avversari malevoli); (b) $k < n$ nel caso di attaccanti passivi (avversari onesti-ma-curiosi).

Nel modello di teoria dell'informazione (caso della sicurezza perfetta o statistica) Ben-Or, Goldwasser e Wigderson [34] hanno dimostrato che la sicurezza perfetta è ottenibile se e solo se: (a) $k < \frac{n}{3}$ nel caso di corruzione attiva; (b) $k < \frac{n}{2}$ nel caso di corruzione passiva. Rabin e Ben-Or [35] hanno evidenziato che, nel caso in cui sia presente un canale di broadcast, che aiuterebbe unicamente nel caso di corruzione attiva, la sicurezza perfetta è garantita unicamente nel caso in cui $k < \frac{n}{2}$.

Questi risultati classici sono riassunti nella tabella 6 [36].

3.3.2 *Risultati nel caso di avversario a soglia mista*

In un modello con avversari a soglia con *corruzione mista*, alcuni giocatori potranno essere corrotti in modo attivo ed altri in modo passivo. Sia k_p il valore di soglia per le parti corrotte passivamente e k_a il valore di soglia le parti corrotte attivamente, allora, dagli studi di Fitzi, Hirt e Maurer [37], sappiamo che si può ottenere sicurezza perfetta se e solo se $3t_a + 2t_p < n$.

3.3.3 *Risultati nel caso di avversario con struttura arbitraria*

I risultati ottenuti nei modelli con avversari a soglia possono essere generalizzati per modelli con avversari con struttura arbitraria (nel caso di corruzione sia passiva che attiva, ma non mista). Per discutere tale generalizzazione è necessario premettere una nuova definizione.

Definizione 3.13 (*Composizione di strutture*)**.** Date due strutture Γ^1 e Γ^2, definiamo la seguente operazione (commutativa e associativa) su strutture, denotata dall'operatore $\sqcup$: $\Gamma^1 \sqcup \Gamma^2 = \{S_1 \cup S_2 |\ S_1 \in \Gamma^1, S_2 \in \Gamma^2\}$ è la struttura costituita da tutte le unioni di un elemento di Γ^1 ed un elemento di Γ^2.

Dato l'insieme dei partecipanti alla computazione congiunta N e data la struttura dell'avversario Γ, sia $Q^2(\Gamma)$ la la condizione per cui non esistono due insiemi in Γ in grado di coprire tutto N, ovvero $Q^2(\Gamma) \Leftrightarrow N \notin \Gamma \sqcup \Gamma$. In modo analogo sia sia $Q^3(\Gamma)$ la la condizione per cui non esistono tre insiemi in Γ in grado di coprire tutto N, ovvero $Q^3(\Gamma) \Leftrightarrow N \notin \Gamma \sqcup \Gamma \sqcup \Gamma$.

Hirt e Maurer [38] hanno dimostrato che, nel caso in cui la struttura dell'avversario sia arbitraria, la condizione necessaria e sufficiente per la sicurezza diventa $Q^2(\Gamma)$ nel caso passivo (e nel caso attivo in presenza di una canale di broadcast) e $Q^3(\Gamma)$ nel caso attivo.

3.4 STRUTTURA DEI PROTOCOLLI MPC

Comunemente la struttura di un protocollo MPC è articolata in tre fasi: condivisione degli input, computazione della funzione con in ingresso i valori di input condivisi, rivelazione dell'output. Prima di poter proseguire con la trattazione è necessario soffermarsi ad analizzare una primitiva fondamentale che costituisce uno dei concetti base nel contesto della MPC: *la condivisione di segreti* [25].

3.4.1 *Condivisione di segreti*

I partecipanti alla computazione condividono i propri input utilizzando la prima fase di una primitiva in due fasi chiamata *condivisione di segreti a soglia (threshold secret sharing)* ed impiegano la seconda fase di tale primitiva per apprendere l'output della funzione.

La *fase di condivisione (sharing phase)* è la prima di una schema di condivisione di segreti a soglia di tipo-(k, n): lo schema (che verrà eseguito da ogni partecipante) prende come input un segreto s in qualche insieme S e restituisce n stringhe binarie $s_1, \ldots, s_n$, dette porzioni di s, per poi consegnare ogni stringa s_i al partecipante P_i. La seconda fase è detta *ricostruzione del segreto (reconstruction phase)* : ogni giocatore ritrasmette agli altri la propria porzione di segreto ed esegue la fase di ricostruzione al fine di recuperarlo.

Inoltre, lo schema dipenderà da un parametro di soglia k: richiederemo che la conoscenza di al più k porzioni di s non riveli nulla su s; d'altra parte richiederemo anche che la conoscenza di $k+1$ (o più) porzioni sia sempre sufficiente a ricostruire completamente s. Ciò implica che, dopo la fase di condivisione, fino a k partecipanti corrotti controllati da un avversario non potranno apprendere alcuna informazione riguardo al segreto s, mentre la presenza di almeno $k+1$ partecipanti onesti permetterà il recupero del segreto s (assumendo che non siano state date porzioni non corrette) nella fase di ricostruzione [25].

Formalmente:

Definizione 3.14 (*Condivisione di segreti a soglia (threshold secret sharing)*)**.** Uno schema di condivisione di segreti a soglia di tipo-(k, n) è un algoritmo randomizzato D, spesso detto distributore (dealer), che prende in ingresso un segreto $s \in S$ e restituisce n porzioni $s_1, \ldots, s_n$ tali che:

- *Segretezza.* Per ogni sottoinsieme $M \subset \{1, \ldots, n\}$ di dimensione al più k, la distribuzione di probabilità di $\{s_i | i \in M\}$ è indipendente da s.
- *Correttezza.* Per ogni sottoinsieme $L \subset \{1, \ldots, n\}$ di dimensione al più $k+1$, il valore s è determinato in maniera univoca dagli elementi $\{s_i | i \in$

L}, ovvero esiste un algoritmo polinomiale per ricostruire s dati i valori $\{s_i | i \in L\}$ [10].

Lo schema classico è lo schema di *condivisione di segreti di Shamir (Shamir secret sharing)*, introdotto da Shamir in [39]. Assumiamo che il segreto s appartenga a qualche campo finito di cardinalità maggiore di n, come ad esempio il campo degli interi modulo p (dove $p > n$ è un primo); dunque, $s \in S = \mathbb{Z}_p$ per qualche primo $p > n > k$. Consideriamo lo schema seguente: si estraggono dapprima i coefficienti $a_1, \ldots, a_k$ da $\mathbb{Z}_p$ e si pone $f(X) = s + a_1 \cdot X + a_2 \cdot X^2 + \cdots + a_k \cdot X^k$. In altre parole, f è un polinomio casuale di grado al più k, con coefficienti in $\mathbb{Z}_p$, tale che $f(0) = s$. Le porzioni di s sono definite da $s_i = f(i) \bmod p$, per ogni $i = 1, 2, \ldots, n$. [3]

L'idea di Shamir è basata sull'interpolazione di Lagrange:

Definizione 3.15 (*Polinomio interpolante di Lagrange*)**.** Data una funzione $f(x)$ e n punti $a_1, a_2 \ldots a_n$ per cui sono noti i valori $f(a_1), f(a_2) \ldots, f(a_n)$ si definisce il *polinomio interpolante di Lagrange* della funzione f il seguente polinomio [41]:

$$P(x) = \sum_{i=1}^{n} f(a_i) \cdot \prod_{j \neq i}^{n} \frac{x - a_j}{a_i - a_j}.$$

È adesso semplice osservare che $k+1$ porzioni sono sufficienti per ottenere $g(X)$ (il polinomio interpolante della funzione $f(X)$) e sarà perciò possibile calcolare $g(0) = s = f(0)$. Sia infatti Φ un qualsiasi sottoinsieme di $\{1, ..., n\}$ con $k+1$ elementi, allora i giocatori in N, corrispondenti agli indici in Φ, possiedono coppie $(i, s_i)_{i \in \Phi}$. Tali giocatori possono ricavare il polinomio:

$$g(X) = \sum_{i \in \Phi} s_i \cdot \prod_{\substack{j \in \Phi \\ j \neq i}} \frac{j - X}{j - i}. \tag{3.1}$$

Si noti che $g(i) = s_i = f(i)$, dunque i polinomi f e g sono al più di grado k e condividono $k+1$ punti. In particolare, dato che $g(0) = s = f(0)$, lo schema di Shamir soddisfa la proprietà di *correttezza*. Il segreto s può essere ricostruito come combinazione lineare delle porzioni $\{s_i\}_{i \in \Phi}$:

$$s = g(0) = \sum_{i \in \Phi} \alpha_i \cdot s_i \qquad \alpha_i = \prod_{\substack{j \in \Phi \\ j \neq i}} \frac{j}{j - i}. \tag{3.2}$$

I coefficienti $\{\alpha_i\}_{i \in \Phi}$ della combinazione lineare dipendono solo da Φ e non dal segreto s. È semplice osservare anche che qualunque insieme di k porzioni è

3 $\mathbb{F}_p$ spesso indicato con $\mathbb{Z}_p$ è il campo delle classi di resto modulo p ed ha esattamente p elementi [40].

indipendente da s (non dà alcuna informazione su s). Infatti, ogni polinomio f di grado al più k e con $f(0) = s$ definisce un insieme di porzioni, ogni insieme potenziale di porzioni R corrisponde ad un unico polinomio, di grado al più k, tale che $f_R(0) = s$ e $f_R(i) = s_i$ per $i \in \Phi$: esiste una corrispondenza uno a uno tra tali p^k polinomi e i possibili p^k insiemi di porzioni. Nello schema il polinomio f è scelto a caso tra questi p^k polinomi ed ogni insieme R di porzioni è ugualmente probabile. Ergo, per qualunque segreto s, il requisito di segretezza è soddisfatto [10].

È importante sottolineare che nello schema di Shamir non è possibile verificare se una porzione ricevuta è corretta oppure no; esistono però schemi di *condivisione di segreti verificabile (verifiable secret sharing)* in cui ciascun giocatore può verificare la correttezza della porzione che ottiene.

Protocollo 3.1 (*Condivisione di segreti di Shamir di tipo-*(n, k))**.** Il protocollo riceve in input un segreto $s \in S = \mathbb{Z}_p$, per qualche primo $p > n > k$

- *Condivisione del segreto.*
  1. Casualmente si scelgono i coefficienti $a_1, \ldots, a_k$ da $\mathbb{Z}_p$.
  2. Si pone $f(X) = s + a_1 \cdot X + a_2 \cdot X^2 + \cdots + a_k \cdot X^k$.
  3. Per ogni $i = 1, 2, \ldots, n$, calcoliamo le porzioni di segreto $s_i = f(i) \bmod p$.
- *Ricostruzione del segreto.* In tale fase le porzioni $\{s_i\}_{i \in \Phi}$ vengono utilizzate per ricostruire il segreto, Φ un qualsiasi sottoinsieme di $\{1, ..., n\}$ con $k+1$ elementi.
  1. $\alpha_i = \prod_{\substack{j \in \Phi \\ j \neq i}} \frac{j}{j-i}$.
  2. $s = \sum_{i \in \Phi} \alpha_i \cdot s_i$

3.4.2 *Protocollo MPC per un caso semplice*

Seguendo la trattazione proposta da Venturi in [10], andiamo a delineare la struttura di un protocollo MPC per un caso semplice del problema MPC.

Problema MPC semplice

Sia $\mathbb{F}$ un campo finito. Consideriamo il seguente problema MPC: dato $P = \{P_1, \ldots, P_n\}$ insieme di n giocatori, ogni giocatore P_i possiede un input segreto $x_i \in \mathbb{F}$ e tutti i giocatori vogliono calcolare una funzione $f(x_1, \ldots, x_n)$ sui loro input. Per calcolare tale funzione i giocatori si scambiano messaggi attraverso *canali sicuri* (sia per quanto concerne la confidenzialità che per quanto concerne l'integrità) e desiderano mantenere la segretezza dei propri input e la correttezza

dell'output. In tale contesto consideriamo unicamente il caso di *avversario a soglia onesto-ma-curioso*: quindi è in grado di corrompere al massimo $k < n$ partecipanti alla computazione e adotta un comportamento passivo, non modificando il comportamento dei giocatori corrotti (si limita ad apprendere i valori di input ed i successivi valori intermedi appresi dal giocatore corrotto). Per quanto riguarda i parametri di sicurezza, in questo contesto, ci limiteremo a ritenere un protocollo sicuro se soddisfa i seguenti requisiti:

- *Input privacy.* Ogni insieme Φ di al massimo k giocatori non apprende niente al di fuori dei valori $\{x_j\}_{j\in\Phi}$ e del valore di output della funzione
- *Correttezza della computazione.* Al termine del protocollo ciascun giocatore ottiene l'output $s = f(x_1, \ldots, x_n)$

Considerazioni sulla funzione f

La funzione da calcolare è completamente arbitraria; l'unico requisito che deve rispettare è che sia calcolabile attraverso un *circuito aritmetico* che gestisce elementi del campo $\mathbb{F}$.

Definizione 3.16 (*Circuito aritmetico*)**.** Un circuito aritmetico con n input è un grafo senza cicli in cui:

1. esiste un unico nodo con grado di uscita 0, l'output del circuito.
2. esistono n nodi con grado di ingresso 1, gli input del circuito.
3. ogni nodo interno ha grado di ingresso 2 e grado di uscita 1 ed è un nodo di moltiplicazione oppure un nodo di addizione.

Considerando un circuito aritmetico su $\mathbb{F}$ con n input, l'output del circuito sarà valutato naturalmente a partire dagli input e tutte le operazioni sono eseguite in $\mathbb{F}$.

Protocollo MPC semplice

Dopo aver modellato il problema MPC e la funzione da calcolare, possiamo finalmente definire un protocollo MPC per il calcolo di tale funzione: in sostanza un protocollo per calcolare in modo sicuro l'output di un circuito aritmetico. L'idea è quella di usare uno schema di condivisione di segreti di Shamir e due sotto-protocolli, uno per il calcolo in modo sicuro della somma di input segreti, l'altro per il calcolo in modo sicuro del prodotto di due input segreti.

Per poter definire i due sotto-protocolli sfrutteremo il seguente teorema.

Teorema 3.1 (Somme e prodotti di segreti)**.** *Siano* $s_1, s_2 \in \mathbb{F}$ *due segreti. Indichiamo con* $s_1^{(i)}, \ldots, s_n^{(i)}$ *le porzioni di* s_i*, con* $i \in \{1, 2\}$*, ottenute usando uno schema di*

Shamir di tipo-(n, k)*. I valori* $s_1^{(1)} + s_1^{(2)}, \ldots, s_n^{(1)} + s_n^{(2)}$ *sono le porzioni del segreto* $s^{(1+2)} = s_1 + s_2$ *relative allo stesso schema di condivisione. Allo stesso modo le porzioni* $s_1^{(1)} \cdot s_1^{(2)}, \ldots, s_n^{(1)} \cdot s_n^{(2)}$ *sono le porzioni del segreto* $s^{(1\cdot 2)} = s_1 \cdot s_2$ *relative ad uno schema di Shamir di tipo-*$(n, 2k)$.

Dunque, ciascun giocatore che disponga delle porzioni relative a due segreti x_1 e x_2, condivisi tramite uno schema di Shamir di tipo-$(n, 2k)$, potrà calcolare, senza interazioni con gli altri giocatori, le porzioni relative alla somma $x^{(1+2)} = x_1 + x_2$ utilizzando il seguente protocollo:

Protocollo 3.2 (*Protocollo di addizione dei valori* x_1 *e* x_2)**.** Ogni giocatore P_i, con $i \in \{1, \ldots, n\}$, possiede le porzioni $s_i^{(1)}$, $s_i^{(2)}$ relative agli input x_1 e x_2.

1. Ciascun giocatore calcola $s_i = s_i^{(1)} + s_i^{(2)}$.

Similmente, ciascun giocatore può calcolare le porzioni del prodotto $x_1 \cdot x_2$. Tali porzioni saranno relative ad uno schema di Shamir di tipo-$(n, 2k)$. Dunque è necessario modellare il relativo protocollo in modo tale che i giocatori, interagendo, possano calcolare le porzioni di $x_1 \cdot x_2$ relative ad uno schema di Shamir di tipo-(n, k).

Protocollo 3.3 (*Protocollo di moltiplicazione dei valori* x_1 *e* x_2)**.** Ogni giocatore P_i, con $i \in \{1, \ldots, n\}$, possiede le porzioni $s_i^{(1)}$, $s_i^{(2)}$ relative agli input x_1 e x_2.

1. Il giocatore P_i calcola $s_i = s_i^{(1)} \cdot s_i^{(2)}$. Dopodiché genera le porzioni relative ad s_i utilizzando uno schema di Shamir di tipo-(n, k) (protocollo 3.1). Tali porzioni saranno indicate con $\bar{s}_1^{(i)}, \ldots, \bar{s}_n^{(i)}$. Ogni giocatore P_j invia la porzione $\bar{s}_i^{(j)}$ al giocatore P_i.
2. Siano $\alpha_1, \ldots \alpha_n$ le costanti per la ricostruzione del segreto in uno schema di Shamir di tipo-$(n, 2k)$ definite in (3.2). Ogni giocatore P_i calcola

$$u_i = \sum_{j=1}^{n} \alpha_j \cdot \bar{s}_i^{(j)}.$$

È importante notare che $s_1, \ldots, s_n$ sono le porzioni relative al valore $x_1 \cdot x_2$ in uno schema di Shamir di tipo-$(n, 2k)$; dunque $x_1 \cdot x_2 = \sum_{j=1}^{n} \alpha_j \cdot s_j$. Le porzioni relative ad s_j in uno schema di Shamir di tipo-(n, k) sono invece $\bar{s}_1^{(j)}, \ldots, \bar{s}_n^{(j)}$, quindi i valori $u_1, \ldots, u_n$ sono a loro volta porzioni di $x_1 \cdot x_2$ nello stesso schema di tipo-(n, k). Per garantire la sicurezza sarà quindi necessario assumere che $n = 2k + 1$.

Possiamo finalmente mostrare la costruzione del protocollo MPC sicuro. I nodi del circuito saranno indicati con $G_1, \ldots, G_d$, e sarà assunta l'esistenza di

un lato da G_j a G_i se e solo se $i > j$. I primi n nodi sono etichettati con gli input del circuito.

Protocollo 3.4 (*Protocollo MPC per la computazione sicura di un circuito aritmetico*)**.** Ogni giocatore P_i, con $i \in \{1, \ldots, n\}$, possiede un input segreto $x_i \in \mathbb{F}$.

- *Condivisione degli input.* Il giocatore P_i condivide x_i usando uno schema di Shamir di tipo-(n, k) (protocollo 3.1). Siano $s_1^{(i)}, \ldots, s_n^{(i)}$ le porzioni risultanti. Ogni giocatore P_i invia $s_j^{(i)}$ al giocatore P_j.
- *Computazione.* Per $c = n + 1, \ldots, d$, si calcola l'output del nodo corrente G_c nel seguente modo:
  1. Supponiamo che i lati in ingresso al nodo G_c provengano dai nodi G_i, G_j per $i, j < c$ ed indichiamo con $s_1^{(i)}, \ldots, s_n^{(i)}$ ed $s_1^{(j)}, \ldots, s_n^{(j)}$ le porzioni dell'output dei nodi G_i e G_j.
  2. Se G_c è un nodo di addizione: ogni giocatore calcola localmente la porzione della somma degli output dei nodi G_i e G_j eseguendo il protocollo 3.2.
  3. Se G_c è un nodo di moltiplicazione: ogni giocatore calcola localmente la porzione del prodotto degli output dei nodi G_i e G_j utilizzano il protocollo 3.3.
- *Rivelazione dell'output.* Ogni giocatore P_i invia la porzione $s_i^{(d)}$ relativa al nodo G_d, nodo output del circuito, al giocatore P_1. P_1 eseguendo la fase di ricostruzione dello schema di Shamir di tipo-(n, k) (protocollo 3.1) ricostruisce l'output del circuito $s = f(x_1, \ldots, x_n)$ a partire da $k + 1$ porzioni ed invia il valore ottenuto a tutti gli altri giocatori.

Per la correttezza dei protocolli di addizione e moltiplicazione, alla fine della computazione i giocatori posseggono le porzioni relative all'output del circuito e dunque la proprietà di correttezza è rispettata. Inoltre, in ogni round, averemo al massimo k partecipanti corrotti, dunque un avversario avrà al massimo k porzioni dello schema di Shamir di tipo-(n, k) e la proprietà di segretezza è rispettata [10] [42].

4

MULTI-PARTY COMPUTATION MODELLATA SECONDO LA TEORIA DEI GIOCHI

Nella trattazione classica del problema MPC, come abbia visto nella sezione 3.1.2, i partecipanti alla computazione sono identificati come onesti oppure disonesti; tuttavia, nella realtà la distinzione non è così netta: anche se per alcune applicazioni può aver senso, per altre sarebbe meglio analizzare i partecipanti vedendoli come né totalmente onesti né totalmente disonesti, ma bensì come individui egoisti, razionali ed intelligenti che cercano unicamente di massimizzare i propri "guadagni". In sostanza possiamo dire che trattiamo con *partecipanti razionali*. Dunque, affrontiamo il problema MPC dal punto di vista della teoria dei giochi: i partecipanti alla computazione, adesso a tutti gli effetti partecipanti ad un gioco per computare una funzione in modo cooperativo, avranno determinate preferenze rispetto ai vari esiti della computazione/gioco e possiamo aspettarci che seguiranno il protocollo se e solo se ciò implicherà un incremento della propria utilità attesa [43].

Prima di poter proseguire nella trattazione è necessario introdurre alcuni concetti avanzati di teoria dei giochi che ci serviranno per poter ridurre il problema MPC ad un gioco strategico.

4.1 CONCETTI AVANZATI DI TEORIA DEI GIOCHI

4.1.1 *Equilibrio correlato*

Introduciamo un *concetto di soluzione* alternativo rispetto all'*equilibrio di Nash*: l'*equilibrio correlato* (seguiamo la breve trattazione proposta da Dodis, Halevi e Rabin in [44]).

L'equilibrio di Nash è un concetto naturale, relativamente semplice ed accattivante: infatti, permette ai giocatori di seguire le proprie strategie indipendentemente l'uno dall'altro. Risulta immediato chiedersi se sia possibile ottenere utilità attese più elevate nel caso in cui si permettano strategie correlate. Dato un gioco in forma normale $NFG = (N, A, u)$, con A_i l'insieme delle possibili azioni del giocatore i-esimo, un *profilo a strategie correlate* $s = (s_1, \ldots, s_n)$ può essere una distribuzione di probabilità arbitraria, non necessariamente una distribuzione prodotto, su $A_1 \times \ldots \times A_n$; al contrario, un profilo strategico che è equilibrio di Nash dovrà essere una distribuzione

prodotto $s(A_1 \times \ldots \times A_n) = s_1(A_1) \times \ldots \times s_n(A_n)$. Un gioco che consente l'esistenza di profili a strategie correlate sarà implementato rafforzando le impostazioni di base di giochi strategici (one-shot) con l'introduzione di una terza parte fidata (trusted third party), detta *mediatore* M. M selezionerà una tupla di azione $a = (a_1, \ldots, a_n)$ in accordo con la distribuzione congiunta di probabilità $s(A_1 \times \ldots \times A_n)$ ed in modo privato raccomanderà l'azione a_i al giocatore P_i (è molto importante che P_i non apprenda alcuna informazione al riguardo delle azioni raccomandate agli altri giocatori). Assumeremo che, dopo aver ricevuto l'azione raccomandata a_i, il giocatore P_i conoscerà anche la distribuzione condizionata dei profili strategici raccomandati agli altri P_{-i} giocatori dato a_i: indichiamo tale distribuzione con $(s_{-i}|a_i)$. A questo punto il giocatore P_i potrà decidere se seguire la raccomandazione a_i sia la miglior scelta possibile assumendo che i rimanenti giocatori seguiranno $s_{-i}|a_i$. In una situazione di *equilibrio correlato* sarà nell'interessa di tutti i giocatori seguire le raccomandazioni del mediatore.

Definizione 4.1 (*Gioco mediato*)**.** Dato un mediatore M, la versione mediata di un gioco in forma normale $NFG = (N, A, u)$ è un gioco che consiste di due fasi:

1. Il mediatore M selezionerà un vettore di azione $a = (a_1, \ldots, a_n) \in A_1 \times \ldots \times A_n$ in accordo con la distribuzione congiunta di probabilità $s(A_1 \times \ldots \times A_n)$. M raccomanda privatamente ad ogni P_i l'azione a_i.

2. I partecipanti giocano il gioco NFG decidendo se seguire oppure no le raccomandazioni del mediatore [13].

Definizione 4.2 (*Equilibrio correlato*)**.** Dato un gioco in forma normale mediato $NFG = (N, A, u)$, un *equilibrio correlato (CE, correlated equilibrium)* è un profilo strategico $s^* = s^*(A_1 \times \ldots \times A_n) = (s_1^*, \ldots, s_n^*)$ tale che, per ogni giocatore $P_i \in N$, per ogni $a_i \in A_i$ e per ogni a_i^* nel supporto di s_i^*, abbiamo che $u_i(a_i^*, (s_{-i}^*|a_i^*)) \geqslant u_i(a_i, (s_{-i}^*|a_i^*))$.

Sostanzialmente $u_i(a_i^*, (s_{-i}^*|a_i^*))$ rappresenta l'utilità attesa $u_i(a_i^*, a_{-i})$, dove a_{-i}^* è il vettore delle azioni eseguite dagli altri giocatori in accordo con la distribuzione $(s_{-i}^*|a_i^*)$. In altre parole, $u_i(a_i^*, (s_{-i}^*|a_i^*))$ ci dà l'utilità attesa nel caso in cui il giocatore P_i giochi l'azione a_i^* raccomandata dal mediatore, mentre $u_i(a_i, (s_{-i}^*|a_i^*))$ quella nel caso in cui decida di giocare a_i al posto di a_i^* [44] [25].

Possiamo dare una definizione di equilibrio correlato equivalente a quella precedente, ma che, come vedremo in seguito, si adatterà meglio al caso dei giochi con coalizioni e giochi con partecipanti limitati computazionalmente. Consideriamo la seguente notazione: $\Delta(A)$ è l'insieme di tutte le possibili distribuzioni di probabilità su A; $\mathcal{M} \in \Delta(A)$ è la distribuzione di probabilità nota secondo cui il mediatore M seleziona il vettore di azioni $a^* = (a_1^*, \ldots, a_n^*)$ da

		Automobilista$_2$	
		Fermati	Continua dritto
Automobilista$_1$	Fermati	-1, -1	0, 1
	Continua dritto	1, 0	-100, -100

Tabella 7: *Gioco del semaforo* (Esempio 4.1)

raccomandare ai giocatori; $u_i(\mathcal{M})$ l'utilità attesa per il giocatore P_i nel caso in cui tutti i giocatori seguano le azioni raccomandate da M. Una definizione alternativa di equilibrio correlato è la seguente:

Definizione 4.3 (*Equilibrio correlato-definizione alternativa*)**.** Dato un gioco in forma normale mediato $NFG = (N, A, u)$, una distribuzione $\mathcal{M} \in \Delta(A)$ è un *equilibrio correlato* se, per ogni P_i e per una qualunque funzione di deviazione $r_i \colon A_i \to A_i$, vale che $u_i(\mathcal{M}) \geqslant u_i(r_i(a_i^*), a_{-i}^*)$ (dove $a^* = (a_1^*, \ldots, a_n^*) = (a_i^*, a_{-i}^*)$ è un campione di azioni selezionato in accordo con $\mathcal{M}$).

Tale definizione alternativa, proposta da Katz in [46], come vedremo in seguito, risulterà più adatta all'estensione del concetto di equilibrio correlato nel caso giochi che ammettono coalizioni.

Esempio 4.1 (*Gioco del semaforo*)**.** Due automobili si incontrano ad un incrocio non regolato da semafori. Ogni automobile deve decidere se passare o non passare. Se passano entrambe probabilmente ci sarà un incidente, e questo è indicato con un costo negativo -100. Se una passa e l'altra aspetta, la prima guadagna 1 mentre l'altra ottiene 0. Infine, se nessuna passa, entrambe ottengono un guadagno -1. Il gioco è rappresentato nella tabella 7 . È semplice vedere che esistono due equilibri di Nash a strategie pure: $s^1 = (F, C)$ e $s^2 = (C, F)$ con utilità attesa pari ad 1 (indichiamo C l'azione "Continua dritto" e con F l'azione "Fermati"). Esiste inoltre un equilibrio di Nash a strategie miste: $s^3 = (\frac{1}{2} \cdot F + \frac{1}{2} \cdot C, \frac{1}{2} \cdot C + \frac{1}{2} \cdot F)$; in questo caso il risultato non desiderato (C, C) accadrà con probabilità pari ad $\frac{1}{4}$ nella distribuzione prodotto, dunque avremo utilità attesa negativa. Introdotto un mediatore, ad esempio un semaforo, possiamo mostrare in modo intuitivo che il profilo strategico $s^* = (\frac{1}{3} \cdot (F, C) + \frac{1}{3} \cdot (C, F) + \frac{1}{3} \cdot (F, F))$ è un *equilibrio correlato*. Se al primo automobilista è stata raccomandata l'azione F, la sua utilità attesa sarà $\frac{1}{2} \cdot (-1) + \frac{1}{2} \cdot 1 = 0$, dato che la probabilità, condizionata da $a_1^* = F$, che il secondo automobilista abbia ricevuto come raccomandazione $a_2^* = F$ (oppure $a_2^* = C$) è pari ad $\frac{1}{2}$; data tale raccomandazione il giocatore non è incentivato a non seguirla, infatti, se decidesse di cambiare la propria azione in C, la sua nuova utilità attesa sarebbe $\frac{1}{2} \cdot (1) + \frac{1}{2} \cdot (-100) = -49.5$. Allo stesso modo, se al primo automobilista fosse stata raccomandata l'azione C, saprebbe per certo che al secondo è stata consigliata l'azione F dato che la coppia (C, C) non verrà mai raccomandata in accordo con il profilo s^*; in questo caso il primo non trarrebbe alcun beneficio dal cambiare la propria azione in F [44] [45].

È importante sottolineare tre aspetti di questo equilibrio: (a) per ogni equilibrio di Nash esiste un corrispondente equilibrio correlato; (b) per ogni gioco strategico esiste sempre un equilibrio correlato; (c) i payoff ottenuti in una situazione di equilibrio correlato possono essere sia migliori sia peggiori di quelli ottenuti in un equilibrio di Nash; ad ogni modo, è possibile implementare il mediatore in modo che i payoff dell'equilibrio correlato siano migliori, per la maggior parte o per tutti i giocatori, rispetto ai payoff che si otterrebbero in un equilibrio di Nash [44] [25].

4.1.2 *Giochi ad informazione incompleta*

I giochi che abbiamo analizzato fino a questo momento erano *giochi ad informazione completa*, in cui ogni giocatore conosceva le proprie strategie, gli avversari, le loro strategie ed i loro payoff. Nella realtà, spesso, ci troviamo ad affrontare situazioni nelle quali alcuni giocatori non possiedono una conoscenza totale delle caratteristiche degli altri partecipanti: in questo caso parleremo di *giochi ad informazione incompleta* [45].

Nei giochi ad informazione incompleta ogni partecipante possiede un *tipo* privato $t_i \in T_i$. In generale il tipo di un giocatore è una variabile che incapsula tutte le informazioni non pubbliche possedute dal giocatore: ad esempio, se un giocatore possiede un input segreto, potremmo vedere tale input come il suo tipo. Il vettore dei tipi $t = (t_1, \ldots, t_n)$ viene selezionato secondo una distribuzione di probabilità nota pubblicamente; in particolare ogni tipo t_i influenzerà la funzione utilità del giocatore i-esimo: precisamente la funzione u_i dipenderà non solo dal vettore di azioni $(a_1, \ldots, a_n)$ ma anche anche dal tipo privato t_i o, in alcuni casi, dall'intero vettore di tipi t. Dunque, la nozione di tipo ci consente una via per definire in modo diretto l'incertezza sulla funzione utilità di un gioco [25] [2] [46].

Definizione 4.4 (*Gioco ad informazione incompleta*)**.** Un *gioco ad informazione incompleta* è costituito da una tupla (N, A, T, p, u), dove:

- N è l'insieme degli n giocatori; $P_i \in N$.
- $A = A_1 \times \ldots \times A_n$, dove A_i è l'insieme delle possibili azioni di P_i.
- $T = T_1 \times \ldots \times T_n$, dove T_i è lo spazio dei tipi del giocatore P_i.
- $p \colon T \to [0, 1]$ è una distribuzione di probabilità sui vettori di tipi $(t_1, \ldots, t_n)$.
- $u = (u_1, \ldots, u_n)$, dove $u_i \colon A \times T \to \mathbb{R}$ è la funzione utilità per P_i.

Dunque, seguendo la definizione proposta da Katz in [46], un gioco ad informazione incompleta è giocato nel seguente modo: prima, il vettore $(t_1, \ldots, t_n)$

è scelto in accordo con la distribuzione di probabilità p ed il tipo t_i viene assegnato ad ogni giocatore P_i; dopodiché, ogni giocatore P_i giocherà un'azione $a_i \in A_i$ (si assume che vengano giocate in modo simultaneo) e, infine, riceverà il proprio payoff $u_i(t_1, \ldots, t_n, a_1, \ldots, a_n)$ (consideriamo il caso in cui ogni tipo è variabile della funzione utilità).

In questo contesto generalizzare la nozione di equilibrio di Nash risulta abbastanza immediato: è un profilo strategico $s = (s_1, \ldots, s_n)$ per cui s_i fa parte dell'insieme delle migliori risposte al profilo strategico s_{-i} adottato dagli altri giocatori (per un approfondimento vedere [2]). La strategia del giocatore P_i corrisponde ad una funzione che mappa l'input/tipo t_i in un'azione $a_i \in A_i$; una strategia pura corrisponde ad una funzione deterministica. È importante notare che dobbiamo considerare l'utilità attesa anche quando consideriamo equilibri di Nash a strategie pure, dato che l'utilità di P_i dipende anche dai tipi degli altri giocatori e questi sono sconosciuti a P_i nel momento in cui seleziona un'azione [46].

Cerchiamo adesso di generalizzare la nozione di *gioco mediato* per i giochi ad informazione incompleta.

Definizione 4.5 (*Giochi mediato ad informazione incompleta*)**.** Dato un mediatore M, la versione mediata di un gioco ad informazione incompleta (N, A, T, p, u) è un gioco che consiste di quattro fasi:

1. Il vettore $(t_1, \ldots, t_n)$ è scelto in accordo con la distribuzione di probabilità p. Ad ogni giocatore P_i viene assegnato il tipo t_i.
2. Ogni P_i invia un tipo t'_i al mediatore M.
3. In base al vettore di tipi $t' = (t'_1, \ldots, t'_n)$ ricevuto, il mediatore seleziona un profilo strategico correlato $s = (s_1, \ldots, s_n)$, seleziona un vettore di azioni $a = (a_1, \ldots, a_n)$ tale che ogni a_i sia nel supporto di s_i ed infine raccomanda ad ogni giocatore P_i l'azione a_i.
4. Ogni giocatore partecipa normalmente al gioco ad informazione incompleta, decidendo se seguire oppure no la raccomandazione del mediatore.

Osserviamo che la strategia canonica prevista per ogni giocatore P_i è quella di riferire onestamente il proprio tipo t_i al mediatore M (dunque $t'_i = t_i$), e quindi seguire l'azione consigliata a_i. Nonostante ciò, al giocatore vengono lasciate due libertà:

- Inviare un tipo sbagliato $t'_i \neq t_i$, oppure non inviare alcun tipo, al mediatore M.
- Decidere di non seguire la raccomandazione a_i e giocare un azione $a'_i \neq a_i$.

Dato che M può ricevere dei tipi "non corretti", per avere una strategia di campionamento totalmente definita per il mediatore è necessario specificare la distribuzione di probabilità congiunta x per ogni vettore di tipi $t = (t_1, \ldots, t_n)$, anche al di fuori del supporto della distribuzione congiunta dei tipi: formalmente x^t dovrebbe essere definita per ogni $t \in \times_i (T_i \cup \{\perp\})$ ($t_i = \perp$ nel caso in cui un giocatore non fornisca il proprio tipo). È utile sottolineare che è possibile modellare anche il rifiuto di partecipare al gioco introducendo un'azione speciale $\perp$ e definendo in modo esplicito i valori di utilità nel caso in cui un giocatore decida di optare per tale azione: in molti giochi è possibile garantire la partecipazione assegnando payoff molto bassi nel caso in cui si decida di non partecipare. In generale, assumeremo che i giocatori partecipino sempre al gioco scegliendo un'azione $a_i \in A_i$ e ricevendo in seguito un appropriato payoff.

In queste situazione la definizione di equilibrio correlato risulta immutata: è necessario unicamente sottolineare che adesso la strategia di un giocatore P_i determina sia quale valore t_i' viene inviato al mediatore, sia quale azione verrà giocata [46] [25].

4.2 GIOCO DI VALUTAZIONE DI FUNZIONE

Dopo aver introdotto i concetti relativi all'equilibrio correlato ed ai giochi ad informazione incompleta, siamo finalmente pronti per iniziare la trattazione relativa alla riduzione del problema MPC in un gioco strategico. Seguiamo le linee guida tracciate da Dodis e Rabin in *"Cryptography and Game Theory"* [25].

4.2.1 *Definizione di gioco di valutazione di funzione*

Per poter rendere "razionale" il calcolo sicuro e cooperativo di una funzione f, per prima cosa, è necessario definire un appropriato *gioco di valutazione di funzione (function evaluation game)*: n giocatori, ognuno con un input segreto t_i, cercano stimare il valore di una funzione $f(t_1, \ldots, t_n)$ (ci concentriamo unicamente su funzione con output singolo, anche se non risulta difficile generalizzare la trattazione nel caso di funzioni multi-output). Tale gioco è ad *informazione incompleta*: infatti, essendo gli input privati, i giocatori non hanno conoscenza totale riguardo le caratteristiche degli altri partecipanti. Modelliamo perciò il gioco seguendo la definizione 4.4 : i tipi $t_{i=1,\ldots,n}$ associati ai giocatori rappresentano gli input privati (saranno scelti nella prima fase del gioco, in accordo con la distribuzione di probabilità p, ed assegnati ad ogni giocatore P_i); l'azione selezionata da ogni P_i sarà una stima riguardo all'output o^* di f, dunque $a_i = o_i$ [25].

Uno dei punti chiave è la definizione delle funzioni utilità per i partecipanti al gioco, ma prima di poter definire in modo esplicito le funzioni utilità è

necessario riflettere su alcuni aspetti crittografici legati al problema MPC. È importante ricordare che, anche avendo a disposizione un protocollo MPC sicuro, non c'è modo di costringere le parti a partecipare al protocollo: come abbiamo visto nella sezione 3.2.2, è impossibile impedire ad un partecipante corrotto di non trasmettere alcun valore di input. Per ovviare a questo problema (o, in generale, a qualunque tipo di deviazione dal protocollo) definiremo i *payoff* dei giocatori in modo tale che sia nel loro interesse partecipare al gioco/protocollo nel caso in cui questo venga eseguito correttamente [43]. Avendo n giocatori $P_1, \ldots, P_n$, seguendo la trattazione proposta da McGrew, Porter e Shoham nell'articolo *"Towards a general theory of non-cooperative computation"*, la funzione utilità che descrive le preferenze di ogni P_i dipenderà dalle seguenti considerazioni crittografiche:

- *Correttezza.* Ogni P_i desidera stimare/calcolare correttamente la funzione f.
- *Esclusività.* Ogni P_i desidera che gli altri giocatori non stimino/calcolino la funzione correttamente.
- *Segretezza.* Ogni P_i desidera che gli altri giocatori non scoprano niente al riguardo del proprio input segreto t_i.
- *Voyeurismo.* Ogni P_i desidera scoprire quanto più possibile riguardo gli input degli altri giocatori.

Detto ciò, possiamo finalmente definire le funzioni utilità per il gioco di valutazione di funzione. Per semplicità, nel definire tali funzioni utilità, prendiamo in considerazione unicamente le proprietà di correttezza ed esclusività, assegnando un maggior valore alla correttezza. È importante ricordare che la funzione u_i dipenderà non solo dal vettore di azioni $(o_1, \ldots, o_n)$, ma anche dall'intero vettore di tipi $t = (t_1, \ldots, t_n)$: t rappresenta il vettore degli input segreti dei giocatori, dunque determina il corretto valore della funzione $o^* = f(t)$ (assumiamo che ogni input t_i sia necessario per il calcolo di f); pertanto $u_i = (o_1, \ldots, o_n, t_1, \ldots, t_n)$. Semplifichiamo la situazione sottolineando che le azioni $(o_1, \ldots, o_n)$ dei giocatori determinano il seguente vettore di valori booleani: $\mathtt{correct} = (\mathtt{correct}_1, \ldots, \mathtt{correct}_n)$, dove $\mathtt{correct}_i = 1$ se e solo se $o_i = o^*$ (sostanzialmente ci indica quale dei giocatori ha stimato il valore della funzione in modo corretto). Ergo, possiamo dire che la funzione utilità di ogni giocatore dipenderà unicamente dal vettore di valori booleani $\mathtt{correct}$; scriveremo $u_i(\mathtt{corret})$ per indicare l'utilità del giocatore i-esimo. Anziché assegnare dei valori arbitrari ai payoff per catturare la correttezza e l'esclusività, descriviamo unicamente i vincoli minimi che implicano queste proprietà:

- *Vincolo di correttezza.* Ogni volta in cui $\mathtt{correct}_i = 1$ e $\mathtt{correct}'_i = 0$, $u_i(\mathtt{corret}) > u_i(\mathtt{corret}')$.

- *Vincolo di esclusività.* Se $\text{corret}_i = \text{corret}'_i$, per ogni $j \neq i$ abbiamo che $\text{corret}_j \leqslant \text{corret}'_j$ fintanto che $\text{corret}_j = 0$ e $\text{corret}'_j = 1$ per qualche j, allora $u_i(\text{corret}) > u_i(\text{corret}')$ (se P_i apprende l'output, preferisce che il numero degli altri partecipanti che lo apprendono sia il più piccolo possibile) [47] [25].

Definizione 4.6 (*Gioco di valutazione di funzione*)**.** Un gioco di valutazione di funzione è un *gioco ad informazione incompleta* (N, A, T, p, u), dove:

- $N = \{P_1, \ldots, P_n\}$ è l'insieme dei giocatori che desiderano stimare correttamente una funzione $f(t_1, \ldots, t_n)$.
- Le possibile azioni di ogni P_i sono le possibili stime riguardo all'output o^* di f.
- Lo spazio dei tipi T_i rappresenta lo spazio dei possibili input del giocatore P_i.
- p è una distribuzione di probabilità sui vettori di input $(t_1, \ldots, t_n)$.
- La funzione utilità $u_i(\text{correct})$ per ogni $P_i \in N$ rispetta il *vincolo di correttezza* ed il *vincolo di esclusività*.

Possiamo adesso chiederci quali siano gli equilibri del gioco appena definito. In questa particolare situazione, gli equilibri di Nash non risultano molto interessanti, dato che i partecipanti possiedono troppe poche informazioni per riuscire a stimare correttamente il valore della funzione f: la probabilità di successo è trascurabile. D'altro canto, risulta molto interessante studiarne gli equilibri correlati; quindi, seguendo la definizione 4.5, caratterizziamo la versione mediata del gioco di valutazione di funzione:

Definizione 4.7 (*Gioco mediato di valutazione di funzione*)**.** Dato un mediatore M in grado di calcolare il valore di una funzione, la versione mediata del *gioco di valutazione di funzione* è un gioco che consiste di quattro fasi:

1. Il vettore degli input privati $(t_1, \ldots, t_n)$ è scelto in accordo con la distribuzione di probabilità p. Ad ogni giocatore P_i viene assegnato l'input t_i.
2. Ogni P_i invia un tipo/input t'_i al mediatore M.
3. Dato che ogni giocatore vuole stimare/calcolare correttamente il valore della funzione f, risulta naturale considerare la seguente strategia per il mediatore: M calcola il valore di f utilizzando il vettore di tipi/input che ha ricevuto dai giocatori $t' = (t'_1, \ldots, t'_n)$ e raccomanda ad ogni P_i l'azione $o' = f(t')$ (nel caso in cui qualche giocatore non invia alcun tipo $t'_i = \perp$, assumiamo che M invii un messaggio di errore e lasci decidere ai giocatori in modo indipendente l'azione da selezionare).

4. Ogni giocatore partecipa normalmente al gioco di valutazione di funzione, decidendo se seguire oppure no la raccomandazione del mediatore.

Ci chiediamo adesso per quale classe di funzioni la strategia adottata del mediatore (calcolare il valore della funzione utilizzando gli input ricevuti $t' = (t'_1, \ldots, t'_n)$ e raccomandare ad ogni giocatore l'azione $o' = f(t')$), d'ora in poi detta *strategia canonica*, rappresenti un equilibrio correlato: formalmente, ci chiediamo per quale classi di funzioni il profilo a strategie pure suggerito dal mediatore $(o'_1, \ldots, o'_n)$ è un equilibrio correlato per il gioco di valutazione di funzione [25].

4.2.2 *Funzioni NCC*

Come abbiamo visto nella definizione 4.5, una delle libertà lasciata ai giocatori è quella di inviare un tipo/input sbagliato $t'_i = t^f_i \neq t_i$, oppure di non inviare alcun tipo $t'_i = \perp$, al mediatore M. Si può dunque facilmente osservare che il profilo strategico suggerito dal mediatore rappresenta un equilibrio solamente per giochi di valutazione di funzione in cui è interesse dei partecipanti inviare i propri input reali al mediatore.

Cerchiamo di mostrare quanto appena detto in modo più formale. Considerando che $f\colon T_1 \times \ldots \times T_n \to O$ è la funzione che i giocatori cercano di stimare, con O codominio di tale funzione, per ogni giocatore P_i definiamo le seguenti funzioni:

- *Funzione di dichiarazione.* La funzione $m_i\colon T_i \to T_i$, detta *funzione di dichiarazione*, determina, in funzione dell'input reale t_i, quale input il giocatore P_i dichiarerà al mediatore. Ci interessa particolarmente la *funzione di dichiarazione sincera*, vale a dire la funzione identità $m^t_i(t_i) = t_i$.
- *Funzione di stima.* La funzione $e_i\colon T_i \to O$, detta *funzione di stima*, è utilizzata da P_i per stimare il valore di f in base al proprio valore di input, dunque $e_i(t_i) = o^e_i$.
- *Funzione di interpretazione.* La funzione $r_i\colon O^2 \to O$, detta *funzione di interpretazione*, è utilizzata dal giocatore P_i per decidere, in funzione del proprio input segreto e del valore raccomandato dal mediatore $f(t'_1, \ldots, t'_n) = o'$, con quale valore stimare la funzione f. Ci interesserà particolarmente la *funzione di interpretazione fiduciosa*, cioè $r^t_i(e_i(t_i), o') = r^t_i(o^e_i, o') = o'$, nella quale P_i semplicemente dichiara come stima il valore suggeritogli dal mediatore.

Quindi, possiamo identificare la strategia pura per un giocatore $P_{i=1,\ldots,n}$ con la coppia di funzioni (m_i, r_i). Chiameremo *strategia onesta* la strategia (m^t_i, r^t_i). Si può notare facilmente che un profilo strategico costituito unicamente da

strategie oneste permetterà ad ogni giocatore di stimare la funzione f in modo corretto per qualunque vettore di input. Siamo dunque interessati a funzioni per le quali tale profilo strategico forma un equilibrio: quindi, richiedere che il profilo strategico suggerito dal mediatore rappresenti un equilibrio per giochi di valutazione di funzione equivale a richiedere che in tali giochi sia interesse dei partecipanti inviare i propri input reali al mediatore [49].

Il problema che stiamo affrontando ricade in quell'area della teoria dei giochi che viene chiamata *progettazione di meccanismi (mechanism design)*.Formalmente consideriamo un *meccanismo* come una coppia (Ω, σ), dove Ω è un gioco e σ un profilo strategico. In modo intuitivo, un *progettista di meccanismi (mechanism designer)* si occupa di progettare il gioco Ω e raccomandare ad ogni giocatore P_i di seguire la strategia $\sigma_i \in \sigma$; l'aspettativa è che un risultato "buono" si otterrà se tutti i giocatori seguiranno la strategia raccomandata nel gioco. La progettazione di un meccanismo sostanzialmente equivale alla progettazione di un protocollo; la strategia raccomandata è il protocollo, e il gioco è definito da tutte le possibili deviazioni dal protocollo [43] [49]. Quindi, nel nostro caso, ci stiamo chiedendo per quali funzioni esiste un meccanismo (Ω, σ), dove Ω è *un gioco mediato di valutazione di funzione* e $\sigma = (m_1^t, r_1^t), \ldots, (m_n^t, r_n^t)$ profilo strategico costituito unicamente da *strategie oneste*, tale che σ è equilibrio correlato per Ω [47].

È interessante notare che, come avevamo sottolineato in un'osservazione relativa alla definizione 3.4, anche in presenza di un protocollo MPC π che *calcola in modo sicuro* una funzione f, un partecipante corrotto potrà sempre mentire sul proprio valore di input. Nonostante non esista un modo per costringere i partecipanti ad usare i propri input reali, modellando il problema MPC tramite un gioco strategico, potremmo domandarci se sia possibile incentivare i giocatori a non mentire [43] [49].

Dunque, tornando al *gioco mediato di valutazione di funzione*, possiamo chiederci come sia possibile motivare i giocatori ad inviare gli input corretti al mediatore, in modo che quest'ultimo calcoli correttamente il valore della funzione f. Per prima cosa, possiamo interrogarci sul motivo per il quale un giocatore che vuole computare una funzione in modo corretto (il *vincolo di correttezza* possiede un peso maggiore rispetto al *vincolo di esclusività*) dovrebbe mentire al riguardo del proprio input segreto: cerchiamo di comprenderlo tramite un esempio.

Esempio 4.2 (*Gioco di valutazione della funzione parità*)**.** Consideriamo un *gioco mediato di valutazione di funzione* in cui i partecipanti desiderano stimare corret-

tamente il valore della *funzione parità* $f\colon \{0,1\}^n \to \{0,1\}$ definita del seguente modo:

$$f(t) = \begin{cases} 1, & \text{se e solo se il numero di 1 nel vettore } t \in \{0,1\}^n \text{ è dispari,} \\ 0, & \text{altrimenti.} \end{cases}$$

In altre parole, è la funzione che esegue lo XOR di tutti gli input dei partecipanti: $f(t) = t_1 \oplus t_2 \oplus \cdots \oplus t_n$ [50]. Assumendo che tutti gli altri giocatori inviino il proprio input corretto, il giocatore P_i sarà incentivato ad inviare l'input $\neg t_i$: in tal modo, il mediatore computerà un valore non corretto o^f e lo raccomanderà ad ogni partecipante. A questo punto, il giocatore P_i potrà semplicemente stimare che il valore corretto della funzione sarà $\neg o^f$, ottenendo così la massima utilità dato che sarà stato l'unico ad aver ottenuto il valore effettivo della funzione.

Possiamo dunque dire che la *funzione parità* non incentiva i partecipanti a dichiarare al mediatore i propri input reali. Alternativamente, consideriamo la *funzione maggioranza* $f\colon \{0,1\}^n \to \{0,1\}$ definita come segue [4]:

$$f(t) = \begin{cases} 1, & \text{se e solo se la maggior parte degli elementi in } t \in \{0,1\}^n \text{ sono 1,} \\ 0, & \text{altrimenti.} \end{cases}$$

In questo caso, non è difficile osservare che, se un giocatore mentisse sul proprio input, non sarebbe in grado di ricostruire il corretto valore della funzione in modo autonomo. Ergo, in modo intuitivo, possiamo dire che una funzione incentiva i partecipanti ad inviare al mediatore i propri input reali se nessun giocatore è in grado di ottenere autonomamente il valore di tale funzione nel caso in cui invii un input non corretto: tale classe di funzione, introdotta per la prima volta da Shoham e Tennenholtz in [48] e trattata ampiamente in [49], sarà detta classe delle funzioni *computabili in modo non-cooperativo (non-cooperatively computable)* o, più semplicemente, funzioni NCC.

Anche se non abbiamo ancora definito in modo formale quali funzioni facciano parte della categoria delle funzioni NCC, abbiamo una prima risposta alla domanda che ci eravamo posti alla fine della sottosezione precedente, cioè per quale classi di funzioni il profilo a strategie pure suggerito dal mediatore $(o'_1, \ldots, o'_n)$ fosse un equilibrio correlato: tale profilo è un equilibrio solamente per giochi di valutazione di funzione in cui è interesse dei partecipanti inviare i propri input reali al mediatore, dunque la classe di funzioni che stavamo cercando è la classe delle funzioni NCC.

Definizione 4.8 (*Funzioni computabili in modo non-cooperativo (NCC)*)**.** Dato un *gioco mediato di valutazione di funzione* f, diremo che f *è computabile in modo non-cooperativo (NCC)* se la *strategia canonica* del mediatore è un equilibrio correlato

4 Formalmente $f(t) = \lfloor \frac{1}{2} + \frac{(\sum_{i=1}^{n} t_i) - 1/2}{n} \rfloor$ [51]

per il gioco di valutazione di funzione f (in sostanza, se è nell'interesse dei giocatori riportare al mediatore i propri input in modo onesto) [25].

Alternativamente, considerando le strategie pure di ogni giocatore P_i tramite la coppia di funzioni (m_i^t, r_i^t), possiamo dare la seguente definizione di funzione NCC [49]:

Definizione 4.9 (*Funzioni computabili in modo non-cooperativo (NCC)-definizione alternativa*)**.** Dato un *gioco mediato di valutazione di funzione* f, diremo che f *è computabile in modo non-cooperativo (NCC)* se, per ogni giocatore P_i, per ogni strategia (m_i^t, r_i^t) e per ogni $t_i \in T_i$, vale la seguente proprietà:

- Esiste un $t_{-i} = (t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n)$ per cui $r_i(e(t_i), f(m_i(t_i), t_{-i})) = r_i(o_i^e, o') \neq o* = f(t_i, t_{-i})$; cioè, esiste un giocatore che ha mentito sul proprio input e nessuno è in grado di ricostruire il corretto valore della funzione in modo autonomo.
- Oppure, per ogni t_{-i}, abbiamo che $f(m_i(t_i), t_{-i}) = f(t_i, t_{-i})$; cioè, ogni giocatore ha fornito al mediatore il proprio input segreto in modo corretto.

Quindi, dato un gioco mediato di valutazione di funzione, se la funzione da valutare non è NCC, non sarà possibile sperare che i partecipanti seguano le raccomandazioni del mediatore, dunque il protocollo non verrà seguito. Questa semplice osservazione rafforza le motivazioni dello studio del problema MPC sotto forma di gioco strategico: per provare che i partecipanti seguiranno effettivamente un protocollo MPC sicuro sarà necessaria una definizione basata sulla teoria dei giochi del problema MPC [47].

Seguendo il lavoro di McGrew, Porter e Shoham [47], illustriamo due classi di funzioni che sono *non-NCC*.

Definizione 4.10 (*Funzione dominata*)**.** Data una funzione $f\colon T^n \to O$, diremo che f è *dominata* se esiste $t_i \in T$ tale che il valore di f è determinato indipendentemente dagli altri input t_{-i}. Formalmente: $\forall t_{-i} t'_{-i},\ f(t_i, t_{-i}) = f(t_i, t'_{-i})$.

Nel caso in cui f sia dominata, il giocatore P_i che possiede l'input segreto che consente di determinare il valore della funzione in modo indipendente non sarà incentivato ad inviarlo al mediatore, ma potrà calcolare autonomamente il valore della funzione: dunque $corret_i = 1$ anche senza l'aiuto di M.

Definizione 4.11 (*Funzione reversibile*)**.** Data una funzione $f\colon T^n \to O$, diremo che f è *reversibile* se, per qualche input $t_i \in T$, esiste un altro input $t'_i \in T$ ed una funzione g tali che:

- (a) $\forall t_{-i}$ abbiamo che $g(f(t'_i, t_{-i}), t_i) = f(t_i, t_{-i})$.
- (b) per qualche t_{-i} abbiamo che $f(t'_i, t_{-i}) \neq f(t_i, t_{-i})$.

Nel caso in cui f sia reversibile: per la proprietà (a), il giocatore P_i, riportando al mediatore t_i' al posto di t_i, non correrà alcun rischio in termini di correttezza; per la proprietà (b), P_i impedirà, almeno qualche volta, agli altri giocatori di stimare in modo corretto il valore della funzione, dunque, per il *vincolo di esclusività*, riceverà un payoff maggiore. La *funzione di parità* dell'esempio 4.2 è una *funzione reversibile* [25].

Sotto l'assunzione che un giocatore consideri troppo grande il rischio di non stimare correttamente il valore della funzione per essere tentato da maggiore esclusività, si può dimostrare che queste due classi sono le uniche funzioni *non-NCC*. In modo formale, dati i vettori di valori booleani $\mathtt{correct}$, $\mathtt{correct}'$ e $\mathtt{correct}''$, tali che $\mathtt{correct}_i = \mathtt{correct}_i' = 1$ e $\mathtt{correct}_i'' = 0$, abbiamo che $u_i(\mathtt{correct}) > \epsilon \cdot u_i(\mathtt{correct}'') + (1-\epsilon) \cdot u_i(\mathtt{correct}')$, dove ϵ è la più piccola probabilità in p (distribuzione di probabilità sui vettori di input); cioè, se esiste anche un minima probabilità che P_i, deviando dalla strategia onesta, non apprenda correttamente il valore di f, la perdita in tale scenario sarà molto maggiore rispetto al guadagno anche nel caso in cui molti altri giocatori non apprendano tale valore. Dunque:

Teorema 4.1 (Shoham-Tennenholtz)**.** *Nel caso in cui: (a) i partecipanti ad un gioco mediato di valutazione di funzione valutino la proprietà di correttezza maggiormente rispetto alla proprietà di esclusività e (b) le loro funzioni utilità considerino unicamente i vincoli di correttezza ed esclusività, una funzione è NCC se e solo se è non-reversibile e non-dominata [47].*

Corollario 4.1.1. *Ogni funzione che è reversibile o dominata è non-NCC.* [5]

Le uniche funzioni che potranno essere calcolate in modo corretto e sicuro da un protocollo MPC, anche sotto l'assunzione che esista una terza parte fidata (mediatore), saranno le *funzioni NCC*: i partecipanti sono incentivati a rivelare i propri input privati unicamente nel caso in cui stiano cercando di calcolare il valore di funzioni NCC [43].

4.2.3 *Funzioni* K*-NCC*

Per il momento, abbiamo considerato il caso in cui più giocatori, in modo indipendente e senza formare coalizioni, decidano di non seguire la strategia canonica di inviare i propri input corretti al mediatore; in sostanza, sono ammesse coalizioni di al massimo $(K = 1)$ giocatori. Cerchiamo adesso di analizzare il caso in cui ci sia collusione tra un gruppo di K partecipanti al gioco: è possibile definire la classe di funzioni che sono *computabili in modo non-cooperativo* anche in

5 Sotto l'assunzione che l'input di ogni giocatore sia necessario per il calcolo della funzione (per ogni giocatore P_i, $\exists t_i, t_i', t_{-i}$ tali che $f(t_i, t_{-i}) \neq f(t_i', t_{-i})$), è facile dimostrare che ogni funzione *dominata* è anche *reversibile*.

presenza di tali coalizioni. Funzioni per le quali anche coalizioni di al massimo K giocatori saranno incentivate ad inviare i propri input segreti in modo corretto al mediatore saranno dette funzioni K-*NCC (*K-*non-cooperatively computable)*.

Tale classe di funzioni è stata introdotta da Ashlagi, Klinger e Tenneholtz nell'articolo *"K-NCC: Stability Against Group Deviations in Non-cooperative Computation"* [52]; dunque, seguendo la loro trattazione, generalizziamo la definizione 4.9 per il caso in cui siamo presenti coalizioni di al massimo K giocatori.

Definizione 4.12 (*Funzioni* K-*NCC*)**.** Dato un *gioco mediato di valutazione di funzione* f, diremo che f è una *funzione* K-*NCC* se, per ogni sottoinsieme di k giocatori che formano una coalizione $C_i = \{P_{i_1}, \ldots, P_{i_k}\}$, $k \leqslant K$, per ogni tupla di loro strategie $((m_{i_1}, r_{i_1}), \ldots, (m_{i_k}, r_{i_k}))$ e per ogni input $t_{i_j} \in T$ con $1 \leqslant j \leqslant k$, vale la seguente proprietà:

- Esiste un t_{-C_i}, vettore degli input dei giocatori che non sono in C, ed esiste $P_j \in N$ tale che $P_{i_j} \in C_i$, per cui

  $$r_{i_j}(e(t_{i_j}), f(m_{i_1}(t_{i_1}), \ldots, m_{i_k}(t_{i_k}), t_{-C_i})) \neq f(t_{i_1}, \ldots, t_{i_k}, t_{-C_i});$$

  ciò significa che esiste un giocatore della coalizione che ha mentito sul proprio input e che la coalizione non è in grado di ricostruire il corretto valore della funzione in modo autonomo.

- Oppure, per ogni t_{-C_i}, abbiamo che

  $$f(m_{i_1}(t_{i_1}), \ldots, m_{i_k}(t_{i_k}), t_{-C_i}) = f(t_{i_1}, \ldots, t_{i_k}, t_{-C_i});$$

  in altre parole, ogni giocatore ha fornito al mediatore il proprio input segreto in modo corretto.

Generalizziamo anche la definizione 4.11 per le coalizioni di esattamente k giocatori.

Definizione 4.13 (*Funzione* k-*reversibile*)**.** Data una funzione $f\colon T^n \to O$, diremo che f è *k-reversibile*, se, per qualche vettore di k input $t_{C_i} = (t_{i_1}, \ldots, t_{i_k}) \in T$, esistono un altro input $t'_{C_i} = (t'_{i_1}, \ldots, t'_{i_k}) \in T$ ed una funzione g tali che:

- (a) $\forall t_{-C_i} \in T$ abbiamo che $g(f(t'_{C_i}, t_{-C_i}), t_{C_i}) = f(t_{C_i}, t_{-C_i})$;
- (b) per qualche t_{-C_i} abbiamo che $f(t'_{C_i}, t_{-C_i}) \neq f(t_{C_i}, t_{-C_i})$.

Dunque, una funzione reversibile può essere considerata come una funzione $(k = 1)$-reversibile. In generale, le funzioni NCC possono essere considerate funzioni $(K = 1)$-NCC.

Teorema 4.2 (Ashlagi-Klinger-Tenneholtz)**.** *Una funzione booleana è* K*-NCC se e solo se è* $(K-1)$*-NCC ed è non-k-reversibile per* $k = K$.

Il teorema appena enunciato, dimostrato da Ashlagi, Klinger e Tenneholtz in [52], non prende in considerazione il caso in cui $K = n$, dove n è il numero di variabili della funzione: una coalizione composta da n partecipanti, comportandosi in modo "scorretto", non porterà dei benefici a nessuno dei suoi affiliati, dal momento che non rimarrà nessun giocatore onesto da indurre in errore. Quindi, il seguente teorema risulta immediato:

Teorema 4.3 (Funzione booleana n-NCC)**.** *Una funzione booleana* $f\colon \{0,1\}^n \to \{0,1\}$ *è* n*-NCC se e solo se è* $(n-1)$*-NCC [52].*

Possiamo adesso enunciare la condizione sufficiente e necessaria affinché una funzione booleana sia K-NCC:

Corollario 4.3.1 (Funzioni booleane K-NCC e fortemente-NCC)**.** *Una funzione booleana* $f\colon \{0,1\}^n \to \{0,1\}$ *è* K*-NCC se e solo se è non-dominata e non-*k*-reversibile per ogni* $1 \leqslant k \leqslant K$*. Infine, diciamo che una funzione booleana è fortemente-NCC se e solo se è non-dominata e non-*k*-reversibile per ogni* $1 \leqslant k < n$ *[52].*

Ashlagi, Klinger e Tenneholtz hanno inoltre ottenuto due importanti risultati per una particolare classe di funzioni booleane: la classe delle *funzioni booleane simmetriche*. Una funzione booleana simmetrica è una funzione booleana il cui valore non dipende dalla permutazione dei suoi bit di ingresso, cioè dipende solo dal numero di 1 in ingresso [53] ; le funzioni parità e maggioranza discusse nell'esempio 4.2 appartengono a tale classe di funzioni. Ashlagi, Klinger e Tenneholtz hanno dimostrato che le funzioni simmetriche [6] ad n variabili che non sono 1-reversibili non sono nemmeno k-reversibili per qualunque $1 < k < n$; conseguenza di questo teorema è il fatto che qualunque funzione booleana simmetrica ad n variabili è fortemente-NCC se e solo se è NCC: dunque, sappiamo che almeno la funzione maggioranza è fortemente-NCC. Hanno inoltre dimostrato che determinare se una data funzione è K-NCC è un problema computazionalmente difficile: risulta essere un problema NP-difficile determinare se una funzione è dominata, così come lo è anche il determinare se una funzione è 1-reversibile.

4.2.4 *Equilibrio* K*-resistente*

Con l'introduzione delle funzioni K-NCC ci siamo posti per la prima volta il problema di analizzare un gioco strategico in cui un gruppo di K giocatori forma

6 Una funzione simmetrica ad n variabili è una funzione che risulta invariante sotto permutazione dei suoi argomenti [54].

una coalizione: in tale situazione, che si discosta dalla trattazione classica in cui si presuppone che i partecipanti al gioco agiscano in modo non-cooperativo ($K = 1$), il concetto di equilibrio di Nash risulta debole. La branca della Teoria dei Giochi che si occupa di giochi con coalizioni (*Coalitional Game Theory* [7]) introduce diverse nozioni di equilibrio cooperativo; nonostante ciò, per quanto riguarda la nostra trattazione, è necessario introdurre un nuovo e più "forte" concetto di equilibrio cooperativo: l'*equilibrio* K*-resistente (*K*-resilient equilibrium)*. Tale equilibrio generalizza l'equilibrio di Nash, ma permette ad una coalizione di al massimo K giocatori di cambiare le proprie strategie.

Seguiamo la definizione di *equilibrio di Nash* K*-resistente* proposta nell'articolo *"Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation"* di Abraham, Dolev, Gonen e Halpern [58].

Definizione 4.14 (*Miglior risposta di gruppo*)**.** Sia $G = (N, A, u)$ un gioco strategico, dato un sottoinsieme non vuoto di k giocatori che formano una coalizione $C_i = \{P_{i_1}, \ldots, P_{i_k}\} \subset N$, diremo che il vettore di strategie $s^*_{C_i} = (s^*_{i_1}, \ldots, s^*_{i_k})$ è *la miglior risposta di gruppo per* C_i *a* s_{-C_i}, vettore di strategie dei giocatori che non appartengono alla coalizione C_i, se, per ogni vettore di strategie di deviazione correlate $s_{C_i} \in S_{C_i}$ (S_{C_i} è l'insieme dei possibile vettori di strategie per la coalizione C_i) e per tutti i $P_{i_j} \in C_i$, abbiamo che $u_{i_j}(s^*_{C_i}, s_{-C_i}) \geqslant u_{i_j}(s_{C_i}, s_{-C_i})$.

Definizione 4.15 (*Equilibrio di Nash* K*-resistente*)**.** Sia $G = (N, A, u)$ un gioco strategico, diremo che il vettore di strategie indipendenti $s^* = (s^*_1, \ldots, s^*_n) = (s^*_{C_i}, s^*_{-C_i})$ è un *equilibrio di Nash* K*-resistente* se, per ogni coalizione $C_i \subset N$ con $|C_i| \leqslant K$, il vettore di strategie dei giocatori aderenti alla coalizione $s^*_{C_i}$ è la *miglior risposta di gruppo per* C_i *a* $s^*_{-C_i}$: in sostanza se nessun appartenente a C_i beneficia di qualunque altra scelta strategica $s_{C_i} \neq s^*_{C_i}$.

Dalla definizione si può notare che un equilibrio di Nash ($K = 1$)-resistente è a tutti gli effetti un semplice equilibrio di Nash. Inoltre, una situazione di equilibrio K-resistente ($s^* = (s^*_{C_i}, s^*_{-C_i})$) implica "protezione" contro tutte le deviazioni di coalizione (situazioni in cui la coalizione C_i è incentivata a cambiare il proprio vettore di strategie $s^*_{C_i}$ con un diverso s_{C_i}) che portano benefici, cioè payoff maggiori, anche ad un solo componente della coalizione. Se considerassimo equilibri cooperativi che "proteggono" unicamente da deviazioni nella quali ogni partecipante alla coalizione ottiene payoff maggiori, consentiremmo, in sostanza, situazioni nelle quali un unico giocatore controlla in modo effettivo la coalizione: cioè, il caso in cui un giocatore malevolo ha corrotto un numero di al massimo K giocatori. Una situazione di equilibrio K-resistente è molto stabile, dato che nessun giocatore della coalizione trarrebbe beneficio da un cambiamento di una strategia nel vettore $s^*_{C_i}$ (quindi da un diverso vettore di

7 Per una trattazione completa della *Coalitional Game Theory* consultare [2].

strategie s_{C_i}) [58].

Possiamo estendere in modo naturale la nozione di K-resistenza all'equilibrio correlato (CE), ottenendo due varianti in base a come definiamo la collusione nei giochi mediati. Nel caso in cui sia permessa *collusione ex ante*, gli appartenenti ad una coalizione C_i potranno coordinare le proprie strategie solo prima di ricevere le azioni consigliate dal mediatore ed in seguito non potranno più comunicare: dunque una strategia di deviazione dirà ad ogni membro della coalizione come cambiare la propria azione raccomandata, ma ciò verrà fatto senza conoscere quelle suggerite agli altri giocatori. Alternativamente, nel caso in cui sia consentita *collusione ex post*, gli appartenenti ad una coalizione C_i potranno comunicare dopo aver ricevuto le proprie azioni raccomandate dal mediatore: dunque i membri della coalizione osserveranno l'intero vettore delle azioni a loro raccomandate $a^*_{C_i}$ e potranno provare a cambiare congiuntamente tale vettore in qualche a_{C_i}.

Facendo riferimento alla definizione 4.3 di equilibrio correlato e seguendo la trattazione proposta nell'articolo *"Bridging game theory and cryptography: recent results and future directions"* di Katz [46]:

Definizione 4.16 (*Equilibrio correlato ex ante* K*-resistente*)**.** Dato un gioco mediato $G = (N, A, u)$, una distribuzione $\mathcal{M} \in \Delta(A)$ è un *equilibrio correlato ex ante* K*-resistente* (con $1 \leqslant K < n$) se, per ogni coalizione $C_i = \{P_{i_1}, \ldots, P_{i_k}\} \subset N$ con $|C_i| \leqslant K$, per un qualunque insieme di funzione di deviazione $\{r_j : A_j \to A_j\}_{\forall P_j \in C_i}$ e per ogni $P_j \in C_i$, vale che

$$u_j(\mathcal{M}) \geqslant u_j(\{r_j(a^*_j)\}_{\forall P_j \in C_i}, a^*_{-C_i}) = u_j(r_{i_1}(a^*_{i_1}), \ldots, r_{i_k}(a^*_{i_k}), a^*_{-C_i})$$

(dove $a^* = (a^*_1, \ldots, a^*_n) = (a^*_{C_i}, a^*_{-C_i})$ è un campione di azioni selezionato in accordo con $\mathcal{M}$).

Definizione 4.17 (*Equilibrio correlato ex post* K*-resistente*)**.** Dato un gioco mediato $G = (N, A, u)$, una distribuzione $\mathcal{M} \in \Delta(A)$ è un *equilibrio correlato ex post* K*-resistente* (con $1 \leqslant K < n$) se, per ogni coalizione $C_i = \{P_{i_1}, \ldots, P_{i_k}\} \subset N$ con $|C_i| \leqslant K$, per una qualunque funzione di deviazione di coalizione $r_{C_i} : A_{C_i} \to A_{C_i}$ e per ogni $P_j \in C_i$, vale che

$$u_j(\mathcal{M}) \geqslant u_j(r_{C_i}(a^*_{C_i}), a^*_{-C_i})$$

(dove $a^* = (a^*_1, \ldots, a^*_n) = (a^*_{C_i}, a^*_{-C_i})$ è un campione di azioni selezionato in accordo con $\mathcal{M}$).

La situazione ex ante pone forti restrizioni ed è facile osservare che, di norma, situazioni di CE ex ante K-resistenti sono più facilmente raggiungibili, spesso con guadagni maggiori, rispetto a situazioni di CE ex post K-resistenti. Date le

definizione precedenti, la loro generalizzazione per i giochi ad informazione incompleta risulta immediata [25].

Per concludere la discussione al riguardo del concetto di equilibrio K-resistente, dobbiamo aggiungere un'ultima considerazione riguardo al problema di *progettazione di meccanismi* introdotto nella sottosezione 4.2.2.

Definizione 4.18 (*Meccanismo* K*-resistente*)**.** Data una qualche funzionalità $\mathcal{F}$, diremo che un meccanismo (Ω, σ) è un *meccanismo* K*-resistente* per $\mathcal{F}$ se σ è un equilibrio K-resistente di Ω e i risultati di (Ω, σ) soddisfano $\mathcal{F}$ [58].

Dunque, dopo aver introdotto il concetto di equilibrio K-resistente ed il concetto di funzioni K-NCC, non risulta difficile giungere alla seguente conclusione: dato un gioco di valutazione di funzione Ω che ammetta la presenza di coalizioni con al massimo K partecipanti, il profilo strategico costituito unicamente da *strategie oneste* σ, informalmente il profilo strategico in cui la strategia adottata da ogni partecipante è inviare correttamente il proprio input al mediatore e seguire l'azione raccomandata, sarà equilibrio correlato ex post K-resistente per Ω solo nel caso in cui la funzione che i giocatori desiderano valutare è una funzione K-NCC. Diremo allora che (Ω, σ) è un *meccanismo* K*-resistente per la valutazione di funzioni* K*-NCC* [58].

4.3 MPC RAZIONALE

Sia Ω un *gioco mediato di valutazione di funzione*, se f, la funzione che i giocatori desiderano stimare in Ω, è una *funzione* K*-NCC*, allora sarà nell'interesse di ogni partecipante ad Ω (ed anche di coalizioni di al massimo K partecipanti) fornire in modo corretto i propri input segreti al mediatore M. Possiamo adesso porci la seguente domanda: è possibile sostituire M, il mediatore che computa il valore dalla funzione sugli input segreti dei giocatori, con un corrispondente protocollo MPC per il calcolo di f? Una risposta positiva alla domanda appena posta implicherebbe che un dato protocollo MPC sicuro π calcola f in modo sicuro non solo da un punto di vista crittografico, ma anche dal punto di vista razionale della teoria dei giochi. In tal modo avremmo i giocatori egoisti e razionali che eseguono a tutti gli effetti un protocollo MPC per il calcolo di f [25].

Per poter sostituire il mediatore con un appropriato protocollo MPC andiamo ad introdurre ulteriori concetti avanzati di teoria dei giochi.

4.3.1 *Giochi estesi tramite cheap-talk*

Analizziamo una particolare classe di giochi nella quale i partecipanti possono comunicare prima dell'inizio del gioco vero e proprio: i *giochi estesi tramite cheap-*

talk.

In sostanza, un gioco G viene esteso aggiungendo una fase iniziale, alle volte detta preambolo, che consente ai giocatori di comunicare, prima dell'inizio di G, tramite un numero finito di scambio messaggi . Tale comunicazione non influenzerà in modo diretto i possibili payoff di G e, per tale motivo, questa prima fase viene chiamata *fase di cheap-talk*, letteralmente fase delle "chiacchiere a buon mercato": inviare e ricevere messaggi avviene in modo "gratuito", cioè non influenza in modo negativo o positivo i possibili guadagni dei giocatori [55] [56].

Definizione 4.19 (*Gioco esteso tramite cheap-talk*)**.** Dato un gioco (ad informazione completa oppure incompleta [8]) G ed un protocollo di comunicazione π, è sempre possibile definire un gioco G^* costituito dalle seguenti fasi:

1. *Fase di cheap-talk.* I partecipanti al gioco si scambiano messaggi seguendo il protocollo π in un appropriato *modello di comunicazione*.
2. *Fase di gioco.* I giocatori partecipano al gioco originale G.

I payoff dei giocatori in G^* saranno identificati unicamente dai payoff che i giocatori ottengono in G. La strategia s_i di un giocatore P_i nel gioco G^* verrà identificata dalla strategia che questi segue nella fase di cheap-talk seguita dalla scelta di un'azione a_i da giocare in G. G^* sarà detto *gioco esteso tramite cheap-talk*.

Si assume che le parti possano sempre interrompere la fase di cheap-talk passando prematuramente alla fase di gioco: quindi, una strategia valida per P_i in G^* dovrà includere le istruzioni riguardo a quale azione P_i dovrà seguire nel caso in cui qualche altro giocatore devii dal protocollo π durante la prima fase del gioco. È importante evidenziare che la definizione di equilibrio (di Nash o correlato) per un gioco esteso tramite cheap-talk rimane invariata [25].

I giochi estesi tramite cheap-talk rappresentano un caso speciale di una categoria più ampia, cioè la categoria dei *giochi sequenziali*. Nei giochi sequenziali il presupposto che i giocatori agiscano contemporaneamente è rimosso; essi, infatti, si sviluppano su più turni: in ogni turno, a seconda di com'è definito il gioco, può essere richiesto che tutti i partecipanti scelgano e giochino simultaneamente un'azione oppure che solamente un loro sottoinsieme agisca [46]. Nel caso specifico dei giochi estesi tramite cheap-talk: (1) la fase di cheap-talk durerà vari turni (il numero esatto di turni varia in base al protocollo π), durante ciascuno dei quali ogni giocatore invierà un messaggio ad un altro partecipante

8 Nel caso in cui il gioco sia ad informazione incompleta $G = (N, A, T, p, u)$, prima dell'inizio della fase di cheap-talk, il vettore dei tipi $(t_1, \ldots, t_n)$ è scelto in accordo con la distribuzione di probabilità p. Ad ogni giocatore P_i viene assegnato il tipo t_i [55].

e tutti i messaggi saranno consegnati prima che il turno successivo cominci (si suppone un *modello di comunicazione sincrona*); (2) la fase di gioco durerà un unico turno durante il quale tutti i giocatori parteciperanno al gioco "one-shot" G.

Giochi in forma estesa

I giochi sequenziali sono di norma rappresentati attraverso la *forma estesa (extensive form)*, cioè, secondo la teoria dei grafi, attraverso una struttura ad albero con radice detta *albero del gioco*: dunque, giochi siffatti saranno chiamati *giochi in forma estesa* [9] [57].

Basandoci sulle trattazioni proposte da Katz in [46], da Abraham, Dolev, Gonen e Halpern in [58] e da Halpern e Teague in [43], descriviamo in modo informale la rappresentazione in *forma estesa* di un gioco sequenziale.
Dato un gioco sequenziale G, è possibile rappresentare tale gioco tramite l'albero $\mathcal{T}_G$. Il nodo radice di $\mathcal{T}_G$ descrive lo stato iniziale del gioco, i nodi successivi rappresentano i risultati delle azioni intraprese dai giocatori. Ad ogni nodo, ogni giocatore P_i possiede uno *stato locale*, che rappresenta a tutti gli effetti la "storia" del giocatore, cioè le proprie informazioni iniziali, i messaggi inviati (le proprie azioni) ed i messaggi ricevuti (le azioni degli altri giocatori): in sostanza, durante l'esecuzione di un gioco, ad ogni turno è associata una *storia*, o *cronologia*, delle azioni selezionate dai giocatori fino a quel momento. Ogni esecuzione del gioco corrisponde ad un cammino dell'albero che comincia dal nodo radice: tale cammino potrà essere infinito oppure potrà terminare in un nodo foglia; la storia associata a tale nodo sarà detta *storia terminale (terminal histories)*. Ad ogni cammino viene associata una tupla $(u_1, \ldots, u_n)$, le funzioni utilità dei giocatori nel caso in cui tale cammino venga intrapreso. Per alberi finiti, dunque per giochi finiti (il nostro caso), possiamo identificare i payoff ricavati dai giocatori dopo l'esecuzione del gioco con i payoff associati alle foglie dell'albero: dunque, la funzione utilità di ogni giocatore adesso è funzione delle *storie terminali*. Normalmente, nella teoria dei giochi, per ogni giocatore P_i, i nodi sono partizionati in *insiemi di informazione (information set)*, cioè nodi che l'i-esimo giocatore non è in grado di distinguere: utilizzando il concetto di *stato locale*, possiamo dire che l'insieme di informazione di P_i ad un dato nodo v è costituito da tutti i nodi v' per i quali lo *stato locale* di P_i è lo stesso. Diremo che un giocatore possiede *memoria perfetta (perfect recall)* se è in grato di ricordare tutti i propri precedenti insiemi di informazione e le proprie azioni (noi considereremo questo caso). La strategia per un giocatore P_i è in sostanza una funzione che va dall'insieme degli stati locali di P_i all'insieme delle sue possibili azioni (dunque, usando la terminologia della teoria dei giochi, è una una funzione che va dagli insiemi di informazione all'insieme delle possibili

9 Per una trattazione completa dei *giochi in forma estesa* consultare [2].

azioni). Alternativamente, possiamo dire che una strategia per un giocatore P_i "dice" al giocatore i-esimo quale azione selezionare in ogni possibile nodo di $\mathcal{T}_G$: ergo, è una funzione che va dall'insieme delle possibili *storie* all'insieme delle azioni di P_i. Per concludere possiamo osservare che un dato profilo strategico $s = (s_1, \ldots, s_n)$ determina una distribuzione di probabilità sui possibili cammini di $\mathcal{T}_G$.

Qualunque gioco in forma estesa può essere visto come un gioco in forma normale facendo corrispondere l'insieme delle azioni disponibili (nella versione in forma normale) con le strategie dei giocatori (nella versione in forma estesa); quindi, tutti i concetti di equilibrio precedentemente analizzati possono essere applicati ai giochi in forma estesa. È, tuttavia, importante notare che l'equilibrio di Nash rappresenta un concetto di soluzione troppo "generoso" per gioco in forma estesa, in quanto consente a profili strategici "irragionevoli" di soddisfarne la definizione [46].

Equilibri nei giochi in forma estesa

La scelta di un equilibrio di Nash in un gioco in forma estesa prende in considerazione *storie irrealizzabili* (una storia h è detta *realizzabile rispetto ad un dato profilo strategico* σ se h ha una probabilità positiva di realizzarsi nel caso in cui tutti i giocatori seguano σ): in generale possiamo dire che, data una qualunque *storia realizzabile*, ogni giocatore giocherà in modo razionale, mentre, per una qualunque *storia irrealizzabile*, "minaccerà" gli altri giocatori di agire in modo irrazionale (*minaccia non credibile*). In sostanza, possiamo dire che un equilibrio di Nash in un gioco in forma estesa può essere rappresentato da un profilo strategico condizionato da una *minaccia non credibile*. Di conseguenza risulta fondamentale raffinare il concetto di equilibrio di Nash per poter eliminare le strategie che contengono *minacce non credibili*.

Un dato vettore di strategie può essere un equilibrio di Nash (NE), ma contenere *minacce irrealizzabili* per uno o più giocatori: la nozione di *NE sottogioco-perfetto (subgame-perfect NE)* elimina tale problema. Per prima cosa definiamo in modo informale il concetto di *sottogioco* G^{*h}, o *gioco ridotto*, del gioco in forma estesa G^*: G^{*h} corrisponde al gioco G^* nel caso in cui si consideri fissata una data *storia* iniziale h; in pratica possiamo vedere G^{*h} come una continuazione di G^* condizionata dal fatto che la storia h è stata osservata. Una strategia σ_i per il gioco G^* induce naturalmente una strategia σ_i^h per il gioco G^{*h} fissando $\sigma_i^h(h') = \sigma_i(h||h')$, dove con $h||h'$, secondo la notazione usata da Katz in [46], identifichiamo la concatenazione della storia h con la storia h'.

Definizione 4.20 (*NE sottogioco-perfetto*)**.** Sia G^* un gioco in forma estesa e sia σ un equilibrio di Nash in G^*. Diremo che σ è *sottogioco-perfetto* se, per tutte le

possibili storie h di G^*, il vettore di strategie σ^h è un equilibrio di Nash per il gioco ridotto G^{*h} [46].

Un profilo strategico σ sarà equilibrio di Nash per il gioco ridotto unicamente nel caso di *storie realizzabili* rispetto al profilo strategico σ; quindi, un profilo strategico che è NE sottogioco-perfetto non conterrà *minacce irrealizzabili*.

Equilibrio di Nash Computazionale

Nei giochi estesi tramite cheap-talk, durante il preambolo, i partecipanti si scambiano messaggi seguendo un dato protocollo di comunicazione π: possiamo considerare il caso in cui tali partecipanti posseggano potere computazionale limitato. Dunque, facciamo riferimento al *modello computazionale* definito nella sottosezione 3.2.3.

Per definire in modo formale il potere computazionale dei giocatori è necessario introdurre un parametro di sicurezza λ e richiedere che:

- (a) tutte le computazioni e le comunicazioni avvengano in tempo polinomiale in λ; quindi, anche le funzioni utilità dovranno essere computate in tempo polinomiale in λ;
- (b) le strategie di tutti i giocatori debbano essere computate in tempo polinomiale in λ

Diremo che le strategie e le funzioni utilità dei giocatori dovranno essere *efficienti in* λ; ergo, in questo modello di gioco con partecipanti limitati computazionalmente, dobbiamo preoccuparci unicamente delle strategie di comportamento scorretto *efficienti* nel parametro di sicurezza.

Per implementare tale versione "computazionale" dei giochi estesi, prima della fase di cheap-talk, viene passato ad ogni giocatore il valore di λ, dopodiché si procede con la normale esecuzione del gioco esteso. In sostanza, la fase di cheap-talk è progettata sotto l'assunzione che esista un problema computazionalmente difficile non risolvibile in un tempo "utile"da un giocatore che segue una strategia disonesta s_i^λ *efficiente in* λ. Questa assunzione introduce una probabilità trascurabile che un partecipante, seguendo la strategia s_i^λ, possa risolvere il problema computazionalmente difficile ed ottenere un payoff maggiore rispetto a quello che avrebbe ottenuto seguendo la strategia $s_i^{*\lambda}$, anch'essa *efficiente in* λ, che appartiene ad un equilibrio $s^{*\lambda}$ del gioco esteso. Ovviamente seguire la strategia s_i^λ può portare un miglioramento dell'utilità attesa del giocatore P_i al massimo di una quantità trascurabile (i parametri di questo gioco "computazionale" sono assunti essere costanti rispetto a λ): dunque, si assume che un giocatore non sarà incentivato a seguire una strategia di deviazione se le sue vincite aumenteranno solo di una quantità trascurabile; i partecipanti saranno

indifferenti a cambiamenti trascurabili nella loro utilità [44] [25] [46].

Definiamo l'*equilibrio di Nash computazionale* generalizzando al caso con n giocatori la definizione introdotta in *"A Cryptographic Solution to a Game Theoretic Problem"* da Dodis, Halevi e Rabin [44].

Definizione 4.21 (*Equilibrio di Nash computazionale*)**.** Dato un gioco esteso tramite cheap-talk con giocatori limitati computazionalmente $G^* = (N, A, u)$ ed un dato parametro di sicurezza λ, un *equilibrio di Nash computazionale* di G^* è un profilo di strategie indipendenti $s^* = (s_1^*, \ldots, s_n^*)$ in cui ogni strategia s_i^* è *efficiente in* λ tale che, per ogni P_i e per ogni altra strategia s_i *efficiente in* λ, $u_i(s_i^*, s_{-i}^*) \geqslant u_i(s_i, s_{-i}^*) - \epsilon(\lambda)$, dove ϵ è una funzione trascurabile (definizione 3.7) [10].

4.3.2 *Gioco di Multi-Party Computation*

Dalla definizione 4.19 sappiamo che, partendo da un gioco G in forma normale (ad informazione completa od incompleta), è sempre possibile definire il gioco G^*, versione estesa di G tramite cheap-talk. Dunque, risulterà chiaro che per sostituire il mediatore in un *gioco mediato di valutazione di funzione* f sarà necessario definire un appropriato *gioco di valutazione di funzione esteso tramite cheap-talk*, nel quale, durante la fase di cheap-talk, i partecipanti al gioco si scambieranno messaggi seguendo un protocollo MPC per il calcolo di f: chiameremo questa versione del gioco di valutazione di funzione *gioco di Multi-Party Computation*.

Definizione 4.22 (*Gioco di Multi-Party Computation*)**.** Dato G *gioco di valutazione di* f *funzione* K*-NCC* che ammette la presenza di coalizioni di al massimo K giocatori e π protocollo MPC sicuro (resistente a coalizioni di al massimo K partecipanti disonesti) per la computazione di f, chiamiamo *gioco di Multi-Party Computation* G^* la versione estesa di G tramite cheap-talk costituita dalle seguenti fasi:

1. *Fase di inizializzazione.* Il vettore degli input privati $(t_1, \ldots, t_n)$ è scelto in accordo con la distribuzione di probabilità p. Ad ogni giocatore P_i viene assegnato l'input t_i. Se i giocatori sono limitati computazionalmente, viene inoltre passato loro il parametro di sicurezza λ.

2. *Fase di cheap-talk.* I partecipanti al gioco si scambiano messaggi per un numero finito di round seguendo il protocollo MPC π in un appropriato *modello di comunicazione*. Conclusa tale fase, ogni giocatore possiede $o' = f(t_1', \ldots, t_n')$, valore della funzione f che i giocatori hanno calcolato condividendo gli input $(t_1', \ldots, t_n')$.

10 Anche le funzioni utilità $u_{i=1,\ldots,n}$ sono *efficienti in* λ.

3. *Fase di gioco.* Ogni giocatore partecipa normalmente al gioco di valutazione di funzione G.

I payoff dei giocatori in G^* saranno identificati unicamente dai payoff che i giocatori ottengono in G. La strategia s_i di un giocatore P_i nel gioco G^* verrà identificata dalla strategia che tale giocatore segue nella fase di esecuzione del protocollo seguita dalla scelta di un'azione a_i da giocare in G.

In sostanza, in un *Gioco di Multi-Party Computation* sostituiamo il mediatore per il calcolo di f con un protocollo MPC: è necessario però chiedersi se tale sostituzione risulti "efficace". Per comprendere cosa si intende per "efficace" è necessario fare un piccolo passo indietro.
Sia Ω il *gioco di valutazione di* f *funzione* K-*NCC* che ammette al massimo coalizioni di K partecipanti, sappiamo che in Ω_M , versione mediata di Ω, σ_M, il profilo strategico costituito unicamente da *strategie oneste* (informalmente il profilo strategico in cui la strategia adottata da ogni partecipante è inviare correttamente il proprio input al mediatore M e seguire l'azione raccomandata), è *equilibrio correlato (ex post)* K-*resistente*. Abbiamo visto che è possibile sostituire il mediatore M in Ω_M con un protocollo MPC sicuro π definendo Ω_{MPC} *gioco di Multi-Party Computation*, versione estesa tramite cheap-talk di Ω in cui i partecipanti illimitati/limitati computazionalmente comunicano seguendo π. In tale gioco identifichiamo con σ_π il profilo strategico in cui i partecipanti seguono onestamente il protocollo MPC e valutano la funzione con il valore ottenuto dall'esecuzione onesta di tale protocollo.
Quindi, chiedersi se questa sostituzione di M tramite π è stata "efficace" equivale a chiedersi se il profilo strategico σ_π risulta essere un equilibrio per Ω_{MPC} esattamente come σ_M era equilibrio per Ω_M: per la precisione ci chiediamo se σ_π risulti equilibrio di Nash regolare/computazionale K-resistente per Ω_{MPC}. Inoltre è necessario chiedersi se per ogni P_i vale $u_i(\sigma_M) = u_i(\sigma_\pi)$. Generalizziamo questa riflessione:

Definizione 4.23 (*Implementazione* K-*resistente di un mediatore*)**.** Sia G_M versione mediata tramite M di un gioco G e sia s_M profilo a *strategie oneste* ed *equilibrio correlato (ex post)* K-*resistente* in G_M. Sia G_{MPC} versione estesa tramite cheap-talk di G con π protocollo MPC, con partecipanti illimitati/limitati computazionalmente e sia s_π profilo strategico in G_{MPC}. Diremo che π è *implementazione* K-*resistente* di M se:

- (a) s_π è *equilibrio di Nash regolare/computazionale* K-*resistente* in G_{MPC};
- (b) per ogni P_i vale $u_i(s_M) = u_i(s_\pi)$.

L'unico elemento non determinato dalla definizione di *implementazione* K-*resistente di* M, definizione basata su quella proposta da Katz in [46], è la specifica del comportamento che i giocatori devono avere in G_{MPC} nel caso in

cui, per qualche ragione, la computazione fallisca. Tale comportamento dipende dai *requisiti di sicurezza MPC* (vedere sottosezione 3.2.1) che il protocollo MPC π soddisfa. Un *protocollo MPC che calcola* f *in modo sicuro* contro coalizioni di al massimo K giocatori garantisce tutti i requisiti di sicurezza MPC.

Dodis, Halevi e Rabin, in *"A Cryptographic Solution to a Game Theoretic Problem"* [44], hanno enunciato e dimostrato il seguente teorema per un caso particolare (due giocatori e nessuna coalizione); la versione proposta è basata su quella enunciata da Dodis e Rabin in [25]:

Teorema 4.4 (Dodis-Rabin)**.** *Sia* G_M *gioco ad informazione incompleta, versione mediata tramite* M *di un gioco* G*, sia* s_M *profilo a strategie oneste ed equilibrio correlato (ex post)* K*-resistente in* G_M*, siano* $(a_1, \ldots, a_n)$ *le azioni selezionate da* M *secondo una data distribuzione di probabilità basata sui tipi segreti dei giocatori, sia* $f(t_1, \ldots, t_n, r) = (a_1, \ldots, a_n)$ *la funzione probabilistica che descrive la strategia di selezione delle azioni raccomandate da* M*, allora è possibile definire il gioco* G_{MPC}*, versione estesa tramite cheap-talk di* G*, in cui il protocollo* π *è implementazione* K*-resistente di* M *se* π *è un protocollo MPC che computa* f *in modo sicuro contro coalizioni di al massimo* K *partecipanti illimitati/limitati computazionalmente.*

Dimostrazione. Facendo riferimento alla definizione 3.4, un protocollo MPC π che *calcola in modo sicuro* f garantisce i *requisiti di sicurezza MPC*; ergo, gli unici comportamenti scorretti ammessi per un giocatore P_i sono: (a) *mentire sul proprio valore di input*; (b) *non trasmettere alcun valore di input*. In sostanza, i giocatori possono scegliere in modo arbitrario i propri input. Osservando la definizione 4.5 vediamo che tale libertà è lasciata ai giocatori anche nel gioco mediato G_M, ma l'equilibrio K-resistente assicura l'irrazionalità di tale comportamento: sappiamo che la strategia adottata da ogni partecipante P_i sarà inviare correttamente il proprio tipo/input t_i e seguire l'azione a_i raccomandata da M (s_M profilo a *strategie oneste*). Dunque, il profilo strategico che forma un *equilibrio* K*-resistente* in G_{MPC} sarà il vettore di strategie s_π secondo cui, durante la fase di cheap-talk, ogni giocatore P_i segue onestamente le istruzioni del protocollo π, computando di conseguenza $(a_1, \ldots, a_n) = f(t_1, \ldots, t_n, r)$, e, durante la fase di gioco, seleziona l'azione a_i ottenuta dall'esecuzione corretta di π (in caso di comportamento scorretto durante la fase di cheap-talk, userà una qualunque strategia per selezionare una mossa nella fase di gioco).
Adesso dobbiamo mostrare che per ogni P_i vale $u_i(s_M) = u_i(s_\pi)$. Il gioco G_M è la versione mediata del gioco G, dunque, facendo riferimento alla definizione 4.5, i payoff di G_M sono stabiliti dai payoff in G: durante l'ultima fase del gioco G_M i giocatori seguono l'azione raccomandata dal mediatore, di conseguenza per ogni P_i vale che $u_i^M(a_i) = u_i(a_i)$. Allo stesso modo, il gioco G_{MPC} è la versione estesa di G e, per la definizione 4.19, sappiamo che i payoff dei giocatori in G_{MPC} sono identificati unicamente dai payoff che questi ottengono in G: durante la fase di gioco di G_{MPC} i giocatori selezionano l'azione ottenuta come output

dell'esecuzione corretta di π; dunque, per ogni P_i vale che $u_i^{MPC}(a_i) = u_i(a_i)$. Ciò ci porta a concludere che $u_i(s_M) = u_i(s_\pi)$ [46] [25]. □

Corollario 4.4.1. *Sia Ω_M la versione mediata tramite M di Ω gioco di valutazione di f funzione K-NCC che ammette al massimo coalizioni di K partecipanti e sia π protocollo MPC che calcola f in modo sicuro contro coalizioni di al massimo K partecipanti illimitati/limitati computazionalmente, allora π è implementazione K-resistente di M nel gioco di Multi-Party Computation Ω_{MPC}, versione estesa tramite cheap-talk di Ω.*

Dunque, da un punto di vista positivo, assumendo che l'obbiettivo dei partecipanti sia semplicemente raggiungere un equilibrio di Nash, tale risultato ci mostra che gli attuali protocollo MPC possono essere spiegati da un punto di vista razionale fintanto che i giocatori desiderano computare f nel modello ideale. Da un punto di vista negativo, tale risultato limita la classe di funzioni che possono essere computate razionalmente in modo congiunto alla sola classe delle funzioni K-NCC: riuscire a razionalizzare la MPC anche per funzioni non-NCC, per le quali la definizione crittografica di MPC continua ad aver senso, rimane un problema aperto. Un altro problema di questa formulazione di gioco di Multi-Party Computation è il fatto che, come abbiamo visto nella sottosezione 4.3.1 , la nozione di equilibrio di Nash risulta troppo debole per i giochi in forma estesa: è necessario cercare di raggiungere una tipologia di equilibrio più "forte" che sia in grado di spiegare gli attuali protocolli MPC oppure, alternativamente, è possibile cercare di progettare protocolli MPC abbastanza robusti in grado di raggiungere tali situazioni di equilibrio.

5

CONCLUSIONI

Le potenzialità dei *protocolli MPC sicuri* sono ovvie: molte applicazioni, come ad esempio le aste elettroniche oppure le votazioni elettroniche, possono essere viste come istanze di un *problema MPC*. Risulta evidente che implementare protocolli sicuri ed allo stesso tempo efficienti dal punto di vista pratico sia di primaria importanza. Per molti anni i *protocolli MPC ad uso generale*, cioè protocolli che consentono di calcolare qualsiasi funzione, sono stati considerati solo di interesse teorico; tale mancanza di applicazioni pratiche era dovuta, almeno in parte, al fatto che un'implementazione diretta dei primi protocolli avrebbe portato a soluzioni molto inefficienti. Recenti ricerche si sono occupate di risolvere questo problema di efficienza, sia per protocolli ad uso generale (vedere i seguenti articoli: [42], [59], [60]), sia per protocolli per casi specifici, come il voto elettronico (vedere i seguenti articoli: [61], [62]); i notevoli miglioramenti raggiunti nelle implementazioni dei protocolli MPC hanno evidenziato che, anche se ancora computazionalmente costosa, la Multi-Party Computation risulta di interesse pratico.

Una linea di ricerca ha posto particolare attenzione ad una vasta gamma di applicazioni economiche dei protocolli MPC: due progetti di ricerca che si occupano di tale campo sono il progetto SCET (Secure Computing, Economy and Trust) ed il progetto SIMAP (Secure Information Management and Processing) [63]. L'ultimo di questi progetti è stato responsabile della prima applicazione pratica di MPC: nel gennaio 2008, in Danimarca, per la prima volta nella storia, la Multi-Party Computation è stata utilizzata per spostare denaro reale nell'ambito di una doppia asta a livello nazionale in cui la compagnia Danisco, unica azienda che tratta la lavorazione di barbabietole da zucchero, ha messo all'asta dei contratti per loro vendita. Circa 1200 agricoltori hanno utilizzato un protocollo MPC per determinare il prezzo di mercato di tali contratti, senza dover rivelare i propri dati sensibili (prezzo di acquisto del contratto e prezzo di vendita della barbabietola) e senza ricorrere ad una terza parte fidata. L'intero calcolo è durato circa 30 minuti, un tempo ragionevole per un'attività che viene svolta non più di una volta l'anno.

I protocolli MPC ad uso generale nei prossimi anni diventeranno sempre più efficienti, permettendo di risolvere una sempre più ampia gamma di scenari reali sensibili alla privacy [64]; quindi, risulterà necessario analizzare i partecipanti a tali protocolli MPC non più come totalmente onesti oppure totalmente disonesti, ma bensì come individui egoisti, razionali ed intelligenti: in altre

parole, individui spinti ad agire unicamente dagli interessi personali. Per far ciò è necessario modellare il problema di *Multi-Party Computation* utilizzando la *Teoria dei Giochi*. Il lavoro di ricerca svolto da Dodis e Rabin [25] ci ha mostrato che, dato protocollo MPC π che calcola una data f funzione K-NCC in modo sicuro contro coalizioni di al massimo K partecipanti illimitati/limitati computazionalmente, è possibile definire un appropriato *gioco di Multi-Party Computation* Ω_{MPC} in cui le parti eseguono il protocollo π per il calcolo di f non solo da un punto di vista crittografico, ma anche dal punto di vista razionale della Teoria dei Giochi: abbiamo giocatori egoisti e razionali che eseguono a tutti gli effetti il protocollo π per il calcolo di f. In tale gioco sappiamo che il *profilo a strategie oneste* σ_π, cioè il profilo strategico in cui la strategia adottata da ogni giocatore è condividere onestamente il proprio input segreto e seguire correttamente π, risulta *equilibrio di Nash regolare/computazionale* K*-resistente*: in sostanza, sappiamo che i giocatori saranno incentivati ad agire in modo corretto. Dal punto di vista della *progettazione di meccanismi* sappiamo che la coppia $(\Omega_{MPC}, \sigma_\pi)$ risulta essere un *meccanismo* K*-resistente per la valutazione di funzioni* K*-NCC*.

Problema aperto è riuscire ad estendere tale risultato anche per funzioni non-NCC. Restringere la MPC razionale alle sole funzioni K-NCC risulta molto limitante; inoltre, stabilire se una funzione è K-NCC, come dimostrato da Ashlagi, Klinger e Tenneholtz in [52], è un problema computazionalmente difficile. Per il momento, sempre grazie ai risultati di Ashlagi et al., sappiamo che qualunque funzione booleana simmetrica ad n variabili è fortemente-NCC (K-NCC per ogni $K = k$ con $1 \leqslant k < n$) se e solo se è NCC: di conseguenza, sappiamo per certo che la funzione maggioranza (definita nella discussione relativa all'esempio 4.2) è fortemente-NCC. Dunque, in un gioco di Multi-Party Computation in cui i giocatori eseguono un protocollo per il calcolo sicuro della funzione maggioranza, il profilo a strategie oneste risulterà equilibrio di Nash regolare/computazionale K-resistente.

Ulteriore problema di questa formulazione di gioco di Multi-Party Computation è legato al fatto che, come abbiamo visto nella sezione 4.3.1, l'equilibrio di Nash nei giochi in forma estesa prende in considerazione anche *storie irrealizzabili*: sviluppi futuri potranno orientarsi verso la progettazione di protocolli MPC sufficientemente robusti per raggiungere situazioni di equilibrio più "forti".

Infine, è importante sottolineare che, nel campo economico della progettazione di meccanismi, il concetto di una terza parte fidata è un'assunzione centrale: al giorno d'oggi molti *meccanismi pratici* richiedono un mediatore e risulta naturale considerare la possibilità di implementarlo utilizzando un protocollo di Multi-Party Computation. Il teorema 4.4, enunciato per la prima volta, in una versione semplificata, nell'articolo *"A Cryptographic Solution to a Game Theoretic Problem"* di Dodis, Halevi e Rabin [44], stabilisce la condizione sufficiente

affinché un protocollo MPC possa essere considerato implementazione della terza parte fidata di un dato gioco mediato in cui il profilo a strategie oneste è equilibrio correlato: il protocollo MPC in questione deve *computare in modo sicuro* la funzione probabilistica che descrive la strategia di selezione delle azioni raccomandate dal mediatore.

BIBLIOGRAFIA

[1] Hermann Hesse, "Siddhartha", *Edizione Italiana Piccola Biblioteca Adelphi, Traduzione di Massimo Mila*, 1975 (Cited on page iii.)

[2] Kevin Leyton-Brown, Yoav Shoham, "Essentials of Game Theory: A Concise Multidisciplinary Introduction", *Morgan and Claypool Publishers*, 2008 (Cited on pages 1, 5, 7, 8, 10, 12, 13, 14, 15, 16, 18, 19, 46, 47, 58, and 62.)

[3] Martin J. Osborne, Ariel Rubinstein, "A Course in Game Theory", *The MIT Press*, 1994 (Cited on pages 1, 5, 6, 7, 9, 11, 13, and 20.)

[4] "Wikipedia: Game theory", `https://en.wikipedia.org/wiki/Game_theory` (Cited on pages 1 and 5.)

[5] John von Neumann, Oskar Morgenstern, "Theory of Games and Economic Behavior", *Princeton University Press*, 1944 (Cited on pages 1 and 7.)

[6] Martin J. Osborne, "An introduction to game theory", *Oxford University Press*, 2004 (Cited on page 1.)

[7] Noam Nisan, Amir Ronen, "Algorithmic mechanism design", *Proceedings of the 31st ACM Symposium on Theory of Computing*, 1999 (Cited on page 1.)

[8] "Wikipedia: Algorithmic game theory", `https://en.wikipedia.org/wiki/Algorithmic_game_theory` (Cited on page 2.)

[9] Andrew C. Yao, "Protocols for secure computations", *FOCS. 23rd Annual Symposium on Foundations of Computer Science*, 1982 (Cited on pages 2, 22, and 35.)

[10] Daniele Venturi, "Crittografia nel Paese delle Meraviglie", *Dipartimento di Ingegneria, Elettronica e Telecomunicazioni, Sapienza Università di Roma, Springer*, 2012 (Cited on pages 2, 26, 29, 33, 34, 38, 39, and 42.)

[11] Myerson, Roger B., "Game Theory: Analysis of Conflict", *Harvard University Press*, 1991 (Cited on page 5.)

[12] A. Agnetis, "Introduzione alla Teoria dei Giochi", *Dipartimento di Ingegneria dell'Informazione, Università degli studi di Siena* (Cited on pages 5, 6, 7, 10, 12, and 13.)

[13] Ran Canetti, Alon Rosen, "Scribe notes on Cryptography and Game Theory",
http://www.cs.tau.ac.il/~canetti/f09-cgt.html, *Tel Aviv University*, 2009 (Cited on pages 5, 6, 7, 8, 15, 16, 17, and 44.)

[14] "Wikipedia: Teoria dei giochi",
https://it.wikipedia.org/wiki/Teoria_dei_giochi (Cited on page 5.)

[15] Noam Nisan, Tim Roughgarden, Eva Tardos, Vijay V. Vazirani, "Algorithmic Game Theory", *Cambridge University Press*, 2007 (Cited on pages 7, 9, and 74.)

[16] "Wikipedia: Giochi a informazione completa",
https://it.wikipedia.org/wiki/Gioco_a_informazione_completa (Cited on page 8.)

[17] "Wikipedia: Dilemma del prigioniero",
https://it.wikipedia.org/wiki/Dilemma_del_prigioniero (Cited on page 9.)

[18] "Wikipedia: Strategy (game theory)",
https://en.wikipedia.org/wiki/Strategy_(game_theory) (Cited on page 10.)

[19] "GameTeory.net dictionary: Pure Strategy",
http://www.gametheory.net/dictionary/PureStrategy.html (Cited on page 10.)

[20] "GameTeory.net dictionary: Mixed Strategy",
http://www.gametheory.net/dictionary/MixedStrategy.html (Cited on page 10.)

[21] "Wikipedia: Solution concept",
https://en.wikipedia.org/wiki/Solution_concept (Cited on page 13.)

[22] Ashish Goel, "Lecture notes on Internet Commerce",
http://web.stanford.edu/~ashishg/msande235/spr08_09/Lecture02.pdf, *Stanford University*, 2009 (Cited on pages 16 and 20.)

[23] Thomas S. Ferguson, "GAME THEORY",
http://www.math.ucla.edu/~tom/Game_Theory/Contents.html
Mathematics Department, UCLA 2014 (Cited on page 19.)

[24] Ronald Cramer, Ivan Damgard, "Multiparty Computation, an Introduction", *Advanced Courses in Mathematics, CRM Barcelona*, 2004 (Cited on pages 22, 25, 28, 29, and 35.)

[25] Yevgeniy Dodis, Tal Rabin, "Cryptography and Game Theory", http://www.cs.nyu.edu/~dodis/ps/game-survey.pdf, invited book chapter in *"Algorithmic Game Theory"* [15], 2007 (Cited on pages 22, 31, 32, 37, 44, 46, 48, 50, 51, 54, 55, 60, 61, 65, 67, 68, and 70.)

[26] "Wikipedia: Secure multi-party computation", https://en.wikipedia.org/wiki/Secure_multi-party_computation (Cited on pages 23 and 25.)

[27] Y. Lindell, B. Pinkasy, "Secure Multiparty Computation for Privacy-Preserving Data Mining", *The Journal of Privacy and Confidentiality*, 2009 (Cited on pages 25, 28, 29, 32, and 35.)

[28] Y. Lindell, Y. Aumann, "Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries", *Journal of Cryptology*, 2010 (Cited on page 26.)

[29] P. Miltersen, J. Nielsen, N. Triandopoulos, "Privacy-enhancing auctions using rational cryptography", *CRYPTO*, 2009 (Cited on page 26.)

[30] O. Goldreich, S. Micali, A. Wigderson, "How to play any mental game", *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, *AMC*, 1987 (Cited on pages 28 and 35.)

[31] R. Canetti, "Security and composition of multiparty cryptographic protocols", *Journal of CRYPTOLOGY*, *Springer*, 2000 (Cited on pages 30 and 33.)

[32] Oded Goldreich, "Foundations of Cryptography Volume 2, Basic Applications", *Cambridge University Press* , 2004 (Cited on page 33.)

[33] Oded Goldreich, "Foundations of Cryptography Volume 1, Basic Tools", *Cambridge University Press* , 2001 (Cited on pages 33 and 34.)

[34] M. Ben-Or, S. Goldwasser, A.Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation", *Proceedings of the 20th ACM Symposium on the Theory of Computing (STOC)*, 1988 (Cited on page 36.)

[35] Tal Rabin, Michael Ben-Or, "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority (Extended Abstract)", *Proceedings of the 21st ACM Symposium on the Theory of Computing (STOC)*, 1989 (Cited on page 36.)

[36] Ueli Maurer, "Secure Multi-party Computation Made Simple", *Security in Communication Networks: Third International Conference*, *Springer*, 2003 (Cited on page 36.)

[37] Matthias Fitzi, Martin Hirt, Ueli M. Maurer, "Trading Correctness for Privacy in Unconditional Multi-Party Computation (Extended Abstract)", *Advances in Cryptology CRYPTO, Springer* 1998 (Cited on page 36.)

[38] M. Hirt, U. Maurer, "Complete characterization of adversaries tolerable in secure multi-party computation", *Proceedings of the 16th ACM Symposium on Principles of Distributed Computing (PODC)*, 1997 (Cited on page 36.)

[39] Adi Shamir, "How to share a secret", *Communications of the ACM* 22, 1979 (Cited on page 38.)

[40] "Wikipedia: Campo finito", `https://it.wikipedia.org/wiki/Campo_finito` (Cited on page 38.)

[41] "Wikipedia: Interpolazione di Lagrange", `https://it.wikipedia.org/wiki/Interpolazione_di_Lagrange` (Cited on page 38.)

[42] Rosario Gennaro, Michael O. Rabin, Tal Rabin, "Simplified VSS and fast-track multiparty computations with applications to threshold cryptography", *Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing*, 1998 (Cited on pages 42 and 69.)

[43] Joseph Halpern, Vanessa Teague, "Rational secret sharing and multiparty computation: extended abstract", *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 2004 (Cited on pages 43, 49, 52, 55, and 62.)

[44] Yevgeniy Dodis, Shai Halevi, Tal Rabin, "A Cryptographic Solution to a Game Theoretic Problem", *Advances in Cryptology*, 2000 (Cited on pages 43, 44, 45, 46, 65, 67, and 70.)

[45] Vincenzo Auletta, "Lezione 4: Equilibri Correlati, Bayesian Games", *Note di "Strumenti della Teoria dei Giochi per l'Informatica" A.A. 2009/10*, `http://libeccio.di.unisa.it/aws/notes/equilibri_correlati.pdf`, 2010 (Cited on pages 45 and 46.)

[46] Katz J., "Bridging game theory and cryptography: recent results and future directions", *Proceedings of the 5th conference on Theory of cryptography*, 2008 (Cited on pages 45, 46, 47, 48, 59, 61, 62, 63, 64, 65, 66, and 68.)

[47] Robert McGrew, Ryan Porter, Yoav Shoham, "Towards a general theory of non-cooperative computation", *Proceedings of the 9th conference on Theoretical aspects of rationality and knowledge*, 2003 (Cited on pages 50, 52, 54, and 55.)

[48] Yoav Shoham, Moshe Tennenholtz, "Non-Cooperative Evaluation of Logical Formulas: The Propositional Case", `https://www.researchgate.net/publication/239666328_Non-Cooperative_Evaluation_of_Logical_Formulas_The_Propositional_Case`, 2003 (Cited on page 53.)

[49] Yoav Shoham, Moshe Tennenholtz, "Non-cooperative computation: Boolean functions with correctness and exclusivity", *Theoretical Computer Science Volume 343, Issues 1-2*, 2005 (Cited on pages 52, 53, and 54.)

[50] "Wikipedia: Parity function", https://en.wikipedia.org/wiki/Parity_function (Cited on page 53.)

[51] "Wikipedia: Majority function", https://en.wikipedia.org/wiki/Majority_function (Cited on page 53.)

[52] Itai Ashlagi, Andrey Klinger, Moshe Tenneholtz, "K-NCC: Stability Against Group Deviations in Non-cooperative Computation", *Internet and Network Economics, Volume 4858 of the series Lecture Notes in Computer Science*, 2007 (Cited on pages 56, 57, and 70.)

[53] "Wikipedia: Symmetric Boolean function", https://en.wikipedia.org/wiki/Symmetric_Boolean_function (Cited on page 57.)

[54] "Wikipedia: Funzione simmetrica", https://it.wikipedia.org/wiki/Funzione_simmetrica (Cited on page 57.)

[55] Elchanan Ben-Porath, "Cheap talk in games with incomplete information", *Journal of Economic Theory vol. 108*, 2003 (Cited on page 61.)

[56] "Wikipedia: Cheap talk", https://en.wikipedia.org/wiki/Cheap_talk (Cited on page 61.)

[57] "GameTeory.net dictionary: Extensive Form", http://www.gametheory.net/dictionary/ExtensiveForm.html (Cited on page 62.)

[58] Ittai Abraham, Danny Dolev, Rica Gonen, Joe Halpern, "Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation", *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, 2006 (Cited on pages 58, 59, 60, and 62.)

[59] I. Damgard, J.B. Nielsen, "Universally Composable Efficient Multiparty Computation from Threshold Homomorphic Encryption", *Advances in Cryptology - CRYPTO*, 2003 (Cited on page 69.)

[60] R. Cramer, I. Damgard, S. Dziembowski, M. Hirt, T. Rabin, "Efficient Multiparty Computations With Dishonest Minority", *Proceedings of EuroCrypt*, 1999 (Cited on page 69.)

[61] R. Cramer, R. Gennaro, B. Schoenmakers, "A Secure and Optimally Efficient Multi-Authority Election Scheme", *Proceedings of EuroCrypt*, 1997 (Cited on page 69.)

[62] I. Damgard, M. Jurik, "A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic", *Public Key Cryptography: Volume 1992 of the series Lecture Notes in Computer Science*, 2001 (Cited on page 69.)

[63] Peter Bogetoft, Dan Lund Christensen, Ivan Damgard, Martin Geisler, Thomas Jakobsen, Mikkel Krigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael Schwartzbach, Tomas Tofty, "Secure Multiparty Computation Goes Live", *Financial Cryptography and Data Security: 13th International Conference*, 2009 (Cited on page 69.)

[64] Claudio Orlandi, "Is multiparty computation any good in practice?", *International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2011 (Cited on page 69.)

RINGRAZIAMENTI

Desidero ringraziare il Professor Michele Boreale per l'aiuto ed il sostegno datomi durante la stesura di questo lavoro di tesi; ringrazio inoltre tutti i docenti del corso di Laurea in Informatica che in questi anni hanno contribuito alla mia crescita.

Ringrazio Elena e Roberto per aver riposto totale fiducia in me, per aver avuto la pazienza di aspettare che trovassi la mia strada, lasciandomi la libertà di farlo in maniera indipendente.

Ringrazio di cuore i miei amici più cari e tutte le persone che mi sono state vicine durante questo percorso formativo.

Infine ringrazio Giuli, mia compagna e sostegno. La ringrazio per aver condiviso con me ore ed ore di studio, ansie e gioie, per avermi sopportato e per avermi aiutato in ogni modo possibile. Per aver sempre stretto la mia mano, per aver sorriso e riso insieme a me, regalandomi serenità.