Rational secret sharing

Zhang, Yun

2012

Zhang, Y. (2012). Rational secret sharing. Doctoral thesis, Nanyang Technological University, Singapore.

https://hdl.handle.net/10356/48667

https://doi.org/10.32657/10356/48667

NANYANG
TECHNOLOGICAL
UNIVERSITY

RATIONAL SECRET SHARING

YUN ZHANG

SCHOOL OF PHYSICAL & MATHEMATICAL SCIENCES

2012

RATIONAL SECRET SHARING

YUN ZHANG

School of Physical & Mathematical Sciences

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2012

ACKNOWLEDGEMENTS

First of all, I would like to show my warm gratitude to my supervisor, Prof. Huaxiong Wang, for his devoted guidance and support. His deep intuition and insightful comments regarding the problems I worked on influenced me a lot. Secondly, many thanks to my co-supervisor, Prof. Guohua Wu, from whose devoted teaching and enlightening lectures I have benefited a lot. Thanks to Prof. Chaoping Xing. I benefit quite a lot from the courses he taught. I benefit even more from his teaching style, which is extremely clear and illuminating. Thanks to Prof. Axel Poschmann, and Prof. Frederique Elise Oggier for giving me many enlightenments and much constructive advice. Besides, thanks to Dr. Christophe Tartary, one of my coauthors, for his generous help when I consult with him about details of research problems. The discussions have benefited me in many ways and also greatly improved my writing. Thanks to Jian Guo and Yeo Sze Ling for their constructive comments on this thesis. Thanks to all the colleagues with whom I shared various discussions during my research studies. Thanks to all my friends, lecturers and also secretaries in the division of mathematical sciences for all the help during the four years. Finally, thanks to the School of Physical and Mathematics Sciences for the funding support for my research assistantship and conference travels.

Papers Related to this Thesis

1. C. Tartary, H. Wang, and Y. Zhang. An efficient and information theoretically secure rational secret sharing scheme based on symmetric bivariate polynomials. *International Journal of Foundations of Computer*, Pages: 1395-1416, 2011. (main contribution)

2. Y. Zhang, C. Tartary, and H. Wang. An efficient rational secret sharing scheme based on the Chinese Remainder Theorem. In *Proceedings of the 16th Astralasian Conference on Information Security and Privacy*, Pages: 259-275, 2011. (main contribution)

3. Y. Zhang, and H. Wang. Transformations from any linear secret sharing scheme to a rational secret sharing scheme. To be submitted. (main contribution)

Notation

$\mathbb{R}$	–	The set of all real numbers
$\mathbb{Z}$	–	The set of all integers
$\mathbb{F}$	–	A finite field
$\mathrm{GF}(q)$	–	The finite field with q elements
P_i	–	The i^{th} player
n	–	The number of players
$\mathcal{P}$	–	The set contains n players $P_1, \ldots, P_n$
$2^{\mathcal{P}}$	–	The power set of $\mathcal{P}$
$\lvert A\rvert$	–	The size of the set A
s	–	A secret in secret sharing schemes
$\mathcal{R}$	–	The domain of random strings
$\mathcal{S}$	–	The domain of the secrets in secret sharing schemes
$\mathcal{S}_i$	–	The share domain of P_i
$\bigoplus$	–	The operation of bitwise XOR
$\langle \alpha, \beta \rangle$	–	The inner product of α and β
$\alpha[i]$	–	The i^{th} coordinate of α
$\mathcal{D}$	–	The dealer in secret sharing schemes
Π	–	A secret sharing scheme
$\Pi(s,r)_i$	–	The share of P_i generated with the secret s and a random string r in the secret sharing scheme Π

$\Pi(s,r)_A$	–	The shares of players in A generated with the secret s and a random string r in the secret sharing scheme Π
$\mathcal{A}$	–	An adversary structure
Γ	–	An access structure
$\mathcal{A}^{\perp}$	–	The dual adversary structure of $\mathcal{A}$
$\Gamma^{\perp}$	–	The dual access structure of Γ
$\mathcal{M}$	–	A monotone span program
$\mathcal{M}^{\perp}$	–	A dual monotone span program of $\mathcal{M}$
LSSS	–	Linear secret sharing scheme
RSSS	–	Rational secret sharing scheme
MSP	–	Monotone span program
MPC	–	Multiparty computation
C	–	A block code
$C^{\perp}$	–	The dual code of C
$\Gamma(C)$	–	The admissible distance of a block code C
$\Delta(C)$	–	The forbidden distance of a block code C
$\mathrm{rank}(M)$	–	The rank of a matrix M
$\mathrm{span}(M)$	–	The vector space spanned by the row vectors in M
M^T	–	The transpose of the matrix M
$ker(M)$	–	The solution space of the linear equation system $M \cdot X = \mathbf{0}$
$(\alpha \vert M)$	–	The matrix containing the columns of M along with an additional column α
(a,β)	–	The vector containing the elements of β along with an additional element a
t	–	The size of each minimal authorized set in a threshold access structure
t^*	–	The size of active players in the secret reconstruction process
k	–	A security parameter
$\mathcal{G}$	–	A game

CONTENTS

ABSTRACT

Cryptography and game theory are both fields dedicated to modeling and facilitating human interactions in real life. In order to produce more robust protocols that allow for a variety of player types and develop more realistic models of, and protocols for, interactions of mutual mistrust, several researchers recently have attempted to bridge the fields of cryptography and game theory. Analyzing the cryptographic protocols using game theoretic methodologies serves to better interpret the behavior of today's systems. One particular problem that has been studied considerably is that of rational secret sharing. Although there are a number of efficient and elegant secret sharing schemes that work in the cryptographic model, it is observed that designing a rational secret sharing protocol which works in the game theoretic model is a little bit challenging.

This thesis contains three main contributions as follows.

First, we observe that most of the previous constructions on rational secret sharing rely on some existing computational-based cryptographic primitives, which make them susceptible to backward induction, since the cryptographic primitives used at the beginning of those protocols can surely be broken after an exponential number of rounds. The first contribution of this thesis is to design a protocol for rational secret sharing that removes this limitation. We borrow the idea from the proactive secret sharing scheme proposed by Arco and Stinson to renew the shares periodically by the interaction between players. In this way, our construction does not need an online dealer. Here the secret s is masked using a one-time pad, which provides information-theoretic security and makes our construction immune to backward induction mentioned previously. Our scheme is based on symmetric bivariate polynomials. Although this technique has already been applied before for multiparty computation protocols, to the best of our knowledge, it is the first time that it has been used in rational cryptography. Our protocol is efficient in terms of round execution (as the dealer will not be needed except during the initial share distribution phase), share size and computations. And it guarantees that all players learn the secret at a Nash equilibrium surviving the iterated elimination of weakly dominated strategies. As in most of the prior work, we need a simultaneous broadcast channel and pairwise secure channels. However, our contribution does not address the scenario where coalitions of multiple players are present.

The second contribution of this thesis is the proposal of an efficient protocol for t-out-of-n rational secret sharing based on the Chinese Remainder Theorem. Under some computational assumptions related to the discrete logarithm problem and RSA, this construction leads to a $(t-1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles. Our protocol does not rely on simultaneous channels. Instead, it only requires a synchronous broadcast channel and pairwise secure channels. As compared to the protocol proposed by Fuchsbauer *et al.* that works in almost the same model as ours, our protocol has smaller share size even when $(t-1)$-resilience to coalitions is required. Our shares are $O(k)$ bits long while theirs need $(n-t+1)\,(2n|s|+O(k))$ bits. The latter share length leads to practical efficiency issues when $n-t+1$ is large or when their technique is used as a building block within more general rational multiparty computation protocols.

It is clear that every rational secret sharing scheme is itself a classical secret sharing scheme, but the converse is not true. The third line of research in this thesis is to demonstrate transformations from any (classical) linear secret sharing scheme to a rational secret sharing scheme with a mediator. The rational secret sharing scheme obtained induces a Nash equilibrium surviving iterated deletion of weakly dominated strategies with resilience to any subset in the adversary structure, relies on no cryptographic assumption and provides information-theoretic security. The first transformation works for any linear secret sharing scheme with an arbitrary adversary structure and the resulting rational secret sharing scheme has expected round complexity relying on participants' utilities. The second transformation works for any linear secret sharing scheme with a Q^2 adversary structure and the resulting rational secret sharing scheme has expected round complexity $O(1)$. Both of these two transformations require knowledge of the participants' utilities. The third transformation works for any linear secret sharing scheme with a Q^3 adversary structure but requires no knowledge of participants' utilities. The rational secret sharing scheme obtained has expected round complexity $O(1)$. While the first transformation requires pairwise secure channels along with a simultaneous broadcast channel, the second and the last transformations only require pairwise secure channels. Moreover, if the underlying linear secret sharing scheme is $\mathcal{A}$-detectable, the rational secret sharing schemes obtained from the second and the last transformations induce a strict Nash equilibrium. To the best of our knowledge, this research marks the first instance that the problem on transformation from any linear secret sharing scheme to a rational secret sharing scheme is considered, and the first time that rational secret sharing schemes for non-threshold access structures are constructed.

1. INTRODUCTION

Cryptography and game theory are both fields dedicated to modeling and facilitating human interactions in real life. Traditional cryptographic models assume that some participants are honest (that is, they faithfully follow a given protocol) while others are malicious participants (that is, they act in an arbitrary manner) against whom the honest players must be protected. In that case, security assurance relies on the assumption that a fraction of the agents follow the protocol specifications accurately, regardless of whether doing so aligns with their own self-interest. However, in many real-world applications, a participant may choose to be dishonest if deviating from the protocol will provide him with some advantage. Game theory can be used to model such a situation where players are *self-interested* (i.e., *rational*), namely, they do everything to maximize their benefit (or utilities). In order to produce more robust protocols that allow for a variety of player types, *rational cryptography* arises. It bridges the fields of cryptography and game theory and desires to develop more realistic models of, and protocols for, interactions of mutual mistrust. The rational analysis of the cryptographic protocols serves to better interpret the behavior of today's systems, which operate in strategic interaction environments.

One particular problem that has been studied considerably is that of rational secret sharing. Informally, (classical) secret sharing is a cryptographic primitive, which allows a dealer to distribute a secret among a set of finitely many players in a way that each authorized subset can later recover the secret by combining its members' shares while any unauthorized subset can not. The standard way to execute the reconstruction is simply for each player in some authorized set to broadcast his share to all others (in this authorized set). In particular, Shamir's t-out-of-n secret sharing scheme [56] specifies a method to split a secret into n values of some polynomial of degree at most $t-1$ among n players in a way that any set containing at least t players can recover the secret by polynomial interpolation, while any set containing less than t players gets no information about the secret. We need to emphasize that the goal of the plain secret sharing schemes may not be achieved without the assumption that the dealer and the players are honest. In the rational setting, in order to instruct each rational player to broadcast his real share during the reconstruction process, first it is assumed that each player prefers the outcome in which he learns the secret. This is a necessary assumption, since without it each rational player has no interest to be involved in the protocol (or game) at all. Secondly, it is assumed that each player prefers the outcome in which as few others as possible learn the secret. Indeed, this assumption is reasonable, especially when knowledge is power, and it reflects players' selfishness. Note that the utilities of the players depend *only* on who learns and who does not learn the secret. We would like to stress that in the rational model, under these two assumptions on players' preferences on outcomes, classical secret sharing schemes will fail completely, since broadcasting one's share during the secret reconstruction phase is *not* rational. To understand this, we might take Shamir's t-out-of-n secret sharing scheme as an example. Suppose that t players $P_{i_1}, \ldots, P_{i_t}$ are involved in the reconstruction phase and they are required to broadcast their shares *simultaneously.* For each $P_{i_j}, 1 \leq j \leq t$, there are two possibilities. Case 1: all the others broadcast their shares. Then, if P_{i_j} keeps silent, he will be the only one that learns the secret; while if he broadcasts his share, he will learn the secret with the others. Case 2: some of the others do not broadcast their shares. Then no matter what P_{i_j} does, no one will learn the secret. Obviously, in either cases, not broadcasting one's share during the secret reconstruction phase can never bring less to him and sometimes can bring more to him no matter what the others do, that is, the rational behavior in the above naive reconstruction procedure is for each player not to broadcast his share. As for the scenario where participants are creating coalitions with size at most $t-1$ against other honest participants, exactly by the same argument one can see that it is rational for each coalition player not to broadcast his share.

This fact motivates the notion of rational secret sharing, which was first proposed by Halpern and Teague in 2004 [31]. Here, the aim is to construct a protocol so that participants have incentive to cooperate and provide their shares in the reconstruction phase, even if each participant prefers to be the only one that learns the secret. In other words, we need to design a strategy for each player during the secret reconstruction phase such that under those two assumptions on preferences on outcomes mentioned previously, each rational player has an interest to follow his strategy no matter what other players do, and when all of them follow their strategies, the secret is open to all. However, due to players' conflicting self interests, this requirement is so stringent that it is pretty hard to be satisfied. Thus, in the literature of rational secret sharing, the main goal is to design protocols for participants such that on one hand, every participant has motivation not to deviate from his strategy, provided that all the other participants stick to theirs, and on the other hand, if each participant sticks to his strategy, the secret will be revealed to all participants. To retell it in game-theoretic terminology [49], designing rational secret sharing protocols essentially amounts to designing a recommended strategy for each player along with the dealer (who is assumed to be honest). Each player's strategy set is defined to be all possible deviations from his recommended strategy. Players' utility functions are supposed to satisfy those two assumptions on preferences on outcomes mentioned previously. The goal is that the tuple of recommended strategies induces a Nash equilibrium or one of its variants, in which each player learns the secret. There are several criteria to measure a rational protocol, such as channel models, coalition-resilience abilities, equilibrium types achieved, utility dependence and efficiencies (including expected round complexity, computational complexity and share bitsize).

In Chapter 2, we will provide an overview on classical secret sharing, including related definitions along with some general constructions for arbitrary access structures. In particular, we focus on linear secret sharing schemes, monotone span programs and linear block codes. It is well known that the first two objects are closely related. Besides this fact, Nikov and Nikova [45] further explored the properties of a monotone span program and the corresponding linear block code C. Briefly, let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be a monotone span program computing an access structure Γ, where without loss of generality, the columns of M are assumed to be linearly independent. Let C be the linear code generated by M^T, the transpose of M. Let M_{P_i} denote the matrix whose rows are those assigned to player P_i via ψ. Without loss of generality, we can assume that $M^T = ((M_{P_1})^T, \ldots, (M_{P_n})^T)$. Let C be the linear code generated by M^T. Note that here we can partition the entries of each codeword of C into n blocks in the same way as we split M^T into n blocks $((M_{P_1})^T, \ldots, (M_{P_n})^T)$. Thus, we can index the blocks of the resulting vector by $\{P_1, \ldots, P_n\}$. In this sense, C can be considered as a block code and each codeword X can be written uniquely as $X = (X_{P_1}, \ldots, X_{P_n})$. Let $\Gamma(C) := \{B|$ there exists a nonzero codeword $X \in C$ such that $\text{supp}(X) \subseteq B\}$, where $\text{supp}(X) := \{P_i : X_{P_i} \neq \mathbf{0}\}$. Let $\Delta(C) := 2^{\mathcal{P}} \setminus \Gamma(C)$. $\Gamma(C)$ is called the set of admissible distances of C and $\Delta(C)$ is called the set of forbidden distances of C. Let E be an error vector. $\text{supp}(E)$ is called the error pattern of E. They demonstrated that C can correct each error pattern $A \in \mathcal{A}$ if and only if C satisfies the condition $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$, where $\mathcal{A}$ denotes the adversary structure and $\mathcal{A} \uplus \mathcal{A} := \{B_1 \cup B_2 | B_1, B_2 \in \mathcal{A}\}$. However, it is hard to check directly from $\mathcal{M}$ whether the condition $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ is satisfied or not. In Chapter 3, we attempt to go further in this direction by proposing an equivalent condition for the inclusion relation $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ from the perspective of the matrix $M^{\perp}$, whose first column is a solution vector of $M^T \cdot X = \mathbf{0}$, and the remaining columns form a basis of the space $ker(M^T)$. Besides, we demonstrate that C can detect all error patterns in $\mathcal{A}$ if and only if $\mathcal{A} \subseteq \Delta(C)$ and give an equivalent condition from the perspective of $M^{\perp}$.

In Chapter 4, we introduce basic notions and definitions from game theory [27, 49] that will be needed to develop and understand rational cryptography. More explicitly, we introduce two types of games: normal form games and extensive games with perfect information, along with some solution concepts: Nash equilibrium and its variants.

In Chapter 5, we present a brief survey on rational secret sharing. Specifically, we summarize the main ideas behind rational secret sharing schemes and the common approach used to design them. Furthermore, we provide comparisons among these protocols based on their channel models, coalition-resilience abilities, equilibrium types achieved, utility dependence and efficiency (including expected round complexity, computational complexity and share bitsize).

In Chapter 6, we propose a protocol for threshold rational secret sharing. Unlike most of the previous protocols, our construction neither requires an online dealer or any trusted parties (the mediator for example), nor relies on secure multiparty computation to redistribute the shares of the secret. Instead, we borrow the idea from proactive secret sharing schemes [20] to renew the shares merely by the interactions between players. A notable difference in our construction as compared to the constructions quoted in Chapter 5 is that on one hand, the secret s is masked by a one-time pad, and on the other hand, the participants may correct a single error using the decoder for generalized Reed-Solomon codes. This provides information-theoretic security and makes our construction immune to backward induction. Our scheme is based on symmetric bivariate polynomials. Although this technique has been employed previously for multiparty computation protocols [17], to the best of our knowledge, it is the first time that it has been used in rational cryptography. Our protocol is efficient in terms of round execution (as the dealer will not be needed), share size and computation and it guarantees that all players learn the secret at a Nash equilibrium whose strategy survives the iterated elimination of weakly dominated strategies. As in most of the prior work, we need a simultaneous broadcast channel and pairwise secure channels.

At TCC'10 [27], Fuchsbauer *et al.* introduced two variants of Nash equilibrium: computational version of strict Nash equilibrium, which offers a computational relaxation of traditional game theory equilibria, and stability with respect to trembles, which models the tiny uncertainty on the expectation about the other players' actions. Using trapdoor permutations, they constructed a t-out-of-n rational secret sharing scheme satisfying these new security models. Their construction only requires standard communication networks, but the share bitsize is $2n|s| + O(k)$ for security against a single player's deviation, and raises to $(n-t+1)\cdot(2n|s|+O(k))$ to achieve $(t-1)$-resilience, where k is a security parameter and $|s|$ denotes the bit length of s. This leads to practical efficiency issues when $n - t + 1$ is large or when Fuchsbauer *et al.*'s technique is used as a building block within more general rational multiparty computation protocols.

In Chapter 7, we propose an efficient protocol for rational t-out-of-n secret sharing based on the Chinese Remainder Theorem with share bitsize $O(k)$. Under some computational assumptions related to the discrete logarithm problem and RSA, this construction leads to a $(t - 1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles. Our protocol does not rely on simultaneous channels. Instead, it only requires a synchronous broadcast channel and pairwise secure channels.

It is well-known that for any given monotone access structure Γ, there always exists a linear secret sharing scheme (LSSS for short) realizing it. Besides, any rational secret sharing scheme (RSSS for short) itself is a classical secret sharing scheme, but the converse is not true. Hence, it is interesting to consider how to transform a classical linear secret sharing scheme to a rational secret sharing scheme. A useful way to realize this transformation is to introduce a mediator, who is a trusted third party. Following Forges [26], we may view a mediator as a communication device: each participant sends a message (input) and the mediator computes a function of all the messages ever obtained and sends each participant some information (output).

In Chapter 8, we propose three transformations from a classical linear secret sharing scheme to a rational secret sharing scheme with a mediator. The RSSS obtained induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies, relies on no cryptographic assumption, provides information theoretical security and hence is automatically immune against backward induction. The first transformation works for any LSSS with an arbitrary adversary structure $\mathcal{A}$ and the resulting RSSS has expected round complexity relying on the participants' utilities. The second transformation works for any LSSS with a Q^2 adversary structure $\mathcal{A}$ and the resulting RSSS has expected round complexity $O(1)$. Both of these two transformations require that knowledge of the participants' utilities be public. The third transformation works for any

LSSS with a Q^3 adversary structure, requires no knowledge of participants' utilities, and the RSSS obtained has expected round complexity $O(1)$. While the first transformation requires pairwise secure channels along with a simultaneous broadcast channel, the second and the last transformations only require pairwise secure channels. Moreover, if the underlying linear secret sharing scheme is $\mathcal{A}$-detectable, the rational secret sharing schemes obtained from the second and the last transformations induce a strict Nash equilibrium. To the best of our knowledge, this research marks the first instance that the problem on transformations from any LSSS to an RSSS is considered and the first time that RSSSs for non-threshold access structures are constructed. As an independent interest, our constructions generalize and improve some of the results proposed by Abraham *et al.* in [1], where they proposed RSSSs over t-out-of-n threshold access structures, which induce a $(t-1)$-resilient Nash equilibrium surviving iterated elimination of weakly dominated strategies. In addition, their constructions make use of digital signature or Rabin's information checking protocol [52] in the initial share distribution phase while ours do not need any verification data in the initial share distribution phase.

Finally, we conclude this thesis with a general conclusion and some open problems in Chapter 9.

2. CLASSICAL SECRET SHARING

The proposals of secret sharing can be dated back to 1979 and they were motivated as a response to the problem of secure information storage. Briefly, secret sharing is a method by which a dealer splits a secret into several shares and distributes them among players in a way that each authorized subset can later reconstruct the secret by combining its members' shares, while each unauthorized set can not. Ever since it was proposed by Shamir [56] and Blakley [9] independently in 1979, secret sharing has been a major building block for cryptographic primitives, particularly in the area of distributed computing, such as *multiparty computation* (MPC for short) [6,14,18], *Byzantine agreement* [51], *threshold cryptography* [21], *access control* [44], *attribute-based encryption* [30] and *generalized oblivious transfer* [59].

2.1 Communication Channels

In real life, people have many ways to communicate with one another, such as face to face, on the phone, by email or through letters, each of which has different properties. However, in cryptographic applications, we can only model the various means of communication by networks. Hereinafter, all the channels are assumed to be secure, that is, they provides privacy and authentication. In this section, we give informal definitions [55] for different communication channels, which enable us to discuss and compare the relative strengths and weaknesses of them.

- A *broadcast channel* is a network among n participants, through which each participant P_i can send a message in a way that each other player will eventually receive the same message as the one sent by P_i. Intuitively, the broadcast channel models a player standing in a room together with all the other players and making an announcement.
- A *simultaneous broadcast channel* is a network among n participants, through which all participants can send messages in a way that for each $1 \leq i \leq n$, players other than P_i will receive the same message as the one sent by P_i. Moreover, all other players know this message

is from P_i and all the messages are received simultaneously. Here by simultaneously, we mean that all participants can simultaneously receive messages and so no participant can see what the others broadcast before sending his own message. Intuitively, this channel models each participant independently putting his message into an envelope marked with his name and a trusted party opening all of the envelopes for all participants.

- A *synchronous broadcast channel* is a broadcast channel among n participants, who are synchronized with respect to a global clock. It does not require participants receive messages simultaneously. Instead, each participant is allowed to broadcast his message after he has received some messages from other participants. Intuitively, this channel models each participant putting his message into an envelope marked with his name, possibly after observing some messages from the other participants and then a trusted party opening his envelope to all participants.

- A *secure channel between two players* P and P' is a channel, through which P (P' resp.) can send a message to P' (P resp.) in a way that P' (P resp.) will eventually receive the same message as the one sent by P (P' resp.) Intuitively, this channel models a player P (P' resp.) writing his message on a note and passing it to P' (P resp.) out of sight of all other players.

- An *envelope channel* allows each player P to seal a secret message into a specially marked envelope in plain view and pass it across the table to another player P' in plain view. Later, P' can open this envelope in such a way that everyone knows he opens it and only he can see the data inside. Obviously, an envelope channel can be implemented only if all the players rendezvous. However, in most cases, this requirement can not be satisfied.

We say that communication channel A is at least as strong as communication channel B if A can be used to simulate B. The stronger a channel is, the more difficult it can be implemented in practice. Clearly, the hierarchy of those channels related to strength is: an envelope channel $>$ a simultaneous broadcast channel $>$ a synchronous broadcast channel $>$ pairwise secure channels.

2.2 Basic Definitions on Secret Sharing

Suppose that n participants $P_1, \ldots, P_n$ want to share a secret key in a way that only specific subsets of these participants can reconstruct the secret. Each of these subsets of participants is called an *authorized subset* and the set of all authorized subsets is called an *access structure*, which is denoted by Γ. It is easy to understand that if a set A of participants can recover the secret, then any superset of A can also recover the secret, namely, if $A \in \Gamma$ and $B \supseteq A$, then $B \in \Gamma$. Conversely, any collection of subsets satisfying the above property can be an access structure. Formally,

Definition 2.2.1. Let $\mathcal{P} := \{P_1, \ldots, P_n\}$ be a finite set of participants. A collection $\Gamma \subseteq 2^{\mathcal{P}}$ is *monotone increasing* if $B \in \Gamma$ whenever $A \subseteq B$ for some $A \in \Gamma$. An *access structure* is a monotone increasing nonempty collection $\Gamma(\subseteq 2^{\mathcal{P}})$ of non-empty subsets of $\mathcal{P}$. Each set in Γ is called an authorized subset and any set not in Γ is called an unauthorized subset. $\mathcal{A} := 2^{\mathcal{P}} \setminus \Gamma$ is called the adversary structure (corresponding to Γ). The *minimal access structure* $\Gamma_{\min}$ of Γ is defined to be the set $\{A \in \Gamma : \text{any proper subset of } A \text{ is not in } \Gamma\}$. Clearly, Γ uniquely determines $\Gamma_{\min}$ and vice versa. $\mathcal{A}$ is Q^2 if $\mathcal{P} \setminus A \in \Gamma$ for any $A \in \mathcal{A}$; $\mathcal{A}$ is Q^3 if $\mathcal{P} \setminus (A \cup B) \in \Gamma$ for any $A, B \in \mathcal{A}$. Given an access structure Γ, $\Gamma^{\perp} := \{A \subseteq \mathcal{P} | A^c \notin \Gamma\}$, where A^c denotes the complement of A, is called the dual access structure of Γ and the corresponding adversary structure is denoted by $\mathcal{A}^{\perp}$, which equals $2^{\mathcal{P}} \setminus \Gamma^{\perp}$.

Example 2.2.2. For any $1 \leq t \leq n$, let $\Gamma := \{A \subseteq \mathcal{P} : |A| \geq t\}$. Γ is called the t-out-of-n *threshold access structure*. The corresponding minimal access structure is $\Gamma_{\min} = \{A \subseteq \mathcal{P} : |A| = t\}$ and the adversary structure is $\mathcal{A} = \{A \subseteq \mathcal{P} : |A| < t\}$. Obviously, $\mathcal{A}$ is Q^2 if and only if $n \geq 2t-1$, and $\mathcal{A}$ is Q^3 if and only if $n \geq 3t-1$. Besides, it is not difficult to check that $\Gamma^{\perp} = \{A \subseteq \mathcal{P} : |A| \geq n-t+1\}$ and $\mathcal{A}^{\perp} = \{A \subseteq \mathcal{P} : |A| \leq n-t\}$.

Definition 2.2.3. [5] Let $\mathcal{S}$ be a finite set of secrets, where $|\mathcal{S}| \geq 2$. A *perfect secret sharing scheme* with domain of secrets $\mathcal{S}$ is specified by a pair $\Sigma = (\Pi, \mu)$, where μ is a probability distribution on some finite set $\mathcal{R}$ (called the set of random strings) and Π, which is called a distribution function, is a mapping from $\mathcal{S} \times \mathcal{R}$ to a set of n-tuples $\times_{i=1}^{n} \mathcal{S}_i$. Here $\mathcal{S}_i$ is called the share domain of participant P_i, $1 \leq i \leq n$. (In most of the cases, μ is the uniform distribution and can be omitted.) A secret $s \in \mathcal{S}$ is shared among n participants of $\mathcal{P}$ according to Σ by first sampling a random string $r \in \mathcal{R}$ via μ, computing a vector of shares $\Pi(s, r) = (s_1, \ldots, s_n) \in \times_{i=1}^{n} \mathcal{S}_i$ and then sending the share s_i to participant P_i secretly for $1 \leq i \leq n$. We say that Σ realizes an access structure Γ if the following two requirements are satisfied:

Correctness. Each authorized subset can reconstruct the secret s. Formally, for any $A \in \Gamma$, say $A = \{P_{i_1}, \ldots, P_{i_{|A|}}\}$, there exists a reconstruction function $\textsc{Recon}_A : \mathcal{S}_{i_1} \times \cdots \times \mathcal{S}_{i_{|A|}} \longrightarrow \mathcal{S}$ such that for every $s \in \mathcal{S}$ and for every $r \in \mathcal{R}$ chosen according to the distribution μ, the following equation holds:

$$\textsc{Recon}_A(\Pi(s,r)_A) = s,$$

where $\Pi(s,r)_A$ denotes the restriction of $\Pi(s,r)$ on A. [1]

Privacy. Each unauthorized subset learns nothing about the secret (in the information-theoretic sense) from their shares. Formally, for any unauthorized subset $B \notin \Gamma$, for every two secrets s and s' in $\mathcal{S}$, and for every possible $|B|$-tuple of shares $(s_i)_{P_i \in B}$, it holds that

$$\Pr[\Pi(s,r)_B = (s_i)_{P_i \in B}] = \Pr[\Pi(s',r)_B = (s_i)_{P_i \in B}].$$

For simplicity, we can assume that $\sum$ is executed by a trusted third party $\mathcal{D}$, called the dealer, who also sends each share s_i to P_i through a secure channel that provides privacy and authentication. Definition 2.2.3 requires correctness with probability 1 and perfect privacy without relying on any computational assumption. Hence it is called a *perfect* secret sharing scheme.

Example 2.2.4. Shamir's t-out-of-n Secret Sharing Scheme [56]

Let $\Gamma = \{A \subseteq \mathcal{P} : |A| \geq t\}$ be a threshold access structure. Shamir [56] constructed a simple and elegant scheme realizing Γ. In this scheme, the domain of the secrets is a finite field $\mathbb{F} = \mathrm{GF}(q)$ for some prime power $q > n$. To share a secret $s \in \mathbb{F}$, the dealer chooses n distinct non-zero elements $\alpha_1, \ldots, \alpha_n$ from $\mathbb{F}$, which are known to all parties. Then he chooses $t-1$ random elements $a_1, \ldots, a_{t-1}$ uniformly and independently from $\mathbb{F}$. Let $f(x) = s + a_1 x + \cdots + a_{t-1} x^{t-1}$ be a polynomial with degree at most $t-1$. Let $\mathcal{R} = \mathbb{F}^{t-1}$, $\mathcal{S}_1 = \cdots = \mathcal{S}_n = \mathbb{F}$. Then the distribution function Π is defined as follows.

$$\Pi : \mathcal{S} \times \mathcal{R} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$$

which maps $(s, a_1, \ldots, a_{t-1})$ to $(f(\alpha_1), \ldots, f(\alpha_n))$. The share for player P_i is $s_i = f(\alpha_i)$.

Roughly speaking, the correctness and privacy conditions of Shamir's scheme follow from the

[1] Hereinafter, $\Pi(s,r)_A$ denotes the shares of player P_i in A generated via Π with the secret s and a random string r. $\Pi(s,r)_i$ denotes the shares of player P_i generated via Π with the secret s and a random string r.

Lagrange's interpolation theorem: For every finite field, any t distinct field elements $x_1, \ldots, x_t$, and any t field elements $y_1, \ldots, y_t$, there exists a unique polynomial Q of degree at most $t-1$ over this field such that $Q(x_i) = y_i$ for $1 \leq i \leq t$.

To verify the correctness condition of Shamir's scheme formally, notice that every subset A of size t holds t points of the polynomial $f(x)$ and hence $f(x)$ can be reconstructed by using Lagrange's interpolation, from which the secret $s = f(0)$ can be recovered. More explicitly, a set $A = \{P_{i_1}, \ldots, P_{i_t}\}$ computes a polynomial

$$g(x) = \sum_{l=1}^{t} s_{i_l} \prod_{1 \leq j \leq t, j \neq l} \frac{\alpha_{i_j} - x}{\alpha_{i_j} - \alpha_{i_l}}.$$

Notice that $g(\alpha_{i_l}) = f(\alpha_{i_l})$ for $1 \leq l \leq t$, namely, f and g are polynomials of degree at most $t-1$ that agree on t points. As a result, $f = g$ and in particular, $f(0) = g(0) = s$. In this way, the participants in A can recover s by computing

$$s = g(0) = \sum_{l=1}^{t} s_{i_l} \prod_{1 \leq j \leq t, j \neq l} \frac{\alpha_{i_j}}{\alpha_{i_j} - \alpha_{i_l}}.$$

Observe that s is the linear combination of the shares of the participants in A, for any $A \in \Gamma_{\min}$, where the coefficients depend only on the set A but are independent of the secret s and $(a_1, \ldots, a_{t-1})$, the random string chosen for distributing shares for s.

Now we proceed to demonstrate the privacy condition. It suffices to show that any unauthorized set B of size $t-1$ gets no information about the secret s. However, since any unauthorized set $B = \{P_{i_1}, \ldots, P_{i_{t-1}}\}$ of size $t-1$ together with every possible secret $a \in \mathbb{F}$ (a can be regarded as a value of some polynomial evaluated at the point 0) determines a unique polynomial f_a of degree at most $t-1$, we have

$$\Pr[\Pi(a, r)_B = (s_{i_l})_{1 \leq l \leq t-1}] = \frac{1}{q^{t-1}}.$$

In other words, the probability is the same for every $a \in \mathbb{F}$, which completes the proof.

In Definition 2.2.3, we require correctness with probability 1 and perfect privacy: for every two secrets s and s', for any unauthorized subset B, the distributions of $\Pi(s, r)_B$ and $\Pi(s', r)_B$ are identical. It is well known that in a perfect secret sharing scheme, the bitsize of each share is at least that of the secret, namely, $\log|\mathcal{S}_i| \geq \log|\mathcal{S}|$ for $1 \leq i \leq n$. However, in practice, it may be desirable for the size of the shares to be small, since it is more difficult and more expensive for participants to store shares with larger bitsize. Hence, it makes sense to relax these requirements by requiring that the correctness holds with high probability and that the statistical distance between $\Pi(s,r)_B$ and $\Pi(s',r)_B$ is sufficiently small.

Let I be an index set and X_i, $i \in I$, be a random variable. $\{X_i\}_{i\in I}$ is called a *sequence of random variables.* In most of the cases that we consider, $I = \mathbb{N}$, the set of natural numbers and the corresponding sequence of random variables is $\{X_i\}_{i\in\mathbb{N}}$, where X_i is a random variable on $\{0,1\}^{poly(i)}$ for some specific polynomial *poly.*

Definition 2.2.5. Let $\epsilon : \mathbb{N} \to [0,\infty)$ be a function. We say ϵ is *negligible* if for every positive polynomial $p(\cdot)$, there exists an integer $N_{p(\cdot)} > 0$ such that for all $l > N_{p(\cdot)}$, it holds that $\epsilon(l) < \frac{1}{p(l)}$. We say that ϵ is *noticeable* if there exists a positive polynomial $p(\cdot)$ and an integer $M_{p(\cdot)}$ such that $\epsilon(l) > \frac{1}{p(l)}$ for any $l > M_{p(\cdot)}$.

Definition 2.2.6. Let $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ be two sequences of random variables. $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ are said to be *perfectly indistinguishable* (i.e., *identical*) if for each $i \in \mathbb{N}$ and for each $a_i \in \{0,1\}^{poly(i)}$, it holds that $\Pr[X_i = a_i] = \Pr[Y_i = a_i]$.

Definition 2.2.7. Let $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ be two sequences of random variables. $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ are said to be *statistically indistinguishable* if

$$d(i) := \frac{1}{2} \sum_{a_i \in \{0,1\}^{poly(i)}} |\Pr[X_i = a_i] - \Pr[Y_i = a_i]|$$

is a negligible function.

Definition 2.2.8. Let $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ be two sequences of random variables. $\{X_i\}_{i\in\mathbb{N}}$ and $\{Y_i\}_{i\in\mathbb{N}}$ are said to be *computationally indistinguishable* (i.e., *polynomial time indistinguishable*) if for any probabilistic polynomial time distinguisher T,

$$t(i) := |\Pr[T(X_i) = 1] - \Pr[T(Y_i) = 1]|$$

is a negligible function. Here $t(i)$ represents the probability that T can distinguish X_i from Y_i.

In Definition 2.2.3, the privacy condition requires that $\{\Pi(s,r)_B\}$ and $\{\Pi(s',r)_B\}$ are perfectly indistinguishable, for any s, s' in the domain of the secrets and for any $B \notin \Gamma$. Observe that the sequences of random variables $\{\Pi(s,r)_B\}$ and $\{\Pi(s',r)_B\}$ may vary when the length of the secret or the number of participants varies. Hence, for simplicity, here we insert a security parameter k, which may depend on the domain of the secrets and the number of participants, and the corresponding sequence of random variables is denoted by $\{\Pi(s,r,k)_B\}$. W are now well prepared to give the following definitions.

Definition 2.2.9. Given an access structure Γ, a *perfect secret sharing scheme* realizing Γ is specified by a distribution function $\Pi : \mathcal{S} \times \mathcal{R} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ satisfying the following conditions:

- *Correctness.* $\forall A \in \Gamma$, $\forall k \in \mathbb{N}$, there exists a set of reconstruction functions $\{\textsc{Recon}_A : (\mathcal{S}_1 \times \cdots \times \mathcal{S}_n)|_A \to \mathcal{S}\}$ such that for every $s \in \mathcal{S}$,

$$\Pr[\textsc{Recon}_A(\Pi(s,r,k)_A) = s] = 1.$$

- *Perfect Privacy.* For any $B \notin \Gamma$, for every two secrets $s, s' \in \mathcal{S}$, $\Pi(s,r,k)_B$ and $\Pi(s',r,k)_B$ are perfectly indistinguishable.

Definition 2.2.10. Given an access structure Γ, a *statistical secret sharing scheme realizing* Γ is specified by a distribution function $\Pi : \mathcal{S} \times \mathcal{R} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ satisfying the following conditions:

- *Correctness.* $\forall A \in \Gamma$, $\forall k \in \mathbb{N}$, there exists a set of reconstruction functions $\{\textsc{Recon}_A : (\mathcal{S}_1 \times \cdots \times \mathcal{S}_n)|_A \to \mathcal{S}\}$ such that for every $s \in \mathcal{S}$,

$$\Pr[\textsc{Recon}_A(\Pi(s,r,k)_A) = s] > 1 - \text{negl}(k),$$

  where negl$(\cdot)$ denotes a negligible function.

- *Privacy.* For any $B \notin \Gamma$, for every two secrets $s, s' \in \mathcal{S}$, $\Pi(s,r,k)_B$ and $\Pi(s',r,k)_B$ are statistically indistinguishable.

Definition 2.2.11. Given an access structure Γ, a *computational secret sharing scheme* realizing Γ is specified by a distribution function $\Pi : \mathcal{S} \times \mathcal{R} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ satisfying the following conditions:

- *Correctness.* $\forall A \in \Gamma$, $\forall k \in \mathbb{N}$, there exists a set of probabilistic polynomial time computable reconstruction functions $\{\textsc{Recon}_A : (\mathcal{S}_1 \times \cdots \times \mathcal{S}_n)|_A \to \mathcal{S}\}$ such that for every $s \in \mathcal{S}$,

$$\Pr[\textsc{Recon}_A(\Pi(s,r,k)_A) = s] > 1 - \text{negl}(k),$$

where negl$(\cdot)$ denotes a negligible function.

- *Privacy.* For any $B \notin \Gamma$, for every two secrets $s, s' \in \mathcal{S}$, $\Pi(s, r, k)_B$ and $\Pi(s', r, k)_B$ are computationally indistinguishable.

It is worth noticing that the privacy requirements in the three types of secret sharing schemes defined above differ. With regard to the correctness condition, we emphasize that the reconstruction functions for a computational secret sharing scheme should be computationally efficient compared with perfect secret sharing schemes, and the correctness requirement for statistical (or computational) secret sharing schemes is relaxed by allowing a negligible error probability. However, in practice it is always required that the reconstruction functions be computationally efficient. The main advantage of computational secret sharing schemes over perfect secret sharing schemes is that in the former, it is likely that $\log|\mathcal{S}_i|$ is strictly less than $\log|\mathcal{S}|$, that is, the share bitsize can be less than the secret bitsize. This property is critical for some practical applications. Schemes that satisfy these relaxed requirements are designed in [4, 8, 39].

Example 2.2.12. [3] The Asmuth-Bloom secret sharing scheme based on the Chinese Remainder Theorem. Here Γ is a t-out-of-n threshold access structure.

- Share distribution phase:
  - To share a secret s, the dealer $\mathcal{D}$ chooses a set of pairwise relatively prime integers $p_0 < p_1 < \cdots < p_n$ of bit length at least k, where k is a security parameter, $s < p_0$ and
  $$\prod_{i=1}^{t} p_i > p_0 \prod_{i=1}^{t-1} p_{n-i+1}.$$
  - $\mathcal{D}$ computes $y = s + a \cdot p_0$, where a is a positive integer generated randomly subject to the condition that
  $$0 \leq y < \prod_{i=1}^{t} p_i.$$
  - The share for P_i is $y_i := y \bmod p_i$.
- Secret reconstruction phase: any group of t players can recover s by using the Chinese Remainder Theorem to solve a linear equation system consisting of t congruence expressions.

As pointed out in [3], the correctness of the preceding scheme holds with probability 1; however, the privacy is not perfect, since from the perspective of any $t-1$ players, for any s', s'' from the domain of the secret, $\Pr[s = s']$ and $\Pr[s = s'']$ asymptotically equal. Indeed, a larger value of p_0 will inevitably lead to a smaller difference between these two probabilities and this difference approaches to zero when p_0 approaches to infinity. More precisely, $|\Pr[s = s'] - \Pr[s = s'']| = O(\frac{1}{p_0^2})$.

2.3 Constructions of Secret Sharing Schemes for General Access Structures

2.3.1 Ito, Saito and Nishizeki's Construction

Ito, Saito, and Nishizeki [33] defined secret sharing schemes for general access structures and proposed a general construction for any monotone access structure. Let Γ be any monotone access structure. The dealer shares the secret independently among each minimal authorized set $A \in \Gamma$. More explicitly, to share a secret $s \in \{0, 1\}$, for every $A = \{P_{i_1}, \ldots, P_{i_l}\} \in \Gamma_{\min}$, the dealer

- chooses $l - 1$ bits $r_1, \ldots, r_{l-1}$ uniformly at random;
- computes $r_l = s \bigoplus r_1 \bigoplus \cdots \bigoplus r_{l-1}$, and
- gives r_j to p_{i_j} secretly.

Note that for each $A \in \Gamma$, the random bits are chosen independently. It is easy to check the correctness and privacy conditions. We would like to emphasize that the number of bits that P_j gets is the number of minimal authorized sets that contain P_j. As a consequence, the scheme is highly inefficient for access structures in which the number of minimal authorized sets is big. Take the t-out-of-n threshold access structure as an example. The number of bits that each participant gets is $\binom{n}{t-1}$, which is $\Theta(2^n/\sqrt{n})$ when $t = \frac{n}{2}$. It is far larger than the number of bits each participant receives in Shamir's secret sharing scheme, which is the same as the size of the secret.

2.3.2 The Monotone Formulae Construction

Benaloh and Leichter [7] proposed a construction of secret sharing schemes using monotone formulae, which generalizes the construction in [33].

The basic idea behind their construction is to use a recursive approach: it begins with schemes for simple access structures, from which a scheme for a composition of those simple access structures is obtained. More explicitly, let Γ_1 and Γ_2 be two access structures on the same set of participants $\mathcal{P} = \{P_1, \ldots, P_n\}$, from which two new access structures $\Gamma_1 \vee \Gamma_2$ and $\Gamma_1 \wedge \Gamma_2$ are defined as follows: $\Gamma_1 \vee \Gamma_2 := \{A | A \in \Gamma_1 \text{ or } A \in \Gamma_2\}$ and $\Gamma_1 \wedge \Gamma_2 := \{A | A \in \Gamma_1 \text{ and } A \in \Gamma_2\}$. Let

$\Pi_i : \mathcal{S} \times \mathcal{R}_i \to \mathcal{S}_1^{(i)} \times \cdots \times \mathcal{S}_n^{(i)}$ be a scheme realizing Γ_i, $i = 1, 2$, where the common domain of secrets is $\mathcal{S}$.

First, a secret sharing scheme realizing $\Gamma_1 \vee \Gamma_2$ is constructed as follows. Let s be the secret to be shared. Let $\Pi_1(s, r_1) = (s_{11}, \ldots, s_{1n})$ and $\Pi_2(s, r_2) = (s_{21}, \ldots, s_{2n})$, where r_i is chosen uniformly and independently from $\mathcal{R}_i$, $i = 1, 2$. Now let

$$\Pi_\vee : \mathcal{S} \times \mathcal{R}_1 \times \mathcal{R}_2 \to \mathcal{S}_1^{(1)} \times \cdots \times \mathcal{S}_n^{(1)} \times \mathcal{S}_1^{(2)} \times \cdots \times \mathcal{S}_n^{(2)}$$

$$(s, r_1, r_2) \mapsto (s_{11}, \ldots, s_{1n}, s_{21}, \ldots, s_{2n}),$$

where the share domain for player P_i is $\mathcal{S}_i^{(1)} \times \mathcal{S}_i^{(2)}$ and the pair (s_{1i}, s_{2i}) is given to P_i as his share, for $1 \leq i \leq n$.

It is not difficult to prove that $\Pi_\vee$ is a perfect (statistical or computational, respectively) secret sharing scheme computing $\Gamma_1 \vee \Gamma_2$ provided that Π_i is a perfect (statistical or computational, respectively) secret sharing scheme computing Γ_i, $i = 1, 2$.

Second, a secret sharing scheme realizing $\Gamma_1 \wedge \Gamma_2$ is constructed below. As before, let s be the secret to be shared. Choose s_1 and s_2 randomly from $\mathcal{S}$ conditioned on $s_1 + s_2 = s$ and then distribute shares for s_i using Π_i, $i = 1, 2$. More explicitly, let $\Pi(s_1, r_1') = (s_{11}', \ldots, s_{1n}')$ and $\Pi(s_2, r_2') = (s_{21}', \ldots, s_{2n}')$, where r_i' is chosen uniformly and independently from $\mathcal{R}_i$, $i = 1, 2$. Let

$$\Pi_\wedge : \mathcal{S} \times \mathcal{R}_1 \times \mathcal{R}_2 \to \mathcal{S}_1^{(1)} \times \cdots \times \mathcal{S}_n^{(1)} \times \mathcal{S}_1^{(2)} \times \cdots \times \mathcal{S}_n^{(2)}$$

$$(s, r_1', r_2') \mapsto (s_{11}', \ldots, s_{1n}', s_{21}', \ldots, s_{2n}'),$$

where the share domain for player P_i is $\mathcal{S}_i^{(1)} \times \mathcal{S}_i^{(2)}$ and the pair (s_{1i}', s_{2i}') is given to P_i as his share, for $1 \leq i \leq n$.

Once again, it can be proved that $\Pi_\wedge$ is a perfect (statistical or computational, respectively) secret sharing scheme realizing $\Gamma_1 \wedge \Gamma_2$ provided that Π_i is a perfect (statistical or computational, respectively) secret sharing scheme realizing Γ_i, $i = 1, 2$.

Now for any access structure $\Gamma = \{A_1, \ldots, A_l\}$, it is theoretically possible to construct a secret sharing scheme realizing it by using the method demonstrated above recursively, since $\Gamma = \{A_1\} \vee \cdots \vee \{A_l\}$ and there is Shamir's $|A_i|$-out-of-$|A_i|$ secret sharing scheme for each $\{A_i\}$, $1 \leq i \leq l$.

As mentioned before, the construction of [7] generalizes that of [33]. To justify this, suppose

$\Gamma_{\min} = \{A_1, \ldots, A_h\}$ and let $\Gamma_i := \{A_1, \ldots, A_i\}$, $1 \leq i \leq h$. Clearly, $\Gamma_i = \Gamma_{i-1} \vee \{A_i\}$ and for each $1 \leq i \leq h$, the Ito-Saito-Nishizeki scheme realizes $\{A_i\}$ with the domain of secrets $\{0, 1\}$, where each player in A_i receives a one-bit share. Now the Ito-Saito-Nishizeki scheme realizing Γ follows directly by applying the Benaloh-Leichter scheme recursively. To characterize the access structures that can be efficiently computed by the construction of Benaloh and Leichter, we need to view each access structure as a function as follows. First we identify each set $A \subseteq \mathcal{P}$ with it's characteristic vector $\vartheta_A \in \{0, 1\}^n$, where the i^{th} entry $\vartheta_A[j]$ equals 1 if and only if $P_j \in A$. On the other hand, for an access structure Γ, we associate the function $f_\Gamma : \{0, 1\}^n \rightarrow \{0, 1\}$, where $f_\Gamma(\vartheta_B) = 1$ if and only if $B \in \Gamma$. In this way, f_Γ and Γ are uniquely determined by each other, that is, f_Γ completely describes Γ and vice versa. The Boolean function f_Γ is monotone increasing since Γ is. Furthermore, $f_{\Gamma_1} \vee f_{\Gamma_2} = f_{\Gamma_1 \vee \Gamma_2}$ and $f_{\Gamma_1} \wedge f_{\Gamma_2} = f_{\Gamma_1 \wedge \Gamma_2}$. Hence, if an access structure can be described by a small monotone formula, which is defined to be a formula with OR and AND gates but *without* NEGATION gates and *without* negated variables, it can be efficiently computed by the construction of Benaloh and Leichter. Here by small, we mean that the size of such formula, which is defined to be the number of leaves in the tree describing the formula, is small.

Proposition 2.3.1. [7] *Let Γ be an access structure and assume that f_Γ can be computed by a monotone formula in which the variable x_j appears a_j times in the formula. Then the share bit size of Benaloh-Leichter scheme realizing Γ is the product of $max_{1 \leq j \leq n} a_j$ and the bit size of the secret.*

Since any monotone Boolean function with n variables can be computed by a monotone formula, any access structure can be computed by the Benaloh-Leichter scheme. The bad news is that for most monotone Boolean functions, the sizes of the smallest monotone formula computing them are exponential in n, that is, in the resulting scheme, the bit length of the shares is exponential in the number of participants even for a one-bit secret. In all, their construction is not efficient in most of the cases. Actually, it remains open whether there exist efficient schemes for any given access structure, or whether there exists an access structure that does not have efficient schemes. As far as these problems are concerned, Beimel gave the following conjecture:

Conjecture 2.3.2. [5] *There exists an $\epsilon > 0$ such that for every positive integer n there is an access structure with n parties, for which every secret sharing scheme realizing it distributes shares of length exponential in n, that is $2^{\epsilon n}$.*

Ever since this conjecture was proposed, proving or disproving it has been one of the most important open questions concerning secret sharing. The best lower bound that is known so far is given by Csirmaz [19], who constructed for each positive integer n, an explicit access structure with n players, for which the sum of the bit sizes of the shares in every secret sharing scheme is the product of $\Omega(n^2/\text{log}n)$ and the bit size of the secret (for each secret domain that is finite). Since the focus of our research is not on the efficiency of secret sharing schemes, we do not go further in this direction.

2.4 Linear Secret Sharing Schemes and Monotone Span Programs

Most of the existing secret sharing schemes, such as [3, 7, 8, 12, 56], belong to a class of schemes which are known as linear secret sharing schemes. Informally, a linear secret sharing scheme is defined over a finite field $\mathbb{F}$ and the secret to be shared is an element in $\mathbb{F}$. Each player receives from the dealer a share consisting of one or more field elements and each share is computed as a fixed linear function of the secret along with some randomly chosen field elements. Formally,

Definition 2.4.1. Given an access structure Γ, a *linear secret sharing scheme* realizing Γ is specified by a distribution function $\Pi : \mathcal{S} \times \mathcal{R} \to \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ and the corresponding set of reconstruction functions $\{\textsc{Recon}_A : (\mathcal{S}_1 \times \cdots \times \mathcal{S}_n)|_A \to \mathcal{S} | A \in \Gamma_{\min}\}$ that satisfy the following conditions:

- $\mathcal{S}$ is a finite field, say $\mathbb{F}$, and each $\mathcal{S}_i$, $1 \leq i \leq n$, as well as $\mathcal{R}$, is a finite dimensional vector space over $\mathbb{F}$, that is, $\forall 1 \leq i \leq n$, $\mathcal{S}_i = \mathbb{F}^{d_i}$ for some positive integer d_i and $\mathcal{R} = \mathbb{F}^l$ for some positive integer l.

- Each reconstruction function is linear. More explicitly, for any $A \in \Gamma_{\min}$, there exists a set of constants $\{a_{ij} \in \mathbb{F} : P_i \in A, 1 \leq j \leq d_i\}$ such that for any $s \in \mathcal{S}$, $r \in \mathcal{R}$, it holds that $s = \sum_{P_i \in A} \sum_{j=1}^{d_i} a_{ij} \Pi_{ij}(s, r)$, where

$$\Pi(s,r) = (\Pi_{11}(s,r), \ldots, \Pi_{1d_1}(s,r), \ldots, \Pi_{n1}(s,r), \ldots, \Pi_{nd_n}(s,r))$$

  is an element in $\mathbb{F}^{d_1} \times \cdots \times \mathbb{F}^{d_n}$ and the tuple $(\Pi_{i1}(s,r), \ldots, \Pi_{id_i}(s,r))$ is P_i's share, for $1 \leq i \leq n$. Note that here the set of constants $\{a_{ij} \in \mathbb{F} : P_i \in A, 1 \leq j \leq d_i\}$ is independent of both s and r, instead, it only depends on A.

- *Privacy:* For any $B \notin \Gamma$, B can not get any information about the secret s.

In the above definition, the requirement for security (or privacy) is in the information-theoretic sense, which however is not a restriction since it has been proved in [41] that every linear secret sharing scheme is perfect.

In 1989, Brickell [11] first defined monotone span programs, in which ψ is assumed to be bijective. Later in 1993, Karchmer and Wigderson [35] explicitly defined span programs and monotone span programs in general case.

Definition 2.4.2. [35] A *monotone span program* $\mathcal{M}$ is a quadruple $(\mathbb{F}, M, \varepsilon, \psi)$, where $\mathbb{F}$ is a finite field, M is an $m \times e$ matrix, $\psi : \{1, \ldots, m\} \rightarrow \{P_1, \ldots, P_n\}$ is a surjective function and $\varepsilon = (1, 0, \ldots, 0) \in \mathbb{F}^e$ is a fixed vector called the target vector. [2]

ψ labels each row with a participant from $\{P_1, \ldots, P_n\}$, so we can regard each participant as being the "owner" of one or more rows. Hereinafter, without loss of generality, we can assume that the columns of M are linearly independent. In the following, M_A denotes M restricted to those rows assigned via ψ to the players in A and span(M_A) denotes the vector space spanned by the rows of M_A, where A is any subset of $\mathcal{P}$. We denote $M_{\{P_i\}}$ as M_{P_i} for simplicity. $\mathcal{M}$ yields a linear secret sharing in the following way: to distribute a secret $s \in \mathbb{F}$, the dealer chooses a random vector $\rho \in \mathbb{F}^{e-1}$ and writes $\theta = (s, \rho)$. For each $1 \leq i \leq n$, $M_{P_i} \cdot \theta^T$ is given to P_i as his share. The following proposition was proved by Brickell [11], Brickell and Davenport [12] and Karchmer and Wigderson [35]. It implies that the resulting linear secret sharing scheme has access structure $\Gamma = \{A \subseteq \mathcal{P} : \varepsilon \in \text{span}(M_A)\}$.

Proposition 2.4.3. [11, 12, 35] *Given an access structure* Γ*, if* $(\mathbb{F}, M, \varepsilon, \psi)$ *is constructed such that:* $\varepsilon \in \text{span}(M_A)$ *if and only if* $A \in \Gamma$*, then the resulting linear secret sharing scheme realizes* Γ *and in this case we say that* $(\mathbb{F}, M, \varepsilon, \psi)$ *computes* Γ*.*

Proof. First, we show that every authorized subset $A \in \Gamma$ can reconstruct the secret. By assumption, $\varepsilon \in \text{span}(M_A)$, that is, there exists some vector ϑ such that $\varepsilon = \vartheta \cdot M_A$. Thus, the participants in A can recover the secret s by computing

$$\vartheta \cdot (M_A \cdot \theta^T) = (\vartheta \cdot M_A) \cdot \theta^T = \varepsilon \cdot \theta^T = \langle \varepsilon, \theta \rangle = s,$$

[2] Any nonzero vector in $\mathbb{F}^e$ can be a target vector.

where $\langle \cdot, \cdot \rangle$ denotes the scalar product of two vectors. Note that $M_A \cdot \theta^T$ is available to A, since its entries are exactly the shares of the participants in A.

Next we prove the privacy condition. For any $B \notin \Gamma$, it holds that $\varepsilon \notin \text{span}(M_B)$. Thus $\text{rank}(M_B) < \text{rank}(\binom{M_B}{\varepsilon})$, where $\binom{M_B}{\varepsilon}$ is the matrix containing the rows of M_B along with an additional row ε. It follows that there is a vector $\omega \in \mathbb{F}^e$ whose first coordinate is 1 (namely, the scalar product $\langle \varepsilon, \omega \rangle$ equals 1) such that $M_B \cdot \omega^T = 0$. Without loss of generality, we assume that $(s_1, \ldots, s_{|B|})$ is the tuple of shares of participants in B when the secret is s. Now for any $s' \in \mathbb{F}$, let $\mu = \theta + (s' - s)\omega$. Note that the first coordinate of μ is s', since the first coordinate of ω is 1 and the first coordinate of θ is s. Since $M_B \cdot \mu^T = M_B \cdot \theta^T + (s' - s)(M_B \cdot \omega^T) = (s_1, \ldots, s_{|B|})^T$, $s_1, \ldots, s_{|B|}$ are also the shares of participants in B when the secret is s'. Thus, for every $s \in \mathbb{F}$, the number of random strings that generate the shares $s_1, \ldots, s_{|B|}$ for participants in B when the secret is s is the same as the number of random strings that generate those shares when the secret is s'. This completes the proof for privacy.

□

Example 2.4.4. Let $\mathcal{P} = \{P_1, \ldots, P_6\}$ be a set of players. Consider the access structure Γ over $\mathcal{P}$ defined by

$$\Gamma_{min} = \{\{P_1, P_2\}, \{P_3, P_4\}, \{P_5, P_6\}, \{P_1, P_5\}, \{P_1, P_6\}, \{P_2, P_6\}, \{P_2, P_5\}, \{P_3, P_6\}, \{P_4, P_5\}\}$$

Define the matrix M over $\mathbb{F} = \text{GF}(2)$ with the labeling map ψ such that

$$M_{P_1} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{P_2} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{P_3} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{P_4} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$M_{P_5} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$M_{P_6} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

It can be easily checked that $\mathcal{M} = (\mathbb{F}, M, \psi, \varepsilon)$ computes Γ.

Karchmer and Wigderson [35] showed that monotone span programs imply linear secret sharing schemes. Beimel [5] proved the converse direction. Thus, linear secret sharing schemes are equivalent to monotone span programs. Hence, we do not distinguish between a monotone span program and the corresponding linear secret sharing scheme.

2.5 Verifiable Secret Sharing

We would like to emphasize that the correctness and privacy of the constructions in the *plain* secret sharing schemes described above cannot be guaranteed without the assumption that each player as well as the dealer is honest, namely, he always follows the prescribed steps. However, in practice, this assumption cannot be guaranteed automatically. Instead, the participants are supposed to be partitioned into two categories: honest participants and malicious participants. Here by malicious participants, we mean that they can act in an arbitrary manner in order to disrupt the protocol. Besides, as malicious parties may communicate with each other to form *secret* coalitions (here *secret* means that each honest player has no idea about who is malicious and who is not), all the malicious parties are often modeled as being controlled by a single entity called an adversary, who can be the dealer, some participant or some outsider. With regard to the adversary, one can consider *semi-honest* (or *passive*) adversaries (who follow the protocol honestly but try to learn more than they should) or *malicious* (or *active*) adversaries (who act in an arbitrary manner). In addition, an adversary may be limited to polynomial-time (yielding the computational setting) or unbounded (yielding the information-theoretic setting). Finally, the adversary may be *static* (meaning that the set of corrupted parties is fixed before the protocol begins) or *adaptive* (meaning that the adversary can adaptively choose which participants to corrupt during the execution of the protocol). Most of the constructions we have introduced so far fail in this adversary model. Take Shamir's t-out-of-n secret sharing scheme [56] as an example. In the process of secret reconstruction, if a single player broadcasts a fake share while the other $t-1$ players act honestly, then he will be the only one learning the real secret. This motivates the primitive of verifiable secret sharing [6, 14, 25, 28, 42, 52, 53, 60], which allows each player to check the consistency of the shares distributed by the dealer when the dealer is corrupted and allows each player to check the validity of other players' shares during the secret reconstruction phase. Besides, the privacy and correctness requirements of a verifiable secret sharing scheme can be achieved even when some percentage of the participants (including the dealer) are controlled by the adversary. Due to these properties, it has turned out to be a useful building block for secure multiparty computation.

Tompa and Woll [60] and McEliece and Sarwate [42] first considered schemes with faulty participants and gave practical solutions for that problem. In their scheme, the dealer is always assumed to be honest. Chor *et al.* [16] first defined the complete notion of verifiable secret sharing and gave a construction based on some cryptographic assumptions. There are many papers which have discussed verifiable secret sharing recently. Most schemes use zero-knowledge proofs, such as [6, 14, 25, 28, 52, 53]. Others use cryptographic assumptions such as the hardness of discrete logarithm [24, 50]. It was shown in [6] that (in the active adversary model) in any unconditional secure t-out-of-n verifiable secret sharing scheme, it holds that $t < \frac{n}{3}$. This result implies that any verifiable secret sharing scheme (resilient to an active adversary) with $t \geq \frac{n}{3}$ will inevitably rely on some cryptographic assumptions. On the other side, in the passive adversary model, in any unconditional secure t-out-of-n verifiable secret sharing scheme, it holds that $t < \frac{n}{2}$.

Verifiable secret sharing can be classified into two groups: interactive verifiable secret sharing - interactions between the players (may include the dealer) are needed in order to verify the shares and non-interactive verifiable secret sharing - no interaction between the players is needed for the verification of shares. Obviously, the latter is easier to be implemented in practice and we have more interest in it. In such a scheme only the dealer is allowed to send messages and in particular, the players can not communicate with each other or with the dealer during the process of verification. There are two famous non-interactive secret sharing schemes in the history, both of which rely on the computational assumption that computing discrete logarithm is hard. One is given by Pedersen in [50], where no information about the secret is revealed but the dealer can succeed in distributing inconsistent shares with a negligible probability. The other is given by Feldman in [24], where each single player can recover the secret with a negligible probability but even an infinitely powerful dealer can never succeed in distributing inconsistent shares. As pointed out by Pederson [50], it is impossible to construct a non-interactive verifiable secret sharing scheme in which no information about the secret is revealed and even a dealer with unlimited computing power can not succeeding in distributing inconsistent shares. In this sense, these two schemes complement each other.

Here we would like to introduce the non-interactive verifiable secret sharing scheme based on the Chinese Remainder Theorem [36], merely for the reason that we will use it (with some necessary modifications for our needs) as a building block for our construction in Chapter 7.

2.6 A t-out-of-n Verifiable Secret Sharing Scheme Based on the Asmuth-Bloom Secret Sharing Scheme

In order to allow players successfully detect inconsistent shares from a dishonest dealer, Kaya and Selcuk used a range proof technique originally proposed by Boudot [10] and later modified by Cao *et al.*[13] to prove that a committed number lies within some interval. We are going to introduce this technique in the following subsection.

2.6.1 Range Proof Techniques

Boudot [10] proposed an efficient and non-interactive technique to prove that a committed number lies within an interval. Cao *et al.* [13] applied the same proof technique with a different commitment scheme to obtain shorter range proofs: the commitment of a number y is computed as

$$E = E(y) = g^y \bmod N,$$

where g is an element in $\mathbb{Z}_N^*$ and the factorization of N is not known. We refer the readers to [13] for further details. Throughout this section, we will use $\mathrm{RngPrf}(E(y), M)$ to denote the range proof that a secret integer y committed with $E(y)$ is in the interval $[0, M)$ and we will use it as a black box.

2.6.2 The Scheme

Share Distribution Phase

1. Parameters Setup

To share a secret s, the dealer chooses $p_0(> s)$ and publishes it.

1. The dealer chooses and publishes a set of prime integers $p_1, \ldots, p_n$ of bit length at least k (a security parameter) such that the following requirements are satisfied:

   (a) $p_0 < p_1 < \cdots < p_n$;

   (b) $\prod_{i=1}^{t} p_i > p_0 \prod_{i=1}^{t-1} p_{n-i+1}$;

   (c) $q_j = 2p_j + 1$ is prime for any $1 \le j \le n$.

2. For any $1 \le i \le n$, let G_i be a subgroup of $\mathbb{Z}_{q_i}^*$ of order p_i and g_i be a generator of G_i. Let $Q = \prod_{i=1}^{n} q_i$. He computes $g = (\sum_{i=1}^{n} g_i \cdot Q_i' \cdot \frac{Q}{q_i}) \bmod Q$, where Q_i' denotes the inverse of $\frac{Q}{q_i}$ in $\mathbb{Z}_{q_i}^*$. The dealer publishes g.

3. The dealer chooses and publishes an RSA modulus N whose factorization is unknown to any of the n players.

2. Share Distribution

To share a secret $s \in \mathbb{Z}_{p_0}$ among a group of n players $\{P_1, \ldots, P_n\}$, the dealer executes the following steps.

1. He sets $M := \prod_{i=1}^{t} p_i$. He computes $y = s + A_0 \cdot p_0$ for some positive integer A_0 generated randomly subject to the condition that $0 < y < M$, calculates $y_i = y \bmod p_i$ and finally sends the share y_i to player P_i secretly, for $1 \le i \le n$.

2. He computes $E(y) := g^y \bmod QN$ and broadcasts $E(y)$ and $\mathrm{RngPrf}(E(y), M)$.

3. Each player P_i checks whether $g_i^{y_i} \equiv E(y) \bmod q_i$ to verify $y_i = y \bmod p_i$. Then he checks the validity of the range proof to verify $y < M$. If either of the two check procedures fails, P_i broadcasts a complain about the dealer.

4. If at least t players complain, the dealer is disqualified.

Secret Reconstruction Phase

Let U be a coalition of at least $2t - 1$ players.

1. The share y_i of player P_i can be verified by the other players in U with the verification equality $g_i^{y_i} \equiv E(y) \bmod q_i$.

2. If all shares are valid, the players can obtain the secret s by using the reconstruction procedure described in the Asmuth-Bloom Secret Sharing Scheme [3]. Otherwise, the corrupted players are disqualified and the remaining players recover the secret by ignoring the corrupted shares.

Theorem 2.6.1. [36] *When the dealer and the players are honest, any coalition of t players can recover the secret s. On the other hand, under the assumptions related to discrete logarithm problem and RSA, for a passive adversary who obtains $t-1$ shares, every candidate for the secret is equally likely, i.e., the probabilities $\Pr[s = s']$ and $\Pr[s = s'']$ are approximately equal for all $s', s'' \in \mathbb{Z}_{p_0}$ provided that p_0 is large enough.*

Theorem 2.6.2. [36] *Under the assumptions related to discrete logarithm problem and RSA, a corrupted dealer cannot cheat in this VSS scheme without being detected except with negligible probability. In other words, if the shares are inconsistent with the secret s, then at least one verification equation does not hold except with negligible probability in the security parameter k.*

Theorem 2.6.3. [36] *Under the assumptions related to discrete logarithm problem and RSA, no player can cheat in this VSS scheme without being detected.*

2.7 Proactive Secret Sharing

Secret sharing schemes protect secrets by splitting them among a set of players. In particular, in t-out-of-n threshold schemes, security (privacy) is guaranteed if *throughout the entire life-time of the secret*, the adversary is restricted to compromise less than t players. However, for long-lived and sensitive secrets, this protection may be insufficient, since the adversary has the *entire life-time* *of the secret* to corrupt players one by one. This limitation motivated the primitive of proactive secret sharing, which was first proposed by Herzberg, Jarecki, Krawczyk and Yung [32]. Proactive security refers to security and correctness in the presence of a mobile adversary, who is able to attack all the players over a long period of time and in any time period there are only a subset of players that are corrupted. Here time is divided into *time periods* which are determined by a common global clock (e.g., a day, a week, etc.). Hence, in a proactive secret sharing scheme, in order to periodically renew their shares without changing and revealing the secret, players need to execute a *share update protocol* merely by interactions among themselves in such a way that any information learned by the adversary during each period becomes obsolete after the shares are renewed, and as a result, the adversary has to start a new attack from scratch during each time period. At the beginning of each time period, the players engage in an interactive *update protocol* and at the end of an updated phase, the players hold new shares of the secret s. The adversary can corrupt players at any moment during a time period. If a player is corrupted during an update phase, he is regarded as being corrupted during both periods adjacent to that update phase. Note that once a player is corrupted, the adversary learns the internal state of the player, which consists of the complete history of that player, and takes full control of that player and can make him deviate from the protocol in any desired manner. There is much research on this topic [20,32,57]. However, in the rest of this section, we would like to introduce the proactive secret sharing scheme proposed by Arco and Stinson [20], since the basic idea behind it will be used in our protocol in Chapter 6.

2.7.1 An Unconditionally Secure Proactive t-out-of-n Secret Sharing Scheme

This scheme was proposed by Arco and Stinson in [20], where they slightly modified the version of the scheme proposed by Stinson and Wei in [57]. There is an adversary who can corrupt up to b players including the dealer. Here $n \geq t + 3b$ and $t > b + 1$. We assume that all players are connected by pairwise secure channels and have access to a broadcast channel. Let $\mathbb{F} = \mathrm{GF}(q)$ be a finite field, of which ω is a primitive element. In the following protocol, all the computations are in the field $\mathbb{F}$.

Share Distribution Phase

- To share a secret s among n participants, the dealer chooses a symmetric polynomial

$$f(x, y) = \sum_{i=0}^{t-1} \sum_{j=0}^{t-1} a_{ij}\, x^i\, y^j,$$

  where $a_{00} = s$ and $a_{ij} (= a_{ji})$ are chosen randomly and independently from $\mathbb{F}$ for all i, j. Then for each l, he sends $h_l(x) = f(x, \omega^l)$ to player P_l through a secure channel. Besides, for each i, P_i generates and sends to every player P_l $(l \neq i)$ a random value $r_{il} \in \mathbb{F}$ through pairwise secure channels.

- After receiving $h_l(x)$ and $r_{1l}, \ldots, r_{nl}$, each player P_l broadcasts the value $h_l(\omega^u) + r_{lu} + r_{ul}$, for each $u \neq l$.
- Each player P_i computes the maximum subset $G \subseteq \{1, \ldots, n\}$ such that any ordered pair (u, l) is *consistent*, that is, $h_l(\omega^u) + r_{lu} + r_{ul} = h_u(\omega^l) + r_{ul} + r_{lu}$. If $|G| > n - t$, then P_i outputs ACCEPT. Otherwise, P_i outputs REJECT.
- If at least $n - b$ of the players output ACCEPT, then the dealer erases all the information about the scheme on his end. Otherwise, the dealer reboots the whole system and initializes the system again.

Remark 2.7.1. Note that here every honest participant computes the same subset G at the end of the share distribution phase. We like to emphasize that the adversary could provide correct information during the share distribution phase but wrong information in the reconstruction phase.

Secret Reconstruction Phase

- Each player P_i sends $h_i(0)$ to each P_l, where $i \in G$, the set of good players after the share distribution phase.
- After receiving the $h_i(0)s$, P_l computes a polynomial $f_l(0, y)$ such that $f_l(0, \omega_i) = h_i(0)$ for at least $n - 2b$ of the data he has received. This operation can be done efficiently by using error correction techniques for Reed-Solomon Codes [54].
- Player P_l computes and outputs $s' = f_l(0, 0)$.

The following steps allow players to update their shares related to the original secret s without revealing any information about s. Besides, after this process, the previous shares become obsolete.

Share Renewal Phase

- Each player P_l selects a random symmetric polynomial $r^l(x,y) = \sum\limits_{i=0}^{t-2}\sum\limits_{j=0}^{t-2} r^l_{ij}\, x^i\, y^j$, where $r^l_{ij} = r^l_{ji}$ for all i, j.
- P_l sends $h^l_u(x) = r^l(x, \omega^u)$ to P_u for $u \in \{1, \ldots, n\} \setminus \{l\}$ by pairwise secure channels.

Share Renewal Phase, cont.

- After receiving $h^l_u(x)$, P_u computes and sends the value $h^l_u(\omega^v)$ to P_v, for $v \in \{1, \ldots, n\}\backslash\{u\}$ by pairwise privacy channel.
- P_v checks whether $h^l_v(\omega^u) = h^l_u(\omega^v)$ for $u = 1, \ldots, n$. If the equation is not true for more than b values of u, then P_v broadcasts an accusation of P_l.
- If P_l is accused by at most b players, then he can defend himself as follows: for those P_i he is accused by, P_l broadcasts $h^l_i(x)$. Then, player P_u checks whether $h^l_i(\omega^u) = h^l_u(\omega^i)$ and broadcasts "yes" or "no" accordingly. If, for every broadcasted $h^l_i(x)$, there are at least $n - b - 2$ players broadcasting yes, then P_l is not a bad player. In this case, if P_i has an $h^l_i(x)$ different from the one that P_l has broadcasted, then he stores the broadcasted one.
- P_v updates the list G of good players (namely, the players found bad in the previous step are not in G) and updates his share as

$$h_v(x) \leftarrow h_v(x) + (x + \omega^v)h^*_v(x),$$

where $h^*_v(x) = \sum_{l \in G} h^l_v(x)$.

During each time period, the players need to check if some of them have been corrupted by the adversary and those corrupted players should be rebooted to recover their secret shares in order to recover a correct secret. The following steps enable the detection of corrupted players and the recovering of good shares, once the corrupted players have been rebooted.

Detection Phase

- P_l computes and sends $h_l(\omega^u)$ to P_u for $u = 1, \ldots, n$ by pairwise secure channels.
- P_u checks whether $h_l(\omega^u) = h_u(\omega^l)$. Then he broadcasts an accusation $list_u$ which contains those l such that $h_l(\omega^u) \neq h_u(\omega^l)$ or $h_l(\omega^u)$ was not received.
- Each good player updates the list G so that it does not contain those l accused by at least $b + 1$ players of the system.

Rebooting Phase

- For each $l \notin G$, every good player P_i computes and sends $h_i(\omega^l)$ to P_l.
- Upon receiving the data, P_l computes the polynomial $h_l(x)$ that agrees with the majority of the values $h_l(\omega^u)$ he has received. P_l sets $h_l(x)$ as his new share.

3. LINEAR SECRET SHARING SCHEMES AND LINEAR BLOCK CODES

In this chapter, we introduce the notion of linear block codes associated to linear secret sharing schemes, which was proposed by Nikova and Nikov [45]. Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be a set of players and Γ be an access structure over $\mathcal{P}$. Let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be a monotone span program (MSP for short) computing Γ, where without loss of generality, we assume that the columns of M are linearly independent and ψ preserves ordering, that is, $\psi(i) \leq \psi(j)$ whenever $i < j$. (Here the ordering on the set of players is defined to be: $P_1 < \cdots < P_n$). Let C be the linear code generated by M^T. As a classical code, C can correct v errors if and only if $2v+1 \leq d$, where d is the minimal distance of C. Let m_i be the number of rows assigned to P_i via ψ, for $1 \leq i \leq n$. In this chapter, we regard C as a linear block code by splitting each codeword of C into n parts such that it is an element of $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ and define the set of admissible distances of C, denoted by $\Gamma(C)$, and the set of forbidden distances of C, denoted by $\Delta(C)$, which plays a role as Hamming distance does in classical codes, since we may measure the error-set correcting ability of the linear block code C by $\Delta(C)$. Nikova and Nikov [45] demonstrated that C can correct each error pattern $A \in \mathcal{A}$ if and only if C satisfies the condition $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$, where $\mathcal{A}$ denotes the adversary structure and $\mathcal{A} \uplus \mathcal{A} = \{B_1 \cup B_2 : B_1, B_2 \in \mathcal{A}\}$. However, it is hard to check directly from $\mathcal{M}$ whether the condition $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ is satisfied or not. In this chapter, we make an attempt to partially solve this problem by proposing an equivalent condition for $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ from the perspective of the matrix $M^{\perp}$, whose first column is a solution vector of $M^T \cdot X = \mathbf{0}$ and the remaining columns form a basis of the space $ker(M^T)$. Besides, we demonstrate that C can detect all error patterns in $\mathcal{A}$ if and only if $\mathcal{A} \subseteq \Delta(C)$ and give an equivalent condition from the perspective of $M^{\perp}$.

3.1 Linear Block Codes Associated to Linear Secret Sharing Schemes

Let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be a monotone span program computing Γ, where M has size $m \times e$. Let $Lsss : (\mathbb{F}, \mathbb{F}^{e-1}) \to \mathbb{F}^m$ be a function defined below:

$$Lsss(s, \rho) = (s, \rho) \cdot M^T,$$

where s is the secret to be shared and ρ is a random vector chosen from $\mathbb{F}^{e-1}$ when sharing s.

Definition 3.1.1. An $[m, e]$ *linear code* C over a finite field $\mathbb{F}$ is an e-dimensional subspace of $\mathbb{F}^m$, where m denotes the length of the vectors in C. Each vector $\mathbf{c}$ in C is called a codeword of C and each vector in $\mathbb{F}^m$ is called a word.

As before, let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP computing an access structure Γ, where without loss of generality we assume that the columns of M are linearly independent. Let $G = M^T$, where ^{T} denotes transposition of matrices. Then G has full rank e, since M has rank e. Let C be the subspace spanned by the rows of G. Clearly, C is an $[m, e]$ linear code and $Lsss(s, \rho) = (s_1, \ldots, s_m)$ is a codeword of C, that is, the sequence of all the shares, when rearranged properly, is a codeword of C. In the following, without loss of generality, we always assume that ψ preserves ordering, i.e., $\psi(i) \leq \psi(j)$ whenever $i < j$, where it is assumed that $P_1 < \cdots < P_n$. Let $m_i = |\{j : \psi(j) = P_i\}|$, that is, m_i is the number of rows assigned to P_i, $1 \leq i \leq n$. Let $\lambda_i = m_1 + \cdots + m_i$ for $1 \leq i \leq n$ and $\lambda_0 = 0$. Then for any $1 \leq i \leq n$, $S_i = (s_{\lambda_{i-1}+1}, \ldots, s_{\lambda_{i-1}+m_i})$ is the share for P_i. It is quite natural to regard $(S_1, \ldots, S_n)$ as an n-tuple that belongs to $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$. Hereinafter, for each codeword $\omega \in C$, we regard it as a tuple in $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$. In this way, each codeword $\omega \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ is called a *block codeword* and $C(\subseteq \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n})$ is called a block code. For each vector in $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$, we index the i^{th} block by P_i.

Definition 3.1.2. Let $\nu = (V_{P_1}, \ldots, V_{P_n}) \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$. $\text{supp}(\nu) := \{P_i : V_{P_i} \neq \mathbf{0}\}$ is called the *block support* of the block vector ν.[1]

If an MSP $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ computing a given access structure Γ has size n, that is, ε is injective, then each $P_i's$ share is a field element. As a result, the bitsize of each share equals the bitsize of the secret. In this case, the corresponding linear block code C is a classical linear code,

[1] Each block support is a subset of players. In this thesis, we only consider block support and we call it support for simplicity.

which is a subspace of $\mathbb{F}^n$. The *Hamming sphere* $B_r(X)$ of radius r centered around a vector $X \in \mathbb{F}^n$ is defined as $B_r(X) = \{Y \in \mathbb{F}^n : d(X, Y) \leq r\}$, where $d(X, Y) := |\text{ supp } (X - Y)|$ is called the Hamming distance between X and Y. Given a code C, the minimal distance of C, denoted by d, is the smallest of all distances between different codewords in C, namely, $d = \min_{w_1, w_2 \in C, w_1 \neq w_2} d(w_1, w_2)$. One of the basic coding theory problems is the so-called *Sphere Packing Problem*: given n and r, what is the maximal number of non-intersecting spheres of radius r that can be placed in $\mathbb{F}^n$, the n-dimensional Hamming space. Note that the distance between any two distinct centers of these pairwise disjoint spheres is at least $2r + 1$. Suppose one codeword of C is transmitted and at most $\lfloor \frac{d-1}{2} \rfloor$ errors occur during the transmission due to channel noise. To decode the received message, namely, to determine which codeword was actually transmitted, we can compute the Hamming distance between the received message and each of the codewords of C. Observe that all the spheres with radius $\lfloor \frac{d-1}{2} \rfloor$ centered at the codewords of C are pairwise disjoint. Since at most $\lfloor \frac{d-1}{2} \rfloor$ errors occur, the transmitted codeword will be the (unique) nearest codeword to the received vector. In this way, all errors can be corrected. It follows that a code (not necessarily linear) with minimal distance d can correct at most $\lfloor \frac{d-1}{2} \rfloor$ errors. Obviously, the minimal distance of a classical code measures its error-correcting ability: the larger the minimal distance is, the larger the number of errors that it can correct. However, in the model of secret sharing, we assume that all the communication channels are secure, and hence there is no channel noise. Instead, an adversary may corrupt some subset $A \in \mathcal{A}$ of players by forcing them to send fake shares in the secret reconstruction process. In this situation, the error positions are not totally random, but instead, they form a set which belongs to $\mathcal{A}$, the adversary structure. More explicitly, let $S = (S_{P_1}, \cdots, S_{P_n})$ be a block codeword, where S_{P_i} is P_i's share from the dealer. In the reconstruction phase, each player broadcasts his share. For $1 \leq i \leq n$, let S^i be the tuple of the messages P_i collected. It is likely that $S^i \neq S$ and in this case, $S - S^i$ is called an error vector. However, what we are certain is that $\text{supp}(S - S^i) \in \mathcal{A}$. An error vector is called an $\mathcal{A}$-error if its support is in $\mathcal{A}$. Nikov *et al.* [46, 47] made a first step to characterize the error-correcting ability of any linear block code by its forbidden distances, where each possible error vector is supposed to be an $\mathcal{A}$-error.

3.2 $\mathcal{A}$-Error Correctable Linear Secret Sharing Schemes

First, we briefly recall some definitions and observations. The following operation for monotone decreasing sets, which is called element-wise union, was introduced in [23, 47].

Definition 3.2.1. [23,47] The operation $\uplus$ for any two monotone decreasing sets $\mathcal{A}_1, \mathcal{A}_2$ is defined as follows: $\mathcal{A}_1 \uplus \mathcal{A}_2 = \{A_1 \cup A_2 : A_1 \in \mathcal{A}_1, A_2 \in \mathcal{A}_2\}$.

For any two vectors $X = (X_{P_1}, \ldots, X_{P_n})$ and $Y = (Y_{P_1}, \ldots, Y_{P_n})$, where $X_{P_i}, Y_{P_i} \in \mathbb{F}^{m_i}$ and $m = m_1 + \cdots + m_n$, we define $\delta_m(X, Y) := \{P_i | X_{P_i} \neq Y_{P_i}\}$. Clearly, $\delta_m(X, Y)$ =supp$(X - Y) \subseteq \mathcal{P}$. In [23], Fehr and Maurer pointed out that $\delta_m(X, Y)$ behaves like a metric, since for all vectors $X, Y, Z \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$, it holds that

1. $\delta_m(X, X) = \emptyset$.

2. $\delta_m(X, Y) = \delta_m(Y, X)$.

3. $\delta_m(X, Z) \subseteq (\delta_m(X, Y) \bigcup \delta_m(Y, Z))$.

In the remainder of this chapter, we use $\delta_m(X, Y)$ instead of Hamming distance to describe the properties of the so defined space as in [23].

Given an access structure Γ, as before, let $\mathcal{A}$ denote the corresponding adversary structure, that is, $\mathcal{A} = 2^{\mathcal{P}} \setminus \Gamma$. Note that $\mathcal{A}$ is a monotone decreasing collection of subsets of players. Then $B_{\mathcal{A}}(X)$, the $\mathcal{A}$-neighborhood of pseudo-radii in $\mathcal{A}$ centered around the vector X,[2] is defined as follows:

$$B_{\mathcal{A}}(X) = \{Y \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n} : \delta_m(X, Y) \in \mathcal{A}\}.$$

When Γ is the t-out-of-n threshold access structure, $B_{\mathcal{A}}(X)$ coincides with the Hamming sphere $B_{t-1}(X)$. The generalized sphere packing problem is described as follows:

Generalized Sphere Packing Problem: Given $m = m_1 + \cdots + m_n$ and $\mathcal{A}$, what is the maximal number of non-intersecting $\mathcal{A}$-neighborhoods that can be placed in the m-dimensional space with metric δ_m?

Let $C \subseteq \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ be a block code. The set of admissible distances, denoted by $\Gamma(C)$,

[2] This notation can be defined for any monotone decreasing collection of subsets of players.

is defined by

$$\Gamma(C) = \{A : \text{ there exist } X, Y \in C \text{ with } X \neq Y \text{ such that } \delta_m(X,Y) \subseteq A\}$$

The set of forbidden distances, denoted by $\Delta(C)$, is defined as the complement of $\Gamma(C)$, that is, $\Delta(C) := 2^{\mathcal{P}} \setminus \Gamma(C)$. Observe that $\Gamma(C)$ is monotone increasing and $\Delta(C)$ is monotone decreasing. In particular, when C is a *linear block code*, the set of admissible distances is

$$\Gamma(C) = \{A : \text{ there exists a nonzero codeword } X \in C \text{ such that } \mathrm{supp}(X) \subseteq A\}.$$

As mentioned previously, a classical code is v-error-correcting (if the minimal distance decoding rule is used) if and only if its minimal distance is at least $2v+1$. Moreover, in coding theory, any subset of coordinates is equally likely to be in error due to noise during transmission. In the secret sharing model, in order to recover the secret, each player needs to send his share to all the others through a secure channel that provides privacy and authentication. However, some players may send fake shares. In this setting, only some particular subsets (those in $\mathcal{A}$) of coordinates can be in error. Let E be an error vector occurring during the secret reconstruction process. Then $\mathrm{supp}(E) \in \mathcal{A}$ is called the error pattern of E. With regard to those error patterns in $\mathcal{A}$, Nikov and Nikova [45] proposed and proved the following proposition, which indicates that block codes have similar error-correcting capabilities as the classical codes.

Proposition 3.2.2. [45] *A block code C with the set of forbidden distances $\Delta(C)$ can correct each error pattern in $\mathcal{A}$ if and only if $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. If it is the case, $\mathcal{M}$ is said to be $\mathcal{A}$-*error correctable.[3]

Proof. Suppose that $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. Let X be the block codeword sent and Y be block word received with $\delta_m(X,Y) \in \mathcal{A}$. We claim that there is no block codeword X' other than X such that $\delta_m(X',Y) \in \mathcal{A}$. Suppose not. Then by the triangle inequality, we have $\delta_m(X,X') \subseteq \delta_m(X,Y) \cup \delta_m(Y,X')$, which implies that $\delta_m(X,Y) \cup \delta_m(Y,X') \in \Gamma(C)$, since $\delta_m(X,X') \in \Gamma(C)$ and $\Gamma(C)$ is monotone increasing. On the other hand, $\delta_m(X,Y) \cup \delta_m(Y,X') \in \Delta(C)$ since $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. A contradiction arises. Thus Y can be decoded correctly and uniquely to X in this way.

[3] We may replace the adversary structure $\mathcal{A}$ by any monotone decreasing set of subset of players while preserving the correctness of this proposition. Besides, C can be nonlinear.

Now it remains to show the converse is also true. Suppose that C can correct all the errors in $\mathcal{A}$, that is, for any block codeword X and any block word Y conditioned on $\delta_m(X,Y) \in \mathcal{A}$, there is no block codeword X' other than X such that $\delta_m(X',Y) \in \mathcal{A}$. Suppose by contradiction that $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ does not hold. Then, there exist $A, B \in \mathcal{A}$ such that $A \cup B \in \Gamma(C)$. We can further assume that $A \cap B = \emptyset$, since $\mathcal{A}$ is monotone decreasing. Thus, there exist two distinct block codewords W and V such that $\delta_m(W - V) = A \cup B$. After a suitable permutation, we can assume that $W = (W_A, W_B, W_{\overline{A \cup B}})$ and $V = (V_A, V_B, W_{\overline{A \cup B}})$. Now let $Z = (W_A, V_B, W_{\overline{A \cup B}})$. Clearly, $\delta_m(W,Z) = B \in \mathcal{A}$ and $\delta_m(V,Z) = A \in \mathcal{A}$, which contradicts with our assumption. This completes the proof. □

Example 3.2.3. In the case of Shamir's t-out-of-n secret sharing scheme, the corresponding linear code C is the generalized Reed-Solomon code with parameters $[n, t, n-t+1]$, where $n-t+1$ is the minimal distance of C. Here $\mathcal{A} = \{B \subseteq \mathcal{P} : |B| \leq t-1\}$ and $B_{\mathcal{A}}(X) = B_{t-1}(X)$. It follows that $\mathcal{A} \uplus \mathcal{A} = \{B : |B| \leq 2(t-1)\}$. Since C is an MDS (maximal distance separable) code, it can be checked easily that $\Gamma(C) = \{A \subseteq \mathcal{P} : |A| \geq n-t+1\}$, which further implies that $\Delta(C) = \{A \subseteq \mathcal{P} : |A| \leq n-t\}$. It is well known in coding theory that C can correct all the errors in $\mathcal{A}$, that is, it can correct at most $t-1$ errors, if and only if $n-t+1 \geq 2(t-1)+1$, namely, $n \geq 3t-2$. In fact, this classical conclusion can also be deduced from Proposition 3.1.2 as follows:

$$\begin{aligned}
& C \text{ can correct all errors with Hamming weight no larger than } t-1 \\
& \Leftrightarrow C \text{ can correct all errors in } \mathcal{A} \\
& \Leftrightarrow \mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C) \\
& \Leftrightarrow 2(t-1) \leq n-t \\
& \Leftrightarrow n \geq 3t-2 \\
& \Leftrightarrow \text{ the minimal distance of } C, \text{ which is } n-t+1, \text{ is at least } 2(t-1)+1.
\end{aligned}$$

Remark 3.2.4. Let Γ be any given access structure and $\mathcal{A}$ be the corresponding adversary structure. Let $\mathcal{M}$ be an MSP computing Γ and the corresponding linear block code be C. Let $S = (S_{P_1}, \ldots, S_{P_n})$ be the block codeword, where S_{P_i} is player P_i's share obtained from the dealer. During the reconstruction phase, some coalition $A \in \mathcal{A}$ may be corrupted and send fake shares to the non-coalition players. Let S^i denote the block word whose entries are the data P_i receives in the reconstruction phase. Thus $\delta_m(S, S^i) \in \mathcal{A}$. Suppose $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. To decode the received message S^i, P_i needs to compute the $\mathcal{A}$-neighborhood $B_{\mathcal{A}}(S^i)$ and decode S^i to the unique codeword lying in this $\mathcal{A}$-neighborhood. In this way, each P_i can correct all the error patterns in $\mathcal{A}$ and thus can recover the real secret even if players in some coalition $A \in \mathcal{A}$ cheat about their shares.

At this point, several questions arise:

1. With regard to error patterns in $\mathcal{A}$, is there any efficient decoding algorithm to execute the decoding process mentioned in Remark 3.2.4?

2. Perhaps, it is not hard to compute $\mathcal{A} \uplus \mathcal{A}$. Can we compute $\Delta(C)$ without listing all the codewords of C?

3. Can we give an equivalent (or merely a sufficient) condition for $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ by merely characterizing the particular properties that M and $\mathcal{A}$ should satisfy?

4. For any given Γ, does there exist an MSP computing it such that the corresponding linear block code satisfies $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$? If the answer is negative, can we identify or classify the access structures Γ that satisfy this property?

Before we give a partial answer to question 2 and question 3, we first need to explore some properties for $\Delta(C)$.

As before, let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP computing Γ and let C be the linear block code generated by M^T. Denote $\tilde{C} \subseteq \mathbb{F} \times \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ the linear block code generated by $\tilde{G} := (\varepsilon^T | M^T)$, where $\tilde{G}$ is the matrix containing the columns of M^T along with an additional column ε^T. Note that here we index the first block of each block codeword of $\tilde{C}$ by the dealer $\mathcal{D}$ and index the i^{th} block by player P_i, $1 \leq i \leq n$. $\tilde{C}$ can be regarded as an extended block code of C, since each block codeword of $\tilde{C}$ after removing the first block becomes a block codeword of C and conversely, each block codeword of C can be obtained by removing the first block from some codeword of $\tilde{C}$. Nikov *et.al* [45] gave a formula for $\Delta(\tilde{C})$ as described below.

Lemma 3.2.5 (Lemma 2. [45])**.** $\Delta(\tilde{C}) = \mathcal{A}^{\perp} \uplus \{\mathcal{D}\}$*, where* $\mathcal{D}$ *denotes the dealer and* $\mathcal{A}^{\perp}$ *denotes the dual adversary structure.*

By the definition of $\Delta(\tilde{C})$, we know that it is monotone decreasing, but $\mathcal{A}^{\perp} \uplus \{\mathcal{D}\}$ is not monotone decreasing. This contradiction suffices to demonstrate that $\Delta(\tilde{C}) \neq \mathcal{A}^{\perp} \uplus \{\mathcal{D}\}$. In other words, this lemma is *incorrect*. In the following, we will prove that $\Delta(\tilde{C}) = \mathcal{A}^{\perp} \uplus \{\mathcal{D}, \emptyset\}$ provided that there exists an $\mathcal{A}^{\perp}$-non-redundant MSP $\mathcal{M}^{\perp} = (\mathbb{F}, M^{\perp}, \varepsilon, \psi)$ computing $\Gamma^{\perp}$, the dual access structure of Γ, such that the linear code $\tilde{C}$ generated by $\tilde{G}$ and the linear code $\tilde{C}^{\perp}$ generated by $\tilde{G}^{\perp} := (\varepsilon^T|(M^{\perp})^T)$ are dual. (Note that here the two ε's are the target vectors of the corresponding MSPs and they may have different lengths.)

Definition 3.2.6. [45] An MSP $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ computing Γ is called $\mathcal{A}$-non-redundant if on one hand for each $B \in \mathcal{A}$, the rows of M_B are linearly independent, and on the other hand, for each $B \in \Gamma$, the rows of M_B are linearly dependent.

In the following, we fix an MSP $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ computing a given Γ. Let $\mathcal{M}^{\perp} = (\mathbb{F}, M^{\perp}, \varepsilon, \psi)$ be an MSP computing $\Gamma^{\perp}$ with the property that the linear code $\tilde{C}$ generated by $\tilde{G}$ and the linear code $\tilde{C}^{\perp}$ generated by $\tilde{G}^{\perp}$ are dual. In particular, given $\mathcal{M}$, we can construct $\mathcal{M}^{\perp}$ in the following way. Suppose that M has size $m \times e$ as before. Let $\alpha_1, \ldots, \alpha_{m-e}$ be a basis of the vector space $ker(M^T)$ and let α_0 be a solution vector of $M^T \cdot X = \varepsilon^T$, where ε is the target vector for $\mathcal{M}$. Let $M^{\perp} := (-\alpha_0, \alpha_1, \ldots, \alpha_{m-e})$. Clearly, $M^{\perp}$ has size $e \times (m - e + 1)$ and the columns of $M^{\perp}$ are linearly independent. It is proved in [22] that $\mathcal{M}^{\perp} = (\mathbb{F}, M^{\perp}, \varepsilon, \psi)$ is an MSP computing $\Gamma^{\perp}$. Furthermore, one can check by direct computations that $\tilde{G} \cdot (\tilde{G}^{\perp})^T = \mathbf{0}$. Note that the size of $\tilde{G}$ (respectively, $\tilde{G}^{\perp}$) is $e \times (m + 1)$ (respectively, $(m - e + 1) \times (m + 1)$). Hence, the code $\tilde{C}$ and the code $\tilde{C}^{\perp}$ are indeed dual when considered as classical linear codes.

On the other hand, suppose that $\mathcal{M}^{\perp} = (F, M^{\perp}, \varepsilon, \psi)$ is an MSP computing $\Gamma^{\perp}$ with the additional property that the linear code $\tilde{C}$ generated by $\tilde{G}$ and the linear code $\tilde{C}^{\perp}$ generated by $\tilde{G}^{\perp}$ are dual. It follows that $\tilde{G} \cdot (\tilde{G}^{\perp})^T = 0$ and the size of $\tilde{G}^{\perp}$ is $(m-e+1) \times (m+1)$. Consequently, $M^T \cdot M^{\perp} = E$, where E is a zero matrix except for the entry in the upper left corner which is -1. We write $M^{\perp}$ as $(\beta_0, \beta_1, \ldots, \beta_{m-e})$, where β_i is the i^{th} column of $M^{\perp}$, for $0 \leq i \leq m - e$. Clearly, $\beta_0, \beta_1, \ldots, \beta_{m-e}$ are linearly independent. Besides, $(-\varepsilon^T, \mathbf{0}, \ldots, \mathbf{0}) = E = M^T \cdot M^{\perp} = M^T \cdot (\beta_0, \beta_1, \ldots, \beta_{m-e}) = (M^T \cdot \beta_0, M^T \cdot \beta_1, \ldots, M^T \cdot \beta_{m-e})$. In other words, $M^T \cdot (-\beta_0) = \varepsilon^T$ and $M^T \cdot \beta_i = \mathbf{0}$, $1 \leq i \leq m - e$, where $\mathbf{0}$ denotes a zero vector of suitable length. Thus, $\beta_1, \ldots, \beta_{m-e}$ is a basis of the space $ker(M^T)$ and $-\beta_0$ is a solution vector of $M^T \cdot X = \varepsilon^T$.

Let DUAL be the collection of all the matrices $M^{\perp}$ appearing in those MSPs $\mathcal{M}^{\perp} = (F, M^{\perp}, \varepsilon, \psi)$ (computing $\Gamma^{\perp}$) with the property that the code $\tilde{C}$ generated by $\tilde{G}$ and the code $\tilde{C}^{\perp}$ generated by

$\tilde{G}^\perp$ are dual. It follows from the argument above that

$$\text{DUAL} = \{(-\alpha_0, \alpha_1, \ldots, \alpha_{m-e}) : (\alpha_1, \ldots, \alpha_{m-e}) \text{ is a basis of } ker(M^T) \text{ and } M^T \cdot \alpha_0 = \varepsilon^T\}.$$

For any two distinct $M_1^\perp, M_2^\perp \in$ DUAL, say $M_1^\perp = (-\alpha_0, \alpha_1, \ldots, \alpha_{m-e})$, $M_2^\perp = (-\beta_0, \beta_1, \ldots, \beta_{m-e})$, there exists a vector $v = (k_1, \ldots, k_{m-e}) \in \mathbb{F}^{m-e}$ such that $\beta_0 = \alpha_0 + k_1\alpha_1 + \cdots + k_{m-e}\alpha_{m-e}$. Besides, since both $\beta_1, \ldots, \beta_{m-e}$ and $\alpha_1, \ldots, \alpha_{m-e}$ are two basis of the same vector space $ker(M^T)$, there exists an invertible matrix H such that $(\beta_1, \ldots, \beta_{m-e})$= $(\alpha_1, \ldots, \alpha_{m-e}) \cdot H$. Thus, $M_2^\perp = M_1^\perp \cdot \overline{H}$, where $\overline{H} = \begin{pmatrix} 1 & 0 \\ -v^T & H \end{pmatrix}$ is invertible too. Therefore, $\forall w \in \mathbb{F}^m$, $w \cdot (M_1^\perp) = \mathbf{0}$ if and only if $w \cdot (M_2^\perp) = \mathbf{0}$. In particular, it follows that if the collection DUAL contains one matrix such that $M^\perp$ is $\mathcal{A}^\perp$-non-redundant, then all the matrices in this collection are $\mathcal{A}^\perp$-non-redundant. In other words, the $\mathcal{A}^\perp$-non-redundant property is independent of the choices of $M^\perp$ (from DUAL), instead, it essentially depends only on M. Due to this, hereafter we can can choose an arbitrary $M^\perp$ from DUAL and fix it.

We are preceding to give a formula for $\Delta(\tilde{C})$ in the special case when $M^\perp$ is $\mathcal{A}^\perp$-non-redundant, from which we can explore some properties for $\Delta(C)$. More explicitly, we will demonstrate that $\Delta(\tilde{C}) = \mathcal{A}^\perp \uplus \{\mathcal{D}, \emptyset\}$ if and only if $M^\perp$ is $\mathcal{A}^\perp$-non-redundant.

Lemma 3.2.7. *If $\Delta(\tilde{C}) = \mathcal{A}^\perp \uplus \{\mathcal{D}, \emptyset\}$, then $M^\perp$ is $\mathcal{A}^\perp$-non-redundant.*

Proof. First we prove that for any $B \subseteq \mathcal{P}$, if $B \in \mathcal{A}^\perp$, then the rows of $(M^\perp)_B$ are linearly independent. In fact, if $B \in \mathcal{A}^\perp$, then $B \in \Delta(\tilde{C})$ by the assumption. Now suppose that the rows of $(M^\perp)_B$ are linearly dependent. Then there exists a vector w such that $w \cdot M^\perp = 0$ and $\text{supp}(w) \subseteq B$. Thus, $(0, w)$, the concatenation of 0 and w, is a codeword of $\tilde{C}$, since $\tilde{G}^\perp = (\varepsilon^T | (M^\perp)^T)$ is a parity check matrix for $\tilde{C}$. As a result, $B \in \Gamma(\tilde{C})$, which contradicts with the conclusion that $B \in \Delta(\tilde{C})$.

Now we prove that for each $B \in \Gamma^\perp$, the rows of $(M^\perp)_B$ are linearly dependent. In fact, since $B \in \Gamma^\perp$ and $\Delta(\tilde{C}) = \mathcal{A}^\perp \uplus \{\mathcal{D}, \emptyset\}$, $B \notin \Delta(\tilde{C})$. Equivalently, $B \in \Gamma(\tilde{C})$. As a result, there exists a block codeword $v \in \tilde{C}$ such that $\text{supp}(v) \subseteq B$. In other words, $(\varepsilon^T | (M^\perp)^T) \cdot v^T = 0$ and $\text{supp}(v) \subseteq B$, which indeed implies that the rows of $(M^\perp)_B$ are linearly dependent. □

Remark 3.2.8. In the first paragraph of the proof above, we specifically proved that under the assumption that $\Delta(\tilde{C}) \supseteq \mathcal{A}^\perp \uplus \{\emptyset, \mathcal{D}\}$, if $B \in \mathcal{A}^\perp$, then the rows of $M_B^\perp$ are linearly independent. On the other hand, in the second paragraph, we considered the assumption that $\Delta(\tilde{C}) \subseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$ and proved that if $B \in \Gamma^{\perp}$, then the rows of $M_B^{\perp}$ are linearly dependent.

Lemma 3.2.9. *If $M^{\perp}$ is $\mathcal{A}^{\perp}$-non-redundant, then $\Delta(\tilde{C}) = \mathcal{A}^{\perp} \uplus \{\mathcal{D}, \emptyset\}$.*

Proof. We begin by proving that $\Delta(\tilde{C}) \supseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$. Suppose not. Then there exists some B in $\mathcal{A}^{\perp}$ such that either $(B \cup \{\mathcal{D}\}) \in \Gamma(\tilde{C})$ or $B \in \Gamma(\tilde{C})$. In either of the two cases, there exists a codeword w of $\tilde{C}$ such that $\text{supp}(w) \subseteq B \cup \{\mathcal{D}\}$. Since $\mathcal{A}^{\perp}$ is monotone decreasing, we can always assume that either $\text{supp}(w) = B \cup \{\mathcal{D}\}$ or $\text{supp}(w) = B$. However, if $\text{supp}(w) = B \cup \{\mathcal{D}\}$, then ε is a linear combination of the rows of $(M^{\perp})_B$, since $\tilde{G}^{\perp} = (\varepsilon^T | (M^{\perp})^T)$ is a parity check matrix for $\tilde{C}$. Consequently, $B \in \Gamma^{\perp}$, which contradicts with the prerequisite that $B \in \mathcal{A}^{\perp}$. On the other hand, if $\text{supp}(w) = B$, then the rows of $(M^{\perp})_B$ are linearly dependent, which contradicts to our assumption that $\mathcal{M}^{\perp}$ is $\mathcal{A}^{\perp}$-non-redundant. Therefore, $\Delta(\tilde{C}) \supseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$.

Now we prove that $\Delta(\tilde{C}) \subseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$. Suppose not. Then there exists an element X in $\Delta(\tilde{C})$ such that $X \notin \mathcal{A}^{\perp}$ and $X \notin \mathcal{A}^{\perp} \uplus \{\mathcal{D}\}$. There are two possibilities. Case 1: $\mathcal{D} \in X$. Then $X = Y \cup \{\mathcal{D}\}$ for some $Y \in \Gamma^{\perp}$. It follows that ε is a linear combination of the rows of $(M^{\perp})_Y$. Hence, there exists a vector u such that $(\varepsilon^T | (M^{\perp})^T) \cdot u^T = 0$ and $\text{supp}(u) \subseteq X$. Thus $u \in \tilde{C}$ and $\text{supp}(u) \subseteq X$. Consequently, $X \in \Gamma(\tilde{C})$, which is impossible since $X \in \Delta(\tilde{C})$. Case 2: $\mathcal{D} \notin X$. Then $X \in \Gamma^{\perp}$ and as a result, the rows of $(M^{\perp})_X$ are linearly dependent, since it is assumed that $M^{\perp}$ is $\mathcal{A}^{\perp}$-non-redundant. Hence, by a similar argument as in Case 1, we have $X \in \Gamma(\tilde{C})$, which contradicts with the assumption that $X \in \Delta(\tilde{C})$. This completes the proof. □

Remark 3.2.10. In the first paragraph of the proof above, we specifically proved that if the rows of $(M^{\perp})_B$ are linearly independent whenever $B \in \mathcal{A}^{\perp}$, then $\Delta(\tilde{C}) \supseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$. On the other hand, we proved in the second paragraph that if the rows of $(M^{\perp})_B$ are linearly dependent whenever $B \in \Gamma^{\perp}$, then $\Delta(\tilde{C}) \subseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$.

The following corollary is a direct consequence of Remark 3.2.8 and 3.2.10.

Corollary 3.2.11. $\Delta(\tilde{C}) \supseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$ *if and only if it holds that the rows of* $(M^{\perp})_B$ *are linearly independent whenever* $B \in \mathcal{A}^{\perp}$. *On the other side,* $\Delta(\tilde{C}) \subseteq \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\}$ *if and only if it holds that the rows of* $(M^{\perp})_B$ *are linearly dependent whenever* $B \in \Gamma^{\perp}$.

Theorem 3.2.12. $\Delta(\tilde{C}) = \mathcal{A}^{\perp} \uplus \{\mathcal{D}, \emptyset\}$ *if and only if* $M^{\perp}$ *is* $\mathcal{A}^{\perp}$*-non-redundant.*

Proof. The proof follows directly from Lemma 3.2.7 and 3.2.9. □

Now we are well prepared to further explore from $\Delta(\tilde{C})$ some properties for $\Delta(C)$. First, we need to explore a connection between C and $\tilde{C}$, from which we can dig out a relation between $\Gamma(C)$ and $\Gamma(\tilde{C})$ and consequently, a relation between $\Delta(C)$ and $\Delta(\tilde{C})$.

Recall that the block code $\tilde{C}$ is generated by $\tilde{G} = (\varepsilon^T|M^T)$ and that C is the block code generated by $G = M^T$. Thus, C can be obtained from $\tilde{C}$ by deleting the first coordinate from each codeword in $\tilde{C}$, that is,

$$C = \{(X_{P_1}, \ldots, X_{P_n}) : \text{ there exists some } X_{\mathcal{D}} \in \mathbb{F} \text{ such that } (X_{\mathcal{D}}, X_{P_1}, \ldots, X_{P_n}) \in \tilde{C}\}.$$

It is easy to see that $\Gamma(C) = \{A \setminus \{\mathcal{D}\} | A \in \Gamma(\tilde{C})\}$, since $\Gamma(C)$ is the collection of all the supersets of supp(ω), where ω runs over all the nonzero block codewords of C, while $\Gamma(\tilde{C})$ is the collection of all the supersets of supp(ν), where ν runs over all the nonzero block codewords of $\tilde{C}$.

The following lemma describes a relation between $\Delta(C)$ and $\Delta(\tilde{C})$.

Lemma 3.2.13. $\{A \setminus \{\mathcal{D}\} : A \in \Delta(\tilde{C})$ *and* $\mathcal{D} \in A\} \subseteq \Delta(C) \subseteq \{A \setminus \{\mathcal{D}\} : A \in \Delta(\tilde{C})\}$.

Proof. First we prove that $\Delta(C) \subseteq \{A \setminus \{\mathcal{D}\} : A \in \Delta(\tilde{C})\}$. For any $B \in \Delta(C)$, $B \notin \Gamma(C)$. Hence, for each $B \in \Delta(C)$, it holds for any $A \in \Gamma(\tilde{C})$ that $B \neq A \setminus \{\mathcal{D}\}$, since $\Gamma(C) = \{A \setminus \{\mathcal{D}\} : A \in \Gamma(\tilde{C})\}$. Now the equality $B = B \setminus \{\mathcal{D}\}$ implies that $B \in \Delta(\tilde{C})$. Thus, $\Delta(C) \subseteq \{A \setminus \{\mathcal{D}\} : A \in \Delta(\tilde{C})\}$.

It remains to prove that $\{A \setminus \{\mathcal{D}\} : A \in \Delta(\tilde{C})$ and $\mathcal{D} \in A\} \subseteq \Delta(C)$. Suppose not. There exists $A \in \Delta(\tilde{C})$ with $\mathcal{D} \in A$ such that $A \setminus \{\mathcal{D}\} \in \Gamma(C)$. Then $A \setminus \{\mathcal{D}\} = B \setminus \{\mathcal{D}\}$ for some $B \in \Gamma(\tilde{C})$, since $\Gamma(C) = \{A \setminus \{\mathcal{D}\} : A \in \Gamma(\tilde{C})\}$. If $\mathcal{D} \in B$, then $A = B \in \Gamma(\tilde{C})$, which leads to a contradiction since $A \in \Delta(\tilde{C})$. If $\mathcal{D} \notin B$, then $A = B \cup \{\mathcal{D}\} \in \Gamma(\tilde{C})$, since $B \in \Gamma(\tilde{C})$ and $\Gamma(\tilde{C})$ is monotone increasing, which also introduces a contradiction. This completes the proof. □

It can be checked directly from Example 3.2.3 that for Shamir's t-out-of-n secret sharing scheme, $\Delta(C) = \mathcal{A}^{\perp}$ and $\Gamma(C) = \Gamma^{\perp}$. However, the following lemmas indicate that these two equalities do not hold in general case.

Lemma 3.2.14. $\{A \cup \{\mathcal{D}\} : A \in \Gamma^{\perp}\} \subseteq \Gamma(\tilde{C})$. *As a result,* $\Gamma^{\perp} \subseteq \Gamma(C)$. *Equivalently,* $\Delta(C) \subseteq \mathcal{A}^{\perp}$.

Proof. For any $A \in \Gamma^{\perp}$, ε is a linear combination of the rows of $(M^{\perp})_A$, that is, there exists a vector w such that $(\varepsilon^T|(M^{\perp})^T) \cdot w^T = 0$ and $\{\mathcal{D}\} \subseteq \text{supp}(w) \subseteq (A \cup \{\mathcal{D}\})$. Hence $(A \cup \{\mathcal{D}\}) \in \Gamma(\tilde{C})$. Therefore, $\{A \cup \{\mathcal{D}\} : A \in \Gamma^{\perp}\} \subseteq \Gamma(\tilde{C})$. Now the relation $\Gamma^{\perp} \subseteq \Gamma(C)$ follows directly from the equality that $\Gamma(C) = \{A \setminus \{\mathcal{D}\} : A \in \Gamma(\tilde{C})\}$. □

Lemma 3.2.15. *$\mathcal{A}^{\perp} \subseteq \Delta(C)$ (equivalently, $\Gamma(C) \subseteq \Gamma^{\perp}$) if and only if the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}^{\perp}$.*

Proof. From Corollary 3.2.11, we know that $\mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\} \subseteq \Delta(\tilde{C})$ if and only if the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}^{\perp}$. Hence,

$$\begin{aligned}
&\forall B \in \mathcal{A}^{\perp}, \text{ the rows of } (M^{\perp})_B \text{ are linearly independent}\\
&\Rightarrow \mathcal{A}^{\perp} \uplus \{\emptyset, \mathcal{D}\} \subseteq \Delta(\tilde{C})\\
&\Rightarrow \mathcal{A}^{\perp} \subseteq \Delta(C)\\
&\Rightarrow \forall B \in \mathcal{A}^{\perp}, \text{ the rows of} (M^{\perp})_B \text{ are linearly independent}
\end{aligned}$$

where the second "$\Rightarrow$" is due to Lemma 3.2.13. The last "$\Rightarrow$" holds because otherwise, there exists a vector w such that $(\varepsilon^T | (M^{\perp})^T) \cdot w^T = 0$ and $\mathrm{supp}(w) \subseteq B$, i.e., w is a codeword of $\tilde{C}$ and $\mathrm{supp}(w) \subseteq B$. Hence, $B \in \Gamma(\tilde{C})$, which implies that $B = B \setminus \{\mathcal{D}\} \in \Gamma(C)$. Thus $B \notin \Delta(C)$, which further implies that $B \notin \mathcal{A}^{\perp}$, since $\mathcal{A}^{\perp} \subseteq \Delta(C)$. However, this is impossible, since $B \in \mathcal{A}^{\perp}$. This contradiction completes the proof. □

Proposition 3.2.16. *$\Delta(C) = \mathcal{A}^{\perp}$ (equivalently, $\Gamma(C) = \Gamma^{\perp}$) if and only if the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}$.*

Proof. The proof follows immediately from Lemmas 3.2.14 and 3.2.15. □

At this point, we are ready for proposing an equivalent condition for $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ merely from the perspective of $M^{\perp}$ and $\mathcal{A}$.

Theorem 3.2.17. *$\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ if and only if $\mathcal{A}$ is Q^3 and for any $B \in \mathcal{A} \uplus \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent.*

Proof. Suppose that $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. First, we prove that $\mathcal{A}$ is Q^3.

$$\begin{aligned}
&\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)\\
&\Rightarrow \mathcal{A} \uplus \mathcal{A} \subseteq \mathcal{A}^{\perp} && \text{by Lemma 3.2.14}\\
&\Leftrightarrow \forall B_1, B_2 \in \mathcal{A}, (B_1 \cup B_2) \in \mathcal{A}^{\perp}\\
&\Leftrightarrow \forall B_1, B_2 \in \mathcal{A}, (B_1 \cup B_2) \notin \Gamma^{\perp}\\
&\Leftrightarrow \forall B_1, B_2 \in \mathcal{A}, (B_1 \cup B_2)^c \in \Gamma\\
&\Leftrightarrow \mathcal{A} \text{ is } Q^3.
\end{aligned}$$

Now we prove that for any $B \in \mathcal{A} \uplus \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent. Suppose not. Then the rows of $(M^{\perp})_B$ are linearly dependent, for some $B \in \mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$. Thus, there exists some vector ω such that $\omega \cdot M^{\perp} = \mathbf{0}$ and $\text{supp}(\omega) \subseteq B$. It follows that $(\varepsilon^T | (M^{\perp})^T) \cdot (0, \omega)^T = \mathbf{0}$, namely, $(0, \omega)$ is a codeword of $\tilde{C}$, which further implies that $\omega \in C$. As a result, $B \in \Gamma(C)$, which contradicts the assumption that $B \in \Delta(C)$.

Conversely, suppose that $\mathcal{A}$ is Q^3 and for any $B \in \mathcal{A} \uplus \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent. Assume that the inclusion relation $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ does not hold. Then there exists some $B \in \mathcal{A} \uplus \mathcal{A}$ such that $B \in \Gamma(C)$. Hence, there exists a codeword $\upsilon \in C$ such that $\text{supp}(\upsilon) \subseteq B$. On the other hand, the argument above indicates that $\mathcal{A} \uplus \mathcal{A} \subseteq \mathcal{A}^{\perp}$, since $\mathcal{A}$ is Q^3. As a result, $B \in \mathcal{A}^{\perp}$. Now due to the relation between C and $\tilde{C}$, there is an element $a \in \mathbb{F}$ such that (a, υ) is a codeword of $\tilde{C}$, that is, $(\varepsilon^T | (M^{\perp})^T) \cdot (a, \upsilon)^T = \mathbf{0}$. We claim that $a = 0$. Suppose not. Then ε is a linear combination of the rows of $(M^{\perp})_B$, that is, $B \in \Gamma^{\perp}$, which is not true, since $B \in \mathcal{A}^{\perp}$. Consequently, $(\varepsilon^T | (M^{\perp})^T) \cdot (0, \upsilon)^T = \mathbf{0}$, which indicates that the rows of $(M^{\perp})_B$ are linearly dependent. This contradiction completes the proof. □

Remark 3.2.18. Theorem 3.2.17 indicates that $\mathcal{M}$ can correct all error patterns in $\mathcal{A}$ if and only if $\mathcal{A}$ is Q^3 and for any $B \in \mathcal{A} \uplus \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent. Thus, for any Q^3 adversary structure $\mathcal{A}$, in order to construct an MSP computing Γ such that it can correct all error patterns in $\mathcal{A}$, a possible strategy is to construct an MSP for $\Gamma^{\perp}$ with the property that the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A} \uplus \mathcal{A}$. However, whether we can always succeed (in constructing such an MSP for $\Gamma^{\perp}$) or how we can ascertain this desired property except by trial and error remains open. On the other hand, suppose for any given Γ, we have already constructed an MSP computing it such that the condition $\mathcal{A} \uplus \mathcal{A} \subseteq \Delta(C)$ is satisfied. If we can find an efficient algorithm to execute the decoding process mentioned in Remark 3.2.4, then we can use this technique to construct non-interactive verifiable secret sharing schemes as well as unconditionally secure multiparty computation protocols. Besides, we can construct an efficient one-round protocol for perfectly secure message transmission tolerating any Q^3 adversary structure.

3.3 $\mathcal{A}$-Error Detectable Linear Secret Sharing Schemes

We begin with a famous result from coding theory. Let C be a classical code with minimal distance d. Suppose that a codeword W is transmitted and W' is received. Here W can be different from W' due to channel noise, which can inject errors to some random chosen positions of W. If the hamming weight of W' is less than d, then W' can not be a codeword and consequently, some errors must have occurred during the process of transmission. In this sense, we say that C can detect $d-1$ errors. Obviously, the minimal distance of a classical code measures its error detecting-ability: the larger the minimal distance, the larger the number of errors it can detect. However, as mentioned in the previous section, in the model of secret sharing, we assume that all the communication channels are authenticated and hence there is no noise. Instead, an adversary may corrupt some subset $A \in \mathcal{A}$ of players by requiring them to send fake shares in the secret reconstruction phase. In this situation, the error positions are not totally random but instead, the index set of error positions can only be in $\mathcal{A}$. In this section, we make an attempt to characterize the error-detecting ability of any linear block code by its set of forbidden distances.

Definition 3.3.1. Let Γ be an access structure and $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP computing Γ. Let C be the linear block code generated by M^T. We say that $\mathcal{M}$ is $\mathcal{A}$-*error detectable* if C is $\mathcal{A}$-error detectable (or C can detect all error patterns in $\mathcal{A}$), that is, for any nonzero vector $W = (W_{P_1}, \ldots, W_{P_n}) \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ with $\text{supp}(W) \in \mathcal{A}$, ω can not be a block codeword of C.

Remark 3.3.2. Due to the relation between Shamir's secret sharing schemes and generalized Reed-Solomon codes, Shamir's t-out-of-n secret sharing scheme is $\mathcal{A}$-error detectable, whenever $n \geq 2t-1$ (that is, whenever the corresponding adversary structure is Q^2.)

As before, let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP computing Γ and C be the linear block code generated by M^T. Recall that we regard each codeword in C as a block codeword, namely, as a tuple in $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$, where $m_i = |\psi^{-1}(P_i)|$ for $1 \leq i \leq n$. In addition, for any $W = (W_{P_1}, \ldots, W_{P_n}) \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$, $\text{supp}(W) := \{P_i | W_{P_i} \neq \mathbf{0}\}$. Let $\Gamma(C)_{\min} := \{B \subseteq \mathcal{P} | B = \text{supp}(W)$ for some nonzero codeword $W \in C\}$.

In the remainder of this section, we are dedicated to exploring equivalent conditions for an MSP $\mathcal{M}$ to be $\mathcal{A}$-error detectable. First, we give an equivalent condition from the perspective of $\Delta(C)$, the set of all forbidden distances of C.

Proposition 3.3.3. *$\mathcal{M}$ is $\mathcal{A}$-error detectable if and only if $\mathcal{A} \subseteq \Delta(C)$.*

Proof. First we show that if $\mathcal{M}$ is $\mathcal{A}$-error detectable, then $\mathcal{A} \subseteq \Delta(C)$. By assumption, for each nonempty $B \in \mathcal{A}$ and for any $W = (W_{P_1}, \ldots, W_{P_n}) \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ with $\mathrm{supp}(W) = B$, W can not be a codeword of C. In other words, for each $B \in \mathcal{A}$, $B \notin \Gamma(C)_{\min}$. Suppose that $B \in \Gamma(C)$. By definition, there exists $B' \subseteq B$ such that $B' = \mathrm{supp}(V)$, for some $\mathbf{0} \neq V \in C$, which contradicts with the assumption that $\mathcal{M}$ is $\mathcal{A}$-error detectable, since $B' \in \mathcal{A}$. Hence, $\mathcal{A} \subseteq \Delta(C)$.

Conversely, the inclusion relation $\mathcal{A} \subseteq \Delta(C)$ implies that $\mathcal{A} \cap \Gamma(C) = \emptyset$, which further implies that $\mathcal{A} \cap \Gamma(C)_{\min} = \emptyset$. Consequently, $\mathcal{M}$ is $\mathcal{A}$-error detectable. □

Now we are going to present an equivalent condition for an MSP $\mathcal{M} = (\mathbb{F}, M, \psi, \varepsilon)$ computing Γ to be $\mathcal{A}$-error detectable merely from the perspective of $M^{\perp}$ and $\mathcal{A}$.

Proposition 3.3.4. *If $\mathcal{M}$ is $\mathcal{A}$-error detectable, then $\mathcal{A}$ is Q^2 and for any $B \in \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent.*

Proof. If $\mathcal{M}$ is $\mathcal{A}$-error detectable, then $\mathcal{A} \subseteq \Delta(C)$ by Proposition 3.3.3. On the other hand, $\Delta(C) \subseteq \mathcal{A}^{\perp}$ by Lemma 3.2.14. As a result, $\mathcal{A} \subseteq \mathcal{A}^{\perp}$. Now, the statement that $\mathcal{A}$ is Q^2 follows directly from the inclusion relation $\mathcal{A} \subseteq \mathcal{A}^{\perp}$.

We proceed to show that for any $B \in \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent. Suppose that the rows of $(M^{\perp})_B$ are linearly dependent for some $B \in \mathcal{A}$. Then, there exists some ω such that $\omega \cdot M^{\perp} = \mathbf{0}$ and $\mathrm{supp}(\omega) \subseteq B$. It follows that $(\varepsilon^T | (M^{\perp})^T) \cdot (0, \omega)^T = \mathbf{0}$, namely, $(0, \omega)$ is a codeword of $\tilde{C}$, which further implies that $\omega \in C$. As a result, $B \in \Gamma(C)$, which contradicts the assumption that $B \in \mathcal{A} \subseteq \Delta(C)$. This completes the proof.

□

Proposition 3.3.5. *If $\mathcal{A}$ is Q^2 and for each $B \in \mathcal{A}$, the rows of $(M^{\perp})_B$ are linearly independent, then $\mathcal{M}$ is $\mathcal{A}$-error detectable.*

Proof. Suppose not, that is, there exists some $B \in \mathcal{A}$ such that $B \in \Gamma(C)$. Thus, there exists a codeword ω of C such that $\mathrm{supp}(\omega) \subseteq B$. In other words, $(\varepsilon^T | (M^{\perp})^T) \cdot \omega^T = \mathbf{0}$ and $\mathrm{supp}(\omega) \subseteq B$, which in fact indicates that the rows of $(M^{\perp})_B$ are linearly dependent. This contradicts with the prerequisite of this proposition. □

We consolidate the above results in the next theorem, whose proof is immediate.

Theorem 3.3.6. *Let $\mathcal{M}$ be an LSSS with adversary structure $\mathcal{A}$. Then $\mathcal{M}$ is $\mathcal{A}$-error detectable if and only if $\mathcal{A}$ is Q^2 and the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}$.*

Remark 3.3.7. Theorem 3.3.6 indicates that $\mathcal{M}$ can detect all error patterns in $\mathcal{A}$ if and only if $\mathcal{A}$ is Q^2 and for any $B \in \mathcal{A}$, the rows of $(M^\perp)_B$ are linearly independent. Thus, for any Q^2 access structure Γ, in order to construct an MSP for it such that it can detect all error patterns in $\mathcal{A}$, a possible strategy is to construct an MSP for $\Gamma^\perp$ with the property that the rows of $(M^\perp)_B$ are linearly independent for any $B \in \mathcal{A}$. However, whether we can always succeed (in constructing such an MSP for $\Gamma^\perp$) or how we can ascertain this desired property except by trial and error remains open.

4. GAME THEORETIC DEFINITIONS

In this chapter, we will introduce basic notions and definitions from game theory that will be needed to develop and understand rational cryptography. Briefly, we will describe two categories of games: normal form games and extensive games with perfect information, as well as the definitions for Nash equilibrium and its variants. Besides, some examples are demonstrated to make those definitions more pellucid. Most of the definitions are based on [49] and the readers can refer to this book for further details.

4.1 Normal Form Games

A *normal form game* is a model of decision-making in which each player chooses his action once and for all in such a way that these choices are made simultaneously and independently. Here by simultaneously, we mean that each player must choose his action without seeing the others' actions. This model consists of a set $\mathcal{P} := \{P_1, \ldots, P_n\}$ of finitely many players, and for each player P_i, a set A_i of actions along with a utility function that represents his preference on action profiles (or outcomes). Here an action profile $(a_i)_{P_i \in \mathcal{P}}$, where $a_i \in A_i$, is regarded as an outcome of the game. The formal definition is as follows.

Definition 4.1.1. [49] An n-player game $\mathcal{G} = (\mathcal{P}, \{A_1, \ldots, A_n\}, \{U_1, \ldots, U_n\})$, presented in *normal/standard form*, is determined by specifying for each player P_i, a set of possible *actions* or *strategies* A_i and a *utility function* $U_i : A_1 \times \cdots \times A_n \to \mathbb{R}$. Any tuple of actions $a := (a_1, \ldots, a_n) \in A_1 \times \cdots \times A_n$ is called an *outcome* or an *action profile.*

The game $\mathcal{G}$ is played as follows: each player P_i chooses an action $a_i \in A_i$ independently and simultaneously and all players play their actions simultaneously. The *payoff* (or *utility*) to P_i from the outcome $(a_1, \ldots, a_n)$ is the value given by his utility function: $U_i(a_1, \ldots, a_n)$. Observe that each player's utility function essentially expresses his preferred choices over outcomes. The goal of each participant of the game is to maximize his payoff.

Definition 4.1.2. [49] A player P_i *prefers* (respectively, *weakly prefers*) outcome a to a' if and only if $U_i(a) > U_i(a')$ (respectively, $U_i(a) \geq U_i(a')$).

A normal form game in which there are two players can be described in a table as the one shown in Figure 4.1.2, where the rows are indexed by one player's possible actions, (this player is called the row player accordingly) and the columns are indexed by the other player's possible actions (this player is called the column player accordingly). The two numbers in the box indexed by row r and column c represent the players' payoffs when the row player chooses r and the column player chooses c, where the first component denotes the payoff of the row player and the second component denotes the payoff of the column player. Thus in the game in Figure 4.1.2, the set of actions of the row player is $\{U, D\}$ and that of the column player is $\{L, M, R\}$. For example, the row player's payoff from the outcome $\{U, L\}$ is b_1 and the column player's payoff is b_2.

	L	M	R
U	b_1, b_2	c_1, c_2	d_1, d_2
D	x_1, x_2	y_1, y_2	z_1, z_2

Figure 4.1.2

Remark 4.1.3. A normal form game models an event that occurs only once. The details of the game and the fact that all the players are *rational* are public knowledge. Each player has no idea about the choices being made by other players when making his own choice, since they are required to choose their actions simultaneously and independently.

Given a game $\mathcal{G}$, a solution concept is essentially a way of predicting how $\mathcal{G}$ will be played. The most commonly used solution concept in game theory is Nash Equilibrium, which captures a *steady state* of the play of a normal form game in which each player holds some belief about the other players' behavior and acts rationally.

Definition 4.1.4. [49] A *Nash equilibrium* of a normal form game $\mathcal{G} = (\mathcal{P}, \{A_1, \ldots, A_n\}, \{U_1, \ldots, U_n\})$ is an action profile $a = (a_1, \ldots, a_n) \in A_1 \times \cdots \times A_n$ with the property that for every player $\mathcal{P}$, it holds that

$$U_i(a) \geq U_i(a'_i, a_{-i}) \text{ for all } a'_i \in A_i,$$

where $a_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$ denotes a tuple consisting of each player's action in a excluding that of player P_i.

Remark 4.1.5. Intuitively, Definition 4.1.4 indicates that each P_i can maximize his utility by choosing a_i, provided that others will play a_{-i}. In other words, each P_i has no incentive to deviate from a_i as long as the remaining participants follow a_{-i} (for all $i \in \{1, \ldots, n\}$). The essential intuition behind a Nash equilibrium is that it may prevent a single player's deviations in the following situation: if all players are told which Nash equilibrium is being played and assuming that all other players stick to their prescribed strategies, it will be irrational for a single player to play anything other than his prescribed strategy. However, the notion of Nash equilibrium tells nothing about how the players agree on the particular equilibrium to play. This is problematic when a game has more than one equilibrium, since each player can not predict for certain which of them will be played. Take a game with only two players as an example. If player P_1 believes that an equilibrium (a_1, a_2) will be played, while player P_2 believes (a'_1, a'_2), then it is likely that the action profile ultimately played is (a_1, a'_2), which needs not to be an equilibrium at all.

Example 4.1.6. A trusted party shares a secret s between two players P_1, P_2 by splitting s into two parts: $s = s_1 + s_2$, where s_i is given to P_i secretly as his share. For simplicity, here we assume that each share carries with a verification message, which can be used to verify the correctness of it whenever it is broadcast. Thus, we do not distinguish the action *Keep silent* from the action *Broadcast a fake share.* Now suppose that the two players want to play a game to recover the secret s. Assume that they are selfish and the main concern of each player is to gain the secret while preventing the other from gaining it. In this situation, for each P_i, $i = 1, 2$, the set of actions is $\{$*Broadcast, Keep silent*$\}$. Representing the individuals' preferences by payoff functions, we can describe the game as in Figure 4.1.6, where $u_i^- < u_i < u_i^+, i = 1, 2$.

	Broadcast	*Keep silent*
Broadcast	u_1, u_2	u_1^-, u_2^+
Keep silent	u_1^+, u_2^-	u_1^-, u_2^-

Figure 4.1.6

Obviously, in this game, P_1 prefers the outcome (*Keep silent, Broadcast*) while P_2 prefers the outcome (*Broadcast, Keep silent*). However, the two preferences are conflict: neither of them can make both players satisfied. On the other hand, this game has a unique Nash equilibrium (*Keep silent, Keep silent*), which leads to the *bad* outcome in which nobody learns the secret.

Example 4.1.7. [49] *Matching Pennies* Each of two players chooses either $Head$ or $Tail$. If the choices differ, P_1 pays P_2 one dollar; otherwise, P_2 pays P_1 one dollar. Each person cares only about the money he receives. A game that models this situation is shown in Figure 4.1.7 illustrated below.

	$Head$	$Tail$
$Head$	$1, -1$	$-1, 1$
$Tail$	$-1, 1$	$1, -1$

Figure 4.1.7

It can be checked directly that this game has no Nash equilibrium.

4.2 Mixed Extension of a Normal Form Game and Mixed Strategy Nash Equilibrium

In order to obtain stable strategies (that is, Nash equilibria), some randomization in the choice of strategies (or actions) is needed. In this section, we introduce the concept of mixed strategy Nash equilibrium in which the players' actions are not deterministic. In fact, this notion is proposed to model a steady state of a game in which players' choices are regulated by probabilistic distributions.

- Each player P_i chooses his action $a_i \in A_i$ using a *distribution* σ_i over A_i.

- We are interested in the *expected utilities* for each player.

Definition 4.2.1. [49] The *mixed extension of a normal form game* $\mathcal{G} = (\mathcal{P}, \{A_1, \ldots, A_n\}, \{U_1, \ldots, U_n\})$ is the strategic game $\tilde{\mathcal{G}} = (\mathcal{P}, \{\nabla(A_1), \ldots, \nabla(A_n)\}, \{u_1, \ldots, u_n\})$, where $\nabla(A_i)$ is the set of probability distributions over A_i, and $u_i : \times_{P_j \in \mathcal{P}} \nabla(A_j) \rightarrow \mathbb{R}$ assigns to each $\sigma = (\sigma_1, \ldots, \sigma_n) \in \times_{P_j \in \mathcal{P}} \nabla(A_j)$ the expected value under U_i over A that is induced by σ as follows:

$$u_i(\sigma) = \sum_{a \in A} (\prod_{P_j \in \mathcal{P}} \sigma_j(a_j)) U_i(a),$$

where A is assumed to be a finite set and $\sigma_j(a_j)$ denotes the probability that σ_j assigns to $a_j \in A_j$. Each $\sigma_i \in \nabla(A_i)$ is called a strategy of player P_i, $1 \leq i \leq n$. In the special case that σ_i assigns probability 1 to some single action, σ_i is called a pure strategy; otherwise, it is called a mixed strategy. (Note that we regard each action in a normal form game as a pure strategy.) A tuple of strategies $\sigma = (\sigma_1, \ldots, \sigma_n)$ is called a strategy profile.

Definition 4.2.2. [49] Let $\tilde{\mathcal{G}} = (\mathcal{P}, \{\nabla(A_1), \ldots, \nabla(A_n)\}, \{u_1, \ldots, u_n\})$ be the mixed extension of a normal form game. Consider a strategy profile $\sigma = (\sigma_1, \ldots, \sigma_n)$. σ_i is a *best response* of P_i to σ_{-i} if it maximizes $u_i(\sigma_i, \sigma_{-i})$, where σ_{-i} represents the $(n-1)$-tuple of strategies played by the remaining players.

Definition 4.2.3. [49] Let $\tilde{\mathcal{G}} = (\mathcal{P}, \{\nabla(A_1), \ldots, \nabla(A_n)\}, \{u_1, \ldots, u_n\})$ be a mixed extension of a normal form game $\mathcal{G}$ and let $\sigma_i \in \nabla(A_i)$ be a distribution over A_i. A tuple $\sigma = (\sigma_1, \ldots, \sigma_n)$ is a *mixed strategy Nash equilibrium* of $\mathcal{G}$ if it is a Nash equilibrium of its mixed extension $\tilde{\mathcal{G}}$, that is, for each $P_i \in \mathcal{P}$ and every distribution $\sigma_i' \in \nabla(A_i)$, we have:

$$u_i(\sigma_i', \sigma_{-i}) \leq u_i(\sigma_i, \sigma_{-i}),$$

that is, for each $i \in \{1, \ldots, n\}, \sigma_i$ is a best response to σ_{-i}.

Remark 4.2.4. The notion of mixed strategy Nash equilibrium is designed to model a steady state of a game in which the players' choices are not determined but are regulated by some probabilistic rules. In other words, in a mixed strategy Nash equilibrium, the players' actions are not deterministic and this notion captures a steady state of the play in which each player holds the correct expectation about the other players' behavior and acts rationally. A mixed strategy $\sigma = (\sigma_1, \ldots, \sigma_n)$ is said to be degenerate if each σ_i assigns probability 1 to some single action; otherwise, it is non-degenerate. In this sense, each Nash equilibrium of a normal form game can be regarded as a degenerate mixed strategy Nash equilibrium or a pure strategy Nash equilibrium.

Theorem 4.2.5. [49] *Any game with a finite set of players and a finite set of strategies has a mixed strategy Nash equilibrium.*

Proposition 4.2.6. [49] *Let $\tilde{\mathcal{G}} = (\mathcal{P}, \{\nabla(A_1), \ldots, \nabla(A_n)\}, \{u_1, \ldots, u_n\})$ be a mixed extension of a normal form game $\mathcal{G}$. Then $\sigma = (\sigma_1, \ldots, \sigma_n) \in \times_{i \in \{1,\ldots,n\}} \nabla(A_i)$ is a mixed strategy Nash equilibrium if and only if for every $i \in \{1, \ldots, n\}$ every pure strategy in the support of σ_i is a best response to σ_{-i}. Here the support of σ_i is defined to be the set of actions $a_i \in A_i$ for which $\sigma_i(a_i) > 0$, and each action in A_i is regarded as a pure strategy automatically, $1 \leq i \leq n$.*

Example 4.2.7. [49] As mentioned previously that the game in Example 4.1.7 has no pure strategy Nash equilibrium. However, it has at least one mixed strategy Nash equilibrium by Theorem 4.2.5. Now we attempt to find all mixed strategy Nash equilibria. Suppose that (σ_1, σ_2) is a mixed strategy Nash equilibrium. Denote $Head$ as H and $Tail$ as T for simplicity. Since this game has no (pure strategy) Nash equilibrium, $0 < \sigma_1(H) < 1$. By Proposition 4.2.6, player P_1's actions H and T are both best responses to σ_2. Thus, $1 - 2\sigma_2(H) = 2\sigma_2(H) - 1$, since $P_1's$ expected utility by executing H is $\sigma_2(H) \cdot 1 + \sigma_2(T) \cdot (-1) = \sigma_2(H) + (1 - \sigma_2(H)) \cdot (-1) = 2\sigma_2(H) - 1$ and his expected utility by executing T is $\sigma_2(H) \cdot (-1) + \sigma_2(T) \cdot 1 = -\sigma_2(H) + (1 - \sigma_2(H)) = 1 - 2\sigma_2(H)$. As a result, $\sigma_2(H) = \frac{1}{2} = \sigma_2(T)$. By a similar argument, we can derive that $\sigma_1(H) = \frac{1}{2} = \sigma_1(T)$. Thus, the only mixed strategy Nash equilibrium of the game is (σ_1, σ_2), where $\sigma_i(H) = \frac{1}{2} = \sigma_i(T)$, $i = 1, 2$. For simplicity, we can denote this mixed strategy Nash equilibrium by $((\frac{1}{2}, \frac{1}{2}), (\frac{1}{2}, \frac{1}{2}))$.

4.3 Extensive Games with Perfect Information

As mentioned previously, in a normal form game, each player makes his action independently and simultaneously and no player is informed of the choice of any other player prior to making his own decision. On the other hand, an extensive game is a detailed description of the *sequential structure* of the decision problems encountered by the players in a game. An extensive game with perfect information is a game in which players take turns to choose their actions and the outcome of the game is determined only after all of the actions are chosen. Here, by perfect information, we mean that each player is aware of all the actions that have occurred prior to his own move. In our definition, we allow each player to make a move at every discrete point in time. Formally,

Definition 4.3.1. [49] An *extensive game with perfect information* is specified by the following components.

- A finite set $\mathcal{P}$ of n players.

A history is a sequence of vectors, where the components of each vector a^L are the actions of players at time L. Each history represents a state that may occur during the procession of the game play.

- A set H of histories (finite or infinite) satisfying the following three properties.
  - The empty sequence $\emptyset$ is a member of H.

- If a history $(a^1, \ldots, a^K) \in H$, then for any $L < K$, $(a^1, \ldots, a^L) \in H$.
- If an infinite sequence $(a^1, a^2, \ldots)$ satisfies $(a^1, \ldots, a^L) \in H$ for every positive integer L, then $(a^1, a^2, \ldots) \in H$.

A history $(a^1, \ldots, a^K) \in H$ is *terminal* if it is infinite or if there is no a^{K+1} such that $(a^1, \ldots, a^K, a^{K+1}) \in H$. The set of terminal histories is denoted by Z. A game is over when a terminal history is reached.

- For each $1 \leq i \leq n$, a payoff function u_i which maps a terminal history to a real number representing player P_i's utility: $u_i : Z \to \mathbb{R}$.
- A function P that assigns to each nonterminal history a member of $\mathcal{P}$. Here P is called the player function and $P(h)$ denotes the player who takes an action after the history h.

In order to play an extensive game, each player must know the actions available to him at each discrete time L. The only way for a player to learn his action from the extensive game is to search through H for valid histories of length $L + 1$ that contain the current history as a subsequence. However, this process may take time exponential in the length of the game. In order to simplify this problem, we have the following definition.

Definition 4.3.2. [49] A *strategy of player $P_i \in \mathcal{P}$* in an extensive game with perfect information $< \mathcal{P}, H, P, (u_i) >$ is a function σ_i that assigns an action in $A(h)$ to each nonterminal history $h \in H \setminus Z$ for which $P(h) = P_i$, where $A(h) := \{a : (h, a) \in H\}$.

A strategy of a player in an extensive game with perfect information is an instruction that allocates the action chosen by him for every history after which it is his turn to move. An extensive form game with perfect information is played as follows: each player P_i independently selects the action $\sigma_i(\emptyset)$. Each player is then informed of the current history h. If h is terminal, each player receives utility $u_i(h)$. If h is non-terminal and $P(h) = P_i$, for some $1 \leq i \leq n$, then player P_i selects the action $\sigma_i(h)$ and the process repeats. Note that it is possible for an extensive form game to be infinite. A game is infinite if and only if it has at least one branch that does not terminate in any finite number of steps.

Definition 4.3.3. [49] A *Nash equilibrium of an extensive game with perfect information*

$< \mathcal{P}, H, P, (u_i) >$ is a strategy profile σ such that for every player P_i, we have

$$u_i(\sigma_i, \sigma_{-i}) \geq u_i(\sigma_i', \sigma_{-i}), \qquad \text{for every strategy } \sigma_i' \text{ of player } P_i.$$

A variation of extensive games with perfect information defined above are extensive games with imperfect information, in which players may not be aware of the actions other players take before he chooses his action. Here we do not give details for this kind of game, since we only focus on extensive games with perfect information.

4.4 More Equilibria

Given a game $< \mathcal{P}, H, P, (u_i) >$, denote by Ω_i the set of all the possible strategies for P_i, $1 \leq i \leq n$. Note that for normal form games, $\Omega_i = A_i$ while for the mixed extensions of normal form games, $\Omega_i = \nabla(A_i)$. Now we define the remaining equilibria only for extensive games with perfect information. But it should be clear that all equilibria apply to normal form games and the mixed extension of normal form games as well.

Definition 4.4.1. [49] Given $< \mathcal{P}, H, P, (u_i) >$, we say that a strategy $\sigma_i' \in \Omega_i$ is *weakly dominated* with respect to $\Omega_{-i} (= \underset{j \neq i}{\times} \Omega_j)$ if there exists a strategy $\sigma_i \in \Omega_i$ such that:

1. $\forall \alpha_{-i} \in \Omega_{-i} : u_i(\sigma_i, \alpha_{-i}) \geq u_i(\sigma_i', \alpha_{-i})$,

2. $\exists \alpha_{-i} \in \Omega_{-i} : u_i(\sigma_i, \alpha_{-i}) > u_i(\sigma_i', \alpha_{-i})$.

It means that P_i can never improve his utility by playing σ_i' and sometimes improve it by not playing σ_i'.[1]

As mentioned previously, the notion of Nash equilibrium is fundamental in game theory. Since no rational player will choose weakly dominated strategies, any Nash equilibrium involving such a strategy will not occur *in practice*. As a consequence, in our cryptographic setting, we can purge those strategies out.

Definition 4.4.2. [49] Let $< \mathcal{P}, H, P, (u_i) >$ be an extensive form game with perfect information. For any $\hat{\Omega} \subseteq \Omega = \Omega_1 \times \cdots \times \Omega_n$, let $DOM_i(\hat{\Omega})$ denote the set of strategies in $\hat{\Omega}_i$ that are weakly

[1] A strategy σ_i' is strictly dominated if player P_i can always improve his utility by not playing σ_i'.

dominated with respect to $\hat{\Omega}_{-i}(= \underset{j \neq i}{\times} \hat{\Omega}_j)$. Set:

$$\Omega_i^{\infty} := \bigcap_{l \geq 1} \Omega_i^l \qquad \text{where} \quad \Omega_i^l := \Omega_i^{l-1} \setminus DOM_i(\Omega^{l-1}), \quad \text{for} \quad l \geq 1.$$

A Nash equilibrium $\sigma = (\sigma_1, \ldots, \sigma_n)$ of $\mathcal{G}$ *survives iterated deletion of weakly dominated strategies* if $\sigma_i \in \Omega_i^{\infty}$ for all $i \in \{1, \ldots, n\}$.

As pointed out by Kol and Naor [38], in the setting of rational secret sharing, the solution concept *Nash equilibrium surviving iterated deletion of weakly dominated strategies* is not strong enough that some *bad* strategies still survive this deletion process. Now, we proceed to introduce a stronger one.

Definition 4.4.3. [49] A tuple $\sigma = (\sigma_1, \ldots, \sigma_n)$ is a *strict Nash equilibrium* if for all i and every strategy $\sigma_i' \in \Omega_i$ other than σ_i, we have: $u_i(\sigma_i', \sigma_{-i}) < u_i(\sigma_i, \sigma_{-i})$.

Remark 4.4.4. Intuitively, Definition 4.4.3 means that each P_i has an incentive not to deviate from σ_i as long as the remaining participants follow σ_{-i} (for all $i \in \{1, \ldots, n\}$), since any single deviation will cause a strict decrease in utility. It is obvious that the solution concept *strict Nash equilibrium* is strictly stronger than *Nash equilibrium* in the following sense: provided that the remaining players follow their strategies, deviations of a single player cannot increase his utility in a Nash equilibrium while any deviation of a single player will lead to a decrease in his utility in a strict Nash equilibrium. Furthermore, the notion of a strict Nash equilibrium is stronger than that of Nash equilibrium surviving iterated deletion of weakly dominated strategies, since in a strict Nash equilibrium, no strategy is weakly dominated.

However, like Nash equilibria, any strict Nash equilibrium can never prevent the deviation of a single player without the assumption that all players agree on which strict Nash equilibrium will be played and that each player holds the belief that all the other players indeed follow their prescribed strategies. In this sense, they are far away from being sufficiently strong. A much stronger solution concept is *dominant solvability.* Informally, a game is called *dominant solvable*, if after the process of iterated deletion of weakly dominant strategies, there is only one strategy left for each player P_i, say σ_i, $1 \leq i \leq n$. In this case, $(\sigma_1, \ldots, \sigma_n)$ is a very strong prediction for the way in which the game will be rationally played.

The strongest possible notion of equilibrium is dominant strategy equilibrium, which yields the best possible outcome to each player no matter what other players do. Formally,

Definition 4.4.5. [49] A strategy profile $\sigma = (\sigma_1, \ldots, \sigma_n)$ is a *dominant strategy* if for any $P_i \in \mathcal{P}$, σ_i is a best response for P_i no matter which strategies the other players may use, that is,

$$u_i(\sigma_i, \alpha_{-i}) \geq u_i(\alpha_i, \alpha_{-i}) \qquad \text{for any } \alpha \in \Omega.$$

Remark 4.4.6. Not all games admit dominant-strategy solutions. If a dominant strategy $\sigma = (\sigma_1, \ldots, \sigma_n)$ does exist, it will be best for each rational player P_i to choose σ_i. Besides, in choosing σ_i, each rational P_i neither needs to rely on his belief that each other player P_j, $(j \neq i)$ will choose σ_j nor needs to rely on the rationality of the other players. Thus, predicting that each P_i will play σ_i is indeed the strongest form of prediction of the game.

Traditional results in game theory mostly consider the equilibrium notions such as Nash equilibrium that only tolerate deviations of a single player. From the cryptographic perspective, these solution concepts are only partially meaningful as they are stated from the perspective of *individual* players and disregarding *collusion* altogether. However, in practice as well as in cryptographic applications, players can form coalitions. It can be the case that if several players form a coalition and they all deviate from their strategies then each of them could gain more. Now we will proceed to give the definitions of equilibria that tolerate deviations of coalitions containing multiple players.

Definition 4.4.7. [1] A strategy profile σ induces an *r-resilient Nash equilibrium* if for any coalition C of at most r players and for any strategy profile σ', it holds that:

$$u_i(\sigma'_C, \sigma_{-C}) \leq u_i(\sigma_C, \sigma_{-C}) \text{ for any } P_i \in C.$$

More generally, a strategy profile σ induces an *$\mathcal{A}$-resilient Nash equilibrium* if for any coalition $C \in \mathcal{A}$ and for any strategy profile σ', it holds that:

$$u_i(\sigma'_C, \sigma_{-C}) \leq u_i(\sigma_C, \sigma_{-C}) \text{ for any } P_i \in C,$$

where $\mathcal{A}$ is a monotone decreasing set (which means $B \in \mathcal{A}$ whenever $B \subseteq B'$ for some $B' \in \mathcal{A}$) whose elements are subsets of $\mathcal{P}$.

It is a general fact that many cryptographic protocols rely on the hardness of some problems such as discrete logarithm. However, these *hard* problems can be solved with negligible probability on the security parameter k. In order to use these cryptographic protocols as building blocks in the game-theoretic model, a slightly weaker variation of the Nash equilibrium, namely, the ϵ-Nash equilibrium, which is useful for implementing games using standard cryptographic computational assumptions, was proposed. Intuitively, this equilibrium guarantees that no player will gain more than ϵ by deviating from his prescribed strategy provided that all the remaining players stick to theirs. Here each player is regarded as a probabilistic polynomial-time (PPT) Turing machine and each utility function is considered as a polynomial of some security parameter k.

Definition 4.4.8. [27] A strategy profile σ induces an *r-resilient computational Nash equilibrium* if for any coalition C of at most r players and for any probabilistic polynomial-time strategy profile σ', it holds that:

$$u_i(k, \sigma'_C, \sigma_{-C}) \leq u_i(k, \sigma_C, \sigma_{-C}) + \epsilon(k) \qquad \text{for any } P_i \in C,$$

where ϵ is a negligible function. More generally, a strategy profile σ induces an *$\mathcal{A}$-resilient computational Nash equilibrium* if for any coalition $C \in \mathcal{A}$ and for any probabilistic polynomial-time strategy profile σ', it holds that:

$$u_i(k, \sigma'_C, \sigma_{-C}) \leq u_i(k, \sigma_C, \sigma_{-C}) + \epsilon(k) \qquad \text{for any } P_i \in C,$$

where $\mathcal{A}$ is a monotone decreasing set whose elements are subsets of $\mathcal{P}$.

Remark 4.4.9. The definition for *computational Nash equilibrium* can be derived directly from Definition 4.4.8 by letting $r = 1$. In a computational setting, we only consider probabilistic polynomial-time deviations.

5. HISTORY OF RATIONAL SECRET SHARING

Traditional cryptographic models assume that some parties are honest (i.e., they faithfully follow a given protocol) while others are malicious participants against whom the honest players must be protected. However, in many real-world applications, a participant may choose to be dishonest if deviating from the protocol will provide him with some advantage. Game theory can be used to model such a situation where players are *self-interested* (that is, *rational*). In other words, they do everything to maximize their utilities based on their beliefs about the actions of the other players. Under this new model, first it is assumed that each player prefers the outcomes in which he learns the secret. Hereinafter, learning a secret means outputting a correct secret. Note that this is the most basic assumption, since without it, no player has any interest to be involved in the protocol at all. Second, it is assumed that each player prefers the outcomes in which the number of players who learn the secret is minimum. This assumption sounds reasonable, especially when knowledge is power. We would like to emphasize that the utilities of the players depend *only* on who learns and who does not learn the secret. Suppose that t^* players are involved in the game for recovering the secret and $\sigma = (\sigma_1, \ldots, \sigma_{t^*})$ denotes a strategy profile. Let $o(\sigma) = (o_1(\sigma), \ldots, o_{t^*}(\sigma))$ be the outcome of the execution of σ, where $o_i(\sigma) = 1$ if and only if P_i learns the secret during the execution of σ. Let σ' be any strategy profile. Those assumptions on the preferences on outcomes mentioned above are described formally below:

- if $o(\sigma) = o(\sigma')$, then $u_i(\sigma) = u_i(\sigma')$,
- if $o_i(\sigma) = 1$ and $o_i(\sigma') = 0$, then $u_i(\sigma) > u_i(\sigma')$,
- if $o_i(\sigma) = o_i(\sigma')$ and $w(o(\sigma)) < w(o(\sigma'))$, then $u_i(\sigma) > u_i(\sigma')$,

where u_i denotes P_i's utility function and $w(x)$ denotes the Hamming weight of the binary vector x.

Under these assumptions, classical secret sharing schemes fail completely in the game-theoretic setting due to players' conflicting self interests, that is, it is rational not to broadcast one's share during the secret reconstruction phase. To understand this, we might take Shamir's t-out-of-n secret sharing scheme as an example. Suppose that t players $P_{i_1}, \ldots, P_{i_t}$ are involved in the reconstruction phase. For each $P_{i_j}, 1 \leq j \leq t$, there are two cases. Case 1: all the others broadcast their shares. Then, if P_{i_j} keeps silent, he will be the only one that learns the secret; otherwise, he will learn the secret with the others. Case 2: some of the others do not broadcast their shares. Then no matter what P_{i_j} does, no one will learn the secret. This suggests that regardless of the actions of the other players, a player will hardly be worse off at all (and perhaps, may even gain some benefit at times) by remaining silent, that is, remaining silent is a weak dominant strategy which will definitely be chosen by a rational player.

This problem motivated the study of rational secret sharing, which was first proposed by Halpern and Teague [31] in 2004. Roughly speaking, under those assumptions on preferences on outcomes mentioned previously, the main goal of a rational secret sharing is to design a recommended strategy for each participant along with the dealer with the requirement that on one hand, every participant has a motivation to follow his recommended strategy and on the other hand, if each participant sticks to his recommended strategy, the secret will be revealed to all. However, due to players' conflicting self interests, this requirement is so stringent that it is pretty hard to be satisfied. Thus, in the literature of rational secret sharing, the main goal is to design protocols for participants (including the dealer) such that on one hand, every participant in some coalition has motivation not to deviate from his strategy provided that all the non-coalition participants stick to theirs and on the other hand, if each participant sticks to his strategy, the secret will be revealed to all. To retell it in game-theoretic terminology, designing rational secret sharing protocols essentially amounts to designing a recommended strategy for each player along with the dealer. Each player's strategy set is defined to be all the possible deviations from his recommended strategy. The requirement is that the tuple of recommended strategies induces a Nash equilibrium or one of its variants, in which each player learns the secret. The stronger the equilibrium, the better the protocol.

In this chapter, we present a brief survey on rational secret sharing. Specifically, we summarize the main ideas behind rational secret sharing schemes and the common approach used to design them. Further, we provide comparisons among these protocols based on their channel models, utility independence, equilibrium types and efficiencies.

5.1 The Basic Idea behind Previous Work

Roughly speaking, designing a rational secret sharing scheme essentially amounts to designing a recommended strategy for each player: regard the secret reconstruction phase as a game among all active players, where each player's strategy set contains all possible deviations from his recommended strategy. The objective is that when the game is *rationally played*, the property that *all the players will learn the secret* holds. Here by *rational*, we mean that all players try to maximize their utilities. The desired protocols for rational secret sharing always consists of several rounds, among which only the last round is the real round, since the real shares are broadcasted only in the last round. If some player keeps silent at some round except the last round, the protocol will terminate prematurely (as a punishment) and consequently, each player loses the chance to discover the secret forever, which is not what he desires. Thus, the main obstacle in designing such a protocol is players' desire to keep silent in the last round, if they can identify it, since they no longer fear future punishments. Therefore, the key idea behind all the existing protocols is that in any given round, the players (even if they form an unauthorized coalition) cannot distinguish in advance whether the current round is going to be the last round, or whether it is just a test round designed to catch cheaters. More explicitly, the basic idea behind most of the previous schemes is as follows.

- Regard the scheme as a game (more precisely, regard the secret reconstruction process as a game), which proceeds with several iterations; Punish players for detectable deviations, that is, the game stops once some deviations are detected. In each iteration:
- The dealer $\mathcal{D}$ distributes shares for either
  - the *valid* (*real*) secret with some probability α
  - or an *invalid* (*fake*) secret.
- Players broadcast their shares.
  - If a player's deviation is detected, the game stops;
  - Otherwise, if an invalid secret is reconstructed, all the players proceed to the next iteration.
  - The game stops once the real secret is recovered.

Here each player can identify whether the recovered value is the real secret or not.

- To maximize his utility by cheating, a player has to guess the *real* iteration; thus provided that α is properly set, it is *rational* to stick to the protocol.

- Online dealer is not always necessary. In fact, he can be removed by using secure multiparty computation or by other techniques.

5.2 Previous Work on Rational Secret Sharing

Halpern and Teague [31] were the first to introduce the idea of rational secret sharing and design a protocol for it under the previous assumptions on players preference on outcomes. Their work, as well as the subsequent work of Gordon and Katz [29] and Abraham, Dolev, Gonen and Halpern [1], used simultaneous broadcast and pairwise secure channels. One of Halpern and Teague's main contributions [31] is to identify rational secret sharing as a functionality, which provides much enlightenment into combining the cryptographic and the game-theoretic realms. Besides, they contributed two technical results concerning rational secret sharing.

First, they proved that there does not exist any protocol for rational secret sharing that has a fixed number of rounds and uses simultaneous broadcast communication channels. More precisely, they demonstrated that there does not exist a rational secret sharing scheme that satisfies the following properties:

- There is a commonly known upper-bound on the number of rounds in the protocol.

- The tuple of prescribed strategies induces a variant of Nash equilibrium that is at least as strong as Nash equilibrium surviving iterated deletion of weakly dominated strategies.

- The channels used by the players during the secret reconstruction phase are not stronger than simultaneous broadcast channels and pairwise secure channels.

The basic idea behind the proof of this impossibility result is described as follows:

- We stress that all the arguments are based on those assumptions on players' preferences on outcomes. They claimed that broadcasting valid information during the last round of the protocol is weakly dominated by keeping silent. Here is the argument to justify this claim:

  - Suppose t is the threshold.

  - We consider player P_i's situation: either $t-1$ other players broadcast their share or not.
  - If they do, then player P_i will recover the secret; otherwise, he can not.
  - Whether or not he broadcasts his share or not does not affect the others' actions (since all the broadcasts are assumed to happen simultaneously).
  - Furthermore, if only $t-1$ other players broadcast their shares, then broadcasting his share will enable others to gain the secret. And if he does not send his share, then he will be the only one that gains the secret. In all, broadcasting his share will not increase his utility.

- Because all strategies in which players broadcast valid information in the last round are weakly dominated, every player knows that each rational player will keep silent in the last round. Therefore, the second last round essentially becomes the last round.

- In this way, every round of iterated deletion of weakly dominated strategies removes those strategies in which information is broadcast during the last round, and as a result, effectively removes the last round itself. After many rounds of iterated deletion of weakly dominated strategies, the only remaining strategy for any player will be to keep silent in any round, which obviously leads to the outcome in which no player reconstructs the secret.

Second, they proposed a randomized t-out-of-n rational secret sharing scheme for any $n \geq t \geq 2$, which relies on an on-line dealer. The sketch of their protocol is summarized below. They described a 3-out-of-3 rational secret sharing protocol, based on which they created a t-out-of-n protocol with $t \geq 3$ by dividing the players into 3 groups and choosing one player from each group as a group leader. All players in each group use pairwise secure channels to send their shares to the group leader. The three group leaders then execute the reconstruction phase of the previous 3-out-of-3 rational secret sharing protocol. Finally, the group leaders broadcast their secret obtained during the previous step and consequently, all players learn the secret. The 3-out-of-3 rational secret sharing scheme proceeds in several rounds and at the beginning of each round, the three players receive three valid secret shares signed by the dealer. (Note that this is different from most of the existing protocols in which the players carry valid shares only during the last round.) During the process of secret reconstruction, a player opens his shares to others only if some probabilistic event happens. Thus, with some probability, the shares opened are sufficient to recover the secret.

If the secret is not recovered in a given round due to insufficient opened shares, and no deviation has been detected so far, the dealer must distribute new secret shares and a new round begins. Whenever any derivation is detected, the game ends. Their protocol induces a Nash equilibrium surviving iterated elimination of weakly dominated strategies. However, their protocol has some limitations that are listed below.

First, their protocol relies on an on-line dealer, which is not always available in practice. Second, in order to allow each player to verify other players' shares during the reconstruction phase, digital signatures are used. However, the usage of digital signatures makes their protocol vulnerable to backward induction, since as pointed out in [37], unlike standard cryptographic protocols, this protocol may run for an exponential number of rounds, at least with some negligible probability. If it happens to be this case, the key problem is that there is a critical integer b such that after b rounds, any player can break the cryptographic primitives used and as a result, either reveals the other players' shares encoded by them or sends some fake shares without being detected. Therefore, the b^{th} round is essentially the last, and the players have no incentive to cooperate once it has reached, since they no longer fear any future punishment. Consequently, the $(b-1)^{th}$ round is now essentially the last round and the players deviate for the same reason. Finally, the protocol induces a Nash equilibrium surviving iterated elimination of weakly dominated strategies. However, it is not resilient to any coalition consisting of multiple players. To justify this claim, recall that in their t-out-of-n rational secret sharing scheme, the players are divided into three groups, and each player sends his share to his group leader. Consider what would happen if any two of the three group leaders were in coalition. Clearly, once they have t shares, they could simply execute the protocol entirely and reconstruct the secret without giving it to any of the other players.

In order to remove the first two limitations, we propose a t-out-of-n rational secret sharing scheme which requires the involvement of the dealer only during the initial share distribution phase. Our construction is information theoretically secure and it is immune against backward induction. Our protocol leads to a Nash equilibrium surviving the iterated deletion of weakly dominated strategies for $t \geq 3$. The reader can refer to Chapter 6 for further details.

Following the work of Halpern and Teague [31], Gordon and Katz [29] demonstrated that a 2-out-of-2 rational secret sharing protocol is possible despite Halpern and Teague's apparent proof to the contrary. Indeed, they described a significantly simplified t-out-of-n rational protocol in simultaneous channels model that has stronger properties than that of Halpern and Teague's.

Under the assumption that the secret s lies in a commonly known subset G of some finite field $\mathbb{F}$, in each round, the on-line dealer distributes shares of $s \in G$ with probability β using Shamir's secret sharing scheme; and with probability $1-\beta$, he distributes shares of some $s' \in \mathbb{F} \setminus G$. Each share is signed by a digital signature, which is assumed to be unforgeable. In each round, every player broadcasts the share he receives during the current round. If at least t correct shares are broadcasted, each player reconstructs a value. If this value lies in G, everyone realizes that this revealed value is the secret and the protocol ends; otherwise, the protocol repeats from scratch. Note that as usual, the game stops whenever any deviation is detected. The equilibrium it provides is a Nash equilibrium surviving iterated elimination of weakly dominated strategies. Furthermore, it is resilient to coalitions of size at most $t-1$, although they did not mention this in their paper. In addition, Gordon and Katz noticed that dealer's periodic involvement during the reconstruction phase is illogical. To solve this problem, they suggested replacing him with a secure function evaluation (SFE) protocol. This SFE protocol takes the true shares as inputs and, with probability β, re-shares the secret and with probability $1-\beta$ shares some s' not in G. It also signs the new shares appropriately. However, because the function being computed by these protocols is complicated, it is unclear whether it is computationally efficient.

Later, Abraham, Dolev, Gonen and Halpern published their paper [1], in which they first considered tolerance of coalitions and proposed the notion of r-resilient Nash equilibrium. Besides, they introduced the notion of ϵ-Nash equilibrium and proposed three protocols for t-out-of-n rational secret sharing with a trusted mediator, who distributes at the beginning of each round the shares for 0 with probability $1-\alpha$ and for the real secret with probability α. Here, it is assumed that the secret $s \neq 0$ and this is known to all participants. Each protocol induces a $(t-1)$-resilient Nash equilibrium surviving iterated elimination of weakly dominated strategies. Finally, they described in detail how to remove the online mediator by using computationally secure function evaluations. The first protocol is designed for arbitrary t and n with $t \leq n$. Just as the protocols mentioned previously, this protocol relies on knowledge on players' utilities (or at least some bounds on them). Since precise knowledge about the players' utilities may not be available or too expensive to collect, the resulting protocol will inevitably be better if less knowledge is required. The second protocol uses an information checking protocol to allow each player to check the correctness of the shares sent by other players, and so each player can identify fake shares with high probability (as long as the size of the underlying field is big enough). It also relies on the assumption that the mediator knows the information about players' utilities and works only when $n \geq 2t - 1$ players are active in the secret reconstruction phase. However, the advantage it has over the first one is its constant round complexity. The third protocol makes use of the error-correcting ability of generalized Reed-Solomon code to correct all possible fake shares during the secret reconstruction process. It works only when $n \geq 3t - 1$ players are involved in the secret reconstruction phase. The good news is that it has constant round complexity and there is no need for the mediator to know anything about the utilities. Here we only describe the sketch of the first protocol as follows, from which one can get the basic idea behind their constructions.

1. The dealer distributes digitally signed shares of s using Shamir's secret sharing scheme.

2. In the 0^{th} round, each player sends his share of the secret to the mediator.

3. If the mediator does not receive all the appropriate messages from all players, it stops playing. Otherwise,

4. In the r^{th} $(r > 0)$ round, with probability α the mediator re-shares the secret. Otherwise, it distributes shares for 0. Then all the players are required to broadcast their shares. Here α depends on the players' utilities.

5. If at least t correct shares were broadcast, each player can reconstruct a value. If it's nonzero, then everyone gains the true secret and the protocol stops.

6. If all shares were correctly broadcast but the reconstructed secret is 0, then each player sends an acknowledgement to the mediator and they switch to step 4. Otherwise the protocol stops.

This protocol is resilient to coalitions of size at most $t - 1$ and runs in time linear in $\frac{1}{\alpha}$.

At this point, it is worthwhile to summarize that all the protocols we have described so far share the following limitations:

- They either need an on-line dealer (or mediator) or replace the on-line dealer by secure multi-party computations which may make the protocols quite inefficient.

- They assume simultaneous broadcast channels which are expensive to be implemented in practice.

- Almost all of them are vulnerable to backward induction due to the usage of computational based cryptographic primitives such as digital signatures.

- The equilibrium provided is a Nash equilibrium surviving iterated deletion of weakly dominated strategies, which is not sufficiently strong since it may not be the only Nash equilibrium in the game. Therefore, there is no guarantee that rational players will not deviate and it is likely that the game ends in a different equilibrium, in which not all the active players learn about the secret, or does not end in an equilibrium at all.

Kol and Naor [37] also noticed the weakness of the previous iterated admissibility (surviving iterated elimination of weakly dominated strategies) and they argued that this criterion used to evaluate such protocols is problematic, since it is possible that some equilibrium surviving iterated elimination of weakly dominated strategies is still a bad one. As a remedy, they suggested a stronger notion: strict Nash equilibrium. On the other hand, although it seems that susceptibility to backward induction is an inherent property of every computational based rational protocols, they demonstrated that it is not the case by proposing a protocol that is immune to backward induction. Roughly speaking, the main idea behind it is to ensure that no iteration except the last one contains any information about the players' private values (in the information-theoretic sense). More explicitly, they constructed a new cryptographic tool called *meaningful/meaningless* encryption with the following property: some public keys yield ciphertexts that cannot be decrypted (even by infinitely powerful players) and in this sense they are called *meaningless* keys, while the other keys are called *meaningful* keys which provide semantic security. One can distinguish *meaningful* keys from *meaningless* ones easily only when she (or he) obtains the private keys. However, in this scheme, the shares of the parties have unbounded lengths. More explicitly, the expected length of the parties' shares in their 2-out-of-2 scheme is $O(\beta^{-1}(|s|+k))$, where $|s|$ denotes the bitsize of s, k is a security parameter, and β, which can be very small, is determined by players' utilities.

In another work of Kol and Naor [38], they pointed out that previous schemes (mostly cryptographic) either relied on computational assumptions, which makes them susceptible to backward induction, or used stronger communication channels (such as envelope and ballot box [34]). As a solution, they produced a scheme with unbounded shares which is non-cryptographic, immune to backward induction, relies on simultaneous broadcast channels and induces a strict Nash equilibrium. Furthermore, they demonstrated that their protocol can also be used to construct an ϵ-Nash equilibrium secret sharing scheme on the non-simultaneous broadcast channel model. The innovation of their protocol is that the share assigned to each player is actually a list of possible secrets. We consider the scenario of two players as an example. One party receives a list of elements of length l (the *short party*), and the other part a list of length $l + d$ (correspondingly, the *long party*). The short list is a strict prefix of the long one. The integer l and d are chosen by the dealer according to the geometric distribution with parameter β, which depends on the utility functions of the players. The real secret is located at the $(l + 1)^{th}$ position in the long list, while all the other elements in the two lists are randomly chosen. Thus, the $(l+1)^{th}$ round is the real round. In addition to the two lists, the dealer selects an independent random permutation for every round to determine the order in which the parties send their list elements in this round. The long list along with the those permutations are given to the party who sends his message first in the real round, while the short list along with those permutations are given to the other party. The key point here is that neither party knows whether he is the short or long party. The parties proceed round by round to reconstruct the secret: in the i^{th} round, each party is required to send the i^{th} element in his list in the order determined by the permutation corresponding to this round. At the $(l + 1)^{th}$ round, the *long party* is the first to send his share. Since the list of the *short party* is finished, there is no more element to send, so he keeps silent in this round. Thus, at the end of this round, both parties realize that the message sent in this round is the real secret. Note that the expected round complexity for t-out-of-n case is $O(\beta^{-1}t)$, where β can be very small.

In order to make an attempt to reduce the share length as well as the expected round complexity, Fuchsbauer, Katz and Naccache [27] proposed a simple and efficient methodology for rational secret sharing leading to various instantiations without the dealer's involvement during the reconstruction process. Their protocols do not require physical assumptions such as envelope channels or simultaneous channels and can even be run over asynchronous, point-to-point networks. In their protocol (we only describe the 2-out-of-2 case), during the reconstruction process, the dealer chooses an integer i^* as the round number according to an appropriate distribution with parameter β. Then he generates the key pairs (vk_1, sk_1), (vk_2, sk_2) and (vk'_1, sk'_1), (vk'_2, sk'_2) for two verifiable random functions (VRF) Eval and Eval', where vk represents a verification key and sk represents a secret key. Denote by $\text{Eval}_{sk}(x)$ the evaluation of Eval on the input x using the secret key sk. The dealer gives the verification keys to both parties and gives sk_j to player P_j,

$j = 1, 2$. He also gives $\text{share}_1 = s \bigoplus \text{Eval}_{sk_2}(i^*)$ along with $\text{signal}_1 = \text{Eval'}_{sk'_2}(i^* + 1)$ to player P_1, and correspondingly, $\text{share}_2 = s \bigoplus \text{Eval}_{sk_1}(i^*)$ along with $\text{signal}_2 = \text{Eval'}_{sk'_1}(i^* + 1)$ to player P_2. Each round consists of one message from each player: in the i^{th} round, player P_j, $j = 1, 2$, sends $\text{Eval}_{sk_j}(i)$ and $\text{Eval'}_{sk'_j}(i)$ to the other player. In this way, in a fake round nothing about the secret is revealed (in a computational sense). Besides, neither party can identify the real round in advance. when $\text{signal}_1 = \text{Eval'}_{sk'_2}(i + 1)$, player P_1 knows that the real round is the i^{th} round and outputs the secret $s = \text{share}_1 \bigoplus \text{Eval}_{sk_2}(i)$. It is similar for player P_2. This protocol induces a *computational strict Nash equilibrium that is stable with respect to trembles.* (Intuitively, stability with respect to trembles models players' uncertainty about other players' behavior, and guarantees that even if a player believes that other players might play some arbitrary strategy with a small probability, there is still no better strategy for him than to follow the protocol. Please refer to Section 7.3.) Notice that in this protocol, as in the previous protocols, each player's deviation bear the risk of causing the protocol to terminate prematurely, unless he is sure that he deviates during the real round. However, the difference now is that once a party realizes that the real round has occurred, the real round is over, and all parties can reconstruct the secret. In this way, the need for simultaneous channels is removed at the cost of adding only a single round. Although this protocol does not share most of the drawbacks mentioned above, for the t-out-of-n case, the share bit size is $2n|s| + O(k)$ for security against a single deviation, and raises to $(n - t + 1) \cdot (2n|s| + O(k))$ to achieve $(t-1)$-resilience, where k is a security parameter. The latter share length leads to practical efficiency issues when $n - t + 1$ is large or when their technique is used as a building block within more general rational multiparty computation protocols. Motivated by this essential drawback, we propose a protocol with share length $O(k)$ based on the Chinese Remainder Theorem. One can refer to Chapter 7 for more details.

Another line of work was pursued by Izmalkov, Lepinski and Micali [34]. Roughly speaking, they proposed fair, rational secure multiparty computation protocols which are resilient to coalitions. However, the hardware requirements needed for these operations, including envelope channels and ballot boxes, are very expensive to be executed in practice. Moreover, we have no idea about whether they can or how they can be implemented for distant participants. Now we would like to mention an interesting protocol presented recently by Ong, Parkes, Rosen and Vadhan [48]. It works in quite a different model. On one side, it does not require any special channels (it relies on ordinary broadcast channels rather than simultaneous broadcast channels). On the other side, it relies on the honesty of a few players.

At this point, we stress that almost all the protocols we have described so far share the common limitation: the designer should know the actual utility values of the players or at least some bounds on them. This assumption is problematic, since the utilities of the players are not necessarily public knowledge. The basic question regarding utility independence was proposed in [31]. Briefly, by utility independence, we mean that the protocol works for all possible values of utilities satisfying the aforementioned assumptions on players' preferences on outcomes. The first partial answer to this question was given by [1], where it is showed that with an online mediator, utility independence is possible for t-out-of-n secret sharing provided that $t < n/3$ and that all n players are involved in the reconstruction protocol. Asharov and Lindell [2] first proved that in the case of two parties, there does not exist a utility independent rational secret sharing scheme in either simultaneous or non-simultaneous channels model. On the positive side, they constructed a utility independent t-out-of-n rational protocol with an on-line dealer. Their protocol has an expected round complexity $O(1)$ and induces a $(\lceil \frac{t}{2} \rceil - 1)$-resilient Nash equilibrium surviving iterated deletions of weakly dominated strategies. Furthermore, they showed that it is optimal in the sense that there does not exist a rational secret sharing protocol that is $\lceil \frac{t}{2} \rceil$-resilient while preserving utility independence.

Finally, we would like to introduce the paper written by Micali and Shelat [43]. In this work, they pointed out that the quality of a rational protocol depends crucially on the solution concept it induced: the stronger the solution concept adopted, the better the underlying protocol. Thus, it is amazingly significant if the game has dominant strategy solution, since in this case, each rational player will definitely choose his prescribed strategy, no matter what the others do and consequently, the secret will be revealed to each player. However, it has only very limited significance if the game induces a Nash equilibrium or one of its variants. Besides, even if each player is guaranteed to learn the secret at each of the possible Nash equilibria of the game, such limited meaningfulness persists as long as the underlying game has multiple Nash equilibria, since it is likely that the game will not end in any equilibrium at all due to *mismatched beliefs on players' prediction* on which Nash equilibrium will be played. Consequently, it can not guarantee that all rational players learn the secret when the game stops. This argument suggests that all the protocols described so far have only very limited significance. The second quality measure for a rational protocol is the amount of knowledge about the players' utilities required in the design of the protocol, since this knowledge may not be available or too expensive to collect. Roughly speaking, a better protocol is attained if the knowledge required is minimal. However, these two measures are not all that one needs to consider when designing a game. Indeed, another quality measure is the strength of the communication models needed during the process of the game. Clearly, a protocol which requires a stronger communication channel is inferior to the one that works with a weaker channel. Besides, the security requirement, that is, whether the protocol achieves information-theoretic security or computational security, is another important quality measure. The final and crucial point concerning the quality of a rational secret sharing protocol is its efficiency, including expected round complexity, computational complexity and space complexity. Here by the space complexity, we mean the amount of space needed to store public information and the amount of space needed to store secret information. In most cases, we care more about the latter. In particular, the smaller the share bitsize, the better the scheme. According to the first two criteria, all the existing protocols fall short. Hence, there is a pretty long way to go in the development of rational secret sharing. One possible and meaningful direction is to explore stronger solution concepts (than Nash equilibrium or its variants) and design protocols achieving them.

6. AN UNCONDITIONALLY SECURE RATIONAL SECRET SHARING SCHEME BASED ON SYMMETRIC BIVARIATE POLYNOMIALS

In this chapter, we propose an unconditional secure protocol for rational threshold secret sharing, which neither assumes an online dealer or any trusted parties (a mediator for example), nor relies on complicated secure multiparty computation to redistribute the shares of the secret in each round. (Only the operation XOR is needed for the MPC used in our protocol). Instead, we borrow the idea from the proactive secret sharing scheme [20] to renew the shares merely by interactions between players. Unlike constructions quoted in the previous chapter, the secret s is masked by a one-time pad. This provides information theoretical security and makes our construction immune to backward induction mentioned previously. Our scheme is based on symmetric bivariate polynomials. Even if this technique has already been applied before for multiparty computation protocols, to the best of our knowledge, it is the first time that it has been used in rational cryptography. Our protocol is efficient in terms of round execution (as the dealer will not be needed), share size and computation, and it guarantees that all players learn the secret at a Nash equilibrium surviving the iterated elimination of weakly dominated strategies. As in most of the prior work, we need a simultaneous broadcast channel and pairwise privacy channels. We need to point out that what we demonstrate in this chapter is a full version of the publication [58], which is a joint work with C. Tartary and H. Wang.

6.1 Our Protocol for t-out-of-n Rational Secret Sharing Secure against a Single Player's Deviation

In order for the reader to get an easier understanding of our protocol, we first give a general view of our secret reconstruction phase. The full description of our scheme is in Sect. 6.1.2.

6.1.1 Overview of the Reconstruction Phase

Our scheme relies on the masking of the secret s by one-time pad. Such an approach already appeared in [27]. However, Fuchsbauer *et al.* used a verifiable random function (VRF) which is a cryptographic primitive and the existence of which is based on some computational assumption.

In order to provide information theoretical security, the dealer will first use a one-time pad r over the secret s. He will also mask r in a similar way with another random element r' and publish $r + r'$ to a register accessible to all players. In order to recover s, the players will need to obtain both r and r'. That is why the second task of the dealer is to distribute r and r' amongst the n players using two independent instances of Shamir's scheme [56] with threshold t. Note that the public value $r + r'$ will be used by the participants to check the consistency of the two reconstructions. The third task of the dealer consists of sharing $s + r$ using a bivariate polynomial with degree $t - 2$ in each of its two unknowns.

Assume that $t^*(\geq t)$ players want to participate in the secret reconstruction process. We first consider the case when t is even. A similar construction holds when t is odd (see Sect. 6.4). Note that this differentiation "t is even/odd" has no influence on the dealer's job when initially sharing s.

The reconstruction phase proceeds in three stages. During the first two stages, the goal of the t^* players is to recover the pad r (using r' and $r+r'$). The third stage is a sequence of *invalid* and *valid* iterations which is a frequently used technique for rational secret sharing schemes. During each of these iterations, the broadcast shares correspond to $s + r$. Those iterations have the following properties:

- each player opens his share to others only when some probabilistic event happens.
- *invalid* iteration: no information about s is revealed since the number of broadcast shares related to $s + r$ is less than the threshold value $t - 1$. At the end of such an iteration, shares are renewed.
- *valid* iteration: every player recovers s under the assumption that every player follows the protocol.

The key in this process is the fact that nobody knows in advance whether the next iteration will be *valid.*

During any iteration of the third stage, each of the t^* participating players P_{i_j} chooses a bit b_{i_j} such that $b_{i_j} = 1$ with probability α depending on the utilities of the n players. Then, all t^* players commonly run a simple multiparty computation protocol to compute the parity value $p := b_{i_1} \oplus b_{i_2} \oplus \cdots \oplus b_{i_{t^*}}$. Our multiparty computation protocol is an extension of what was done in [31] in the case of three players.

If $p = 0$ then the t^* players are asked to repeat the previous iteration. Otherwise, each P_{i_j} broadcasts his share to the $t^* - 1$ other members if $b_{i_j} = 1$.

When the protocol did not abort before this point, we have two possibilities:

1. $\underline{P_{i_j}\text{ has at most } t-2 \text{ shares (for some } j)}$: the players run a check phase to catch potential cheaters. If the shares are correct, then the t^* players renew their shares of s using a technique from proactive secret sharing scheme [20] and they start over by choosing a new random bit.

2. $\underline{\text{All players have at least } t-1 \text{ shares}}$: the set of t^* players attempt to reconstruct $s + r$ using polynomial interpolation or error-correcting techniques (refer to Sect. 6.1.2 for details). Once they obtain $s + r$, they can deduce s since they got r at the end of the second stage.

6.1.2 Our Construction

Our computations will be done in the finite field $\mathbb{F} = \mathrm{GF}(q)$ for which ω is a primitive element. As before, we denote $\mathcal{P} := \{P_1, \ldots, P_n\}$ the set of participants and the secret value to be distributed is $s \in \mathbb{F}$. As said in the previous section, we consider the case when the threshold value t is even.

During the secret reconstruction phase, we assume the existence of a simultaneous broadcast channel for all participating players and the presence of secure channels between any pair of these players. All these channels are authenticated. The protocol stops once some player aborts.

For each $i \in \{1, \ldots, n\}$, denote u_i (respectively, u_i^+) the minimal (respectively, maximal) payoff of P_i when he retrieves the secret and denote u_i^- his maximal payoff when P_i does not recover s. As usually assumed in the rational cryptographic context, we consider: $u_i^+ > u_i > u_i^-$ for all $i \in \{1, \ldots, n\}$.

Initial Share Phase

This is the only phase where the dealer is active. His goal is to distribute s over $\mathcal{P}$.

1. The dealer chooses two independent random values r and r' uniformly distributed over $\mathrm{GF}(q)$, and publishes the value $r + r'$ in a public register.

2. The dealer shares r into $(r_1, \ldots, r_n)$ and r' into $(r'_1, \ldots, r'_n)$ using two independent instances of Shamir's t-out-of-n secret sharing scheme. He distributes through a secure channel the pair (r_i, r'_i) to P_i for all $i \in \{1, \ldots, n\}$.

3. Denote $v := s + r$. The dealer constructs a symmetric bivariate polynomial $f(x, y) = \sum_{i=0}^{t-2} \sum_{j=0}^{t-2} a_{ij}\, x^i\, y^j$ where $a_{00} = v$ and $a_{ij} = a_{ji}$ for all $0 \leq i, j \leq t-1$. For each i, the dealer sends the univariate polynomial $h_i(x) := f(x, \omega^i)$ to P_i through a secure channel.

Remark 6.1.1. Due to the symmetry of f, we have $h_j(\omega^i) = h_i(\omega^j)$ for any pair (i, j). This property is fundamental in our work.

Remark 6.1.2. During the initial share phase, the dealer uses a symmetric polynomial f to distribute the shares of v. We would like to emphasize why the degree of f in each of these two variable is $t - 2$ rather than $t - 1$. Since those shares are not authenticated by any cryptographic primitive, in order to verify the consistency of the data sent by a given P_i we require all remaining $t^* - 1 (\geq t - 1)$ participants to run the following check phase .

Check Phase

The goal of this phase is to check the consistency of the share λ broadcasted by P_i. This task is done by the players participating in the secret reconstruction process – except P_i.

1. Each participating player P_j $(j \neq i)$ broadcasts his check value $h_j(\omega^i)$.

2. Each of these players computes, using polynomial interpolation, a polynomial and checks whether it's constant term equals λ.

Share Renewal Phase

As the check phase, share renewal is done by the players participating in the secret reconstruction process. We assume that there are $t^*(\geq t)$ such players. For ease of description, we can assume without loss of generality that those players are $P_1, \ldots, P_{t^*}$. In this phase, each participating P_i plays a similar role to the dealer's (initial share phase) to renew his share for v.

1. Each P_i selects a random symmetric polynomial $\delta_i(x, y)$ of degree at most $t-3$ (with respect to either variable). He sends $\delta_{i,j}(x) := \delta_i(x, \omega^j)$ to P_j over a secure channel (for all $j \in \{1, \ldots, t^*\} \setminus \{i\}$).
2. After receiving $\delta_{i,j}(x)$, P_j computes and sends the value $\delta_{i,j}(\omega^l)$ to P_l, for $l \in \{1, \ldots, t^*\} \setminus \{j\}$, over a privacy channel.
3. P_l checks whether $\delta_{i,j}(\omega^l) = \delta_{i,l}(\omega^j)$, for $j = 1, \ldots, t^*$.
4. If one of these equalities is not satisfied for some pair (i, j), then P_l aborts the whole protocol. Otherwise, each P_l updates his share as: $h_l(x) \leftarrow h_l(x) + (x + \omega^l) \sum_{i=1}^{t^*} \delta_{i,l}(x)$.

Remark 6.1.3. After the renewal phase, we have the following relation for the new shares: $h_j(\omega^l) = h_l(\omega^j)$ for any $1 \leq j, l \leq t^*$.

We assume that $t^*(\geq t)$ players participate in the secret reconstruction. As before, we can assume that they are $P_1, \ldots, P_{t^*}$. Our reconstruction protocol contains three stages for each of these t^* players. The first stage is dedicated to the recovery of the random value r used by the dealer as a pad over the secret s. The complete reconstruction protocol is described in the next two pages.

Remark 6.1.4. When the t^* players reach Stage 3, we have: $\forall i \in \{1, \ldots, t^*\}, \quad p_i = 1$. Note that $t-1$ (odd number) is the minimum number of shares that a participant needs to uniquely determine a polynomial of degree $t-2$. Requiring that all the p_i's be equal to 1 means that each participant is holding an odd number of shares. This requirement is essential to guarantee that players who broadcast their shares can recover the secret whenever players who keep silent can recover it, where each of the latter obtains one more share.

Secret Reconstruction Phase

Stage 1

1. Each P_i broadcasts his pair (r_i, r'_i). If some player P_j obtains less than t^* pairs (including his own), then P_j aborts the whole protocol.

2. Each P_i constructs two sets of shares. The first one $S_{i,r}$ consists of all the first components of those t^* pairs (namely, all the r_i) and the second set $S_{i,r'}$ contains all the second components of the pairs (namely, all the r'_i). Player P_i checks if each of these sets can be interpolated by a polynomial of degree at most $t-1$. If this checking process is unsuccessful for some P_j, then P_j stops the protocol.

3. For each $i \in \{1, \ldots, n\}$, we denote $\mathcal{R}_i$ (respectively $\mathcal{R}'_i$) the constant term of the polynomial reconstructed by P_i corresponding to the set $S_{i,r}$ (respectively $S_{i,r'}$). Each P_i checks whether the sum $\mathcal{R}_i + \mathcal{R}'_i$ is equal to the public value $r + r'$.

   - If the verification is unsuccessful for some P_j, then he aborts the protocol.
   - Otherwise, all participants proceed to Stage 2.

The remaining two stages are used to recover v. Note that the threshold now is $t-1$ rather than t since the symmetric bivariate polynomial f has degree $t-2$ in each of its variables.

Stage 2

1. Each P_i chooses a bit b_i with $\Pr(b_i = 1) = \alpha$ as well as a uniformly distributed random bit b'_i.
2. Denote $d_i := b_i \oplus b'_i$. Let i^+ denote $i+1$ except that $(t^*)^+$ is 1. Similarly i^- denotes $i-1$ except that 1^- is t^*. Each P_i sends b'_i to player P_{i^+} and d_i to player P_{i^-} using secure channels. If some P_j does not send data to both neighbors P_{j^-} and P_{j^+}, then the protocol aborts.

3. Each P_i computes and broadcasts $b'_{i^-} \oplus d_{i^+}$. If some P_j does not receive the bits as prescribed, then the protocol aborts. Otherwise, denote $\Delta_{1,i}, \ldots, \Delta_{t^*,i}$ the t^* elements collected by P_i during this broadcast including his own. Each P_i computes $p_i = \bigoplus_{j=1}^{t^*} \Delta_{j,i}$.
4. If $p_j = 0$ for some P_j, then the whole protocol goes back to the first step of Stage 2. Otherwise, all participants proceed to Stage 3.

Secret Reconstruction Phase, Cont.

Stage 3

1. Each P_i broadcasts his share $h_i(0)$ if $b_i = 1$. Denote l_i the number of shares that P_i received during the previous broadcast (including his own if $b_i = 1$).

2. If $l_i < t - 1$ for at least $t^* - 1$ players P_i, then:

   (a) If l_i is even for at least $t^* - 1$ players P_i, then the protocol stops.

   (b) Otherwise, all t^* active players participate in the check phase. If some P_j does not broadcast $h_j(\omega^i)$ as required or some check fails then the protocol aborts. Otherwise, all players go to the renewal phase and then they proceed to the beginning of Stage 2.

3. If $l_i \in \{t-1, t\}$ for at least $t^* - 1$ players P_i, then each of these players P_i interpolates the shares into a polynomial $f_i(0, y)$. If the degree of $f_i(0, y)$ is $t - 1$ then P_i aborts the protocol. Otherwise, he outputs $f_i(0, y) + \mathcal{R}_i$, where $\mathcal{R}_i$ was computed at the third step of Stage 1. After this computation, the protocol ends.

4. If $l_i \geq t + 1$ for at least $t^* - 1$ players P_i, then each of these players P_i chooses any $(t + 1)$-subset $h_{i,1}, \ldots, h_{i,t+1}$ from his l_i values. Each P_i forms a $(t + 1)$-vector $(h_{i,1}, \cdots, h_{i,t+1})$ which is decoded using a $[t + 1, t - 1, 3]$ generalized Reed-Solomon (GRS) decoder. Finally, each P_i extracts the secret value s using the previous corrected codeword and $\mathcal{R}_i$ and the protocol ends.

Remark 6.1.5. Our choice of l_i at step 1 of Stage 3 is to insure that all participants following the protocol's instructions will obtain the same value $l_i = l$. This value l represents the number of elements which were broadcast in stage 3.

Remark 6.1.6. The goal of the check phase played at step 2 of Stage 3 is to punish a single deviating player since, in such a case, the protocol would abort and no one would learn s. Since we use a simultaneous broadcast channel to send data (step 1 of Stage 3), no player knows whether the next iteration will correspond to step 2 or step 3/4 – called *invalid-valid* in Sect. 7.2.3 – before data transmission. Furthermore, step 2.b does not reveal anything about v since the check phase run to check the consistency of at most $t - 2$ values.

Remark 6.1.7. Since $t \neq q-1$ (in fact, we have $q > n$), we cannot use Reed-Solomon codes [54] but we have to work with their generalized form.

Remark 6.1.8. The use of a generalized Reed-Solomon decoder at step 4 of Stage 3 is due to the fact that $(h_1(0), \ldots, h_n(0))$ can be interpreted as a sharing of $s+r$ using Shamir's technique with threshold parameter $t-1$ and the well-known relation between Shamir's secret sharing scheme and generalized Reed-Solomon codes [54]. Note that we cannot use the Lagrange interpolation technique directly, since we need to ensure correct secret reconstruction in the presence of (at most) one deviating player (see proof of Theorem 6.2.1 (Stage 3)).

Remark 6.1.9. Our protocol requires $t \geq 3$. Indeed, consider $t^* = t = 2$ and two participating players P_1 and P_2. If P_1 always remains silent at step 1 of Stage 3 (even if $b_1 = 1$) then P_2 is forced to permanently run step 2.b. Since P_2 follows the protocol faithfully, at some iteration, his share is to be broadcasted to P_1 (who will still be silent). Thus, P_1 will recover s and P_2 will not. To prevent a player from choosing such a strategy, we need $t-1 > 1$, that is $t \geq 3$.

Now in order to make it easy to follow our protocol, we provide a graphic representation of the information flow of our protocol on the next page.

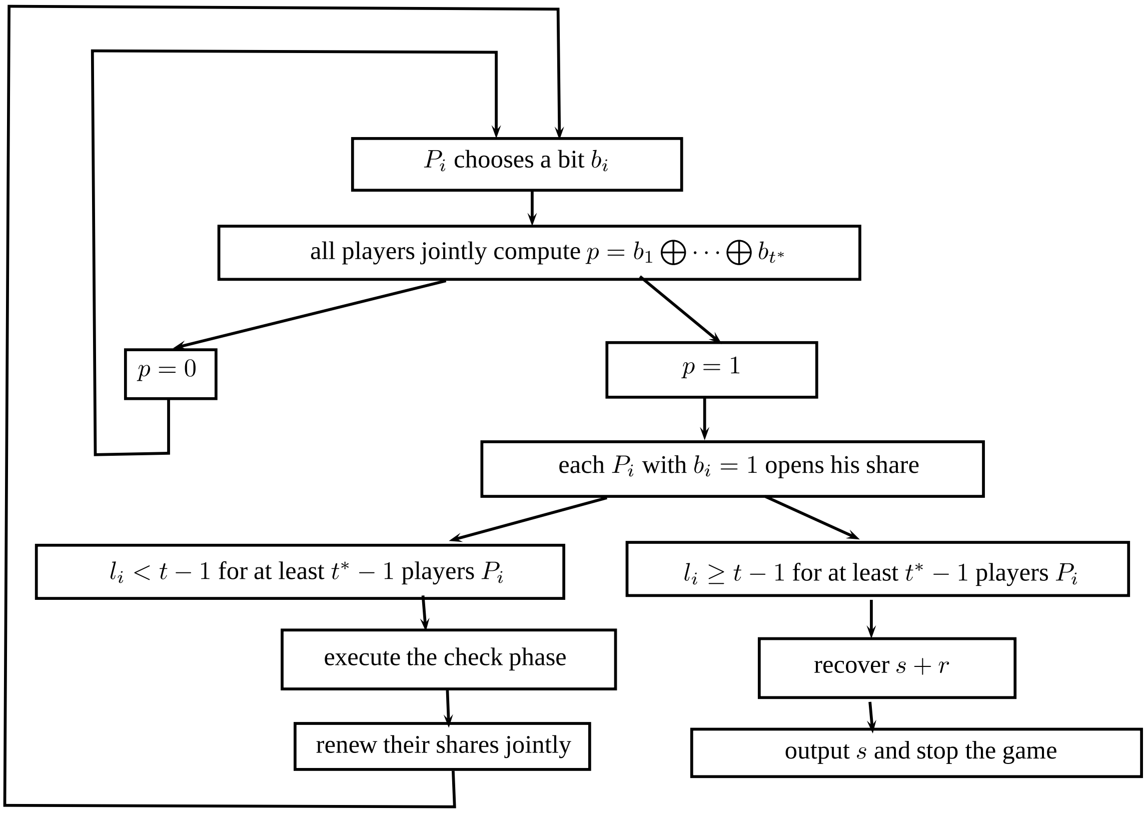
6.2 Security of our Rational Secret Sharing Scheme

We introduce the following notations:

- The secret reconstruction protocol is denoted as $\Pi(\alpha)$.
- As said at the beginning of Sect. 6.1.2, for each $i \in \{1, \ldots, n\}$, denote u_i (respectively, u_i^+) the minimal (respectively, maximal) payoff of P_i when he gains the secret and denote u_i^- his maximal payoff when P_i does not recover s. As usually assumed in the rational context, we assume that $u_i^+ > u_i > u_i^-$ for all $i \in \{1, \ldots, n\}$.

The following theorem shows the consistency of our scheme in a rational environment.

Theorem 6.2.1. *Assume that $t \geq 4$. There exists an α^* such that for any $\alpha \leq \alpha^*$, $\Pi(\alpha)$ induces a Nash equilibrium surviving iterated deletion of weakly dominated strategies.*


P_i chooses a bit b_i
all players jointly compute $p = b_1 \bigoplus \cdots \bigoplus b_{t^*}$
$p = 0$
$p = 1$
each P_i with $b_i = 1$ opens his share
$l_i < t - 1$ for at least $t^* - 1$ players P_i
$l_i \geq t - 1$ for at least $t^* - 1$ players P_i
execute the check phase
recover $s + r$
renew their shares jointly
output s and stop the game

Proof. We first show that the recommended protocol $\Pi(\alpha)$ is a Nash equilibrium.

Without loss of generality, we can assume that the active players are $P_1, \ldots, P_{t^*}$. If one of the t^* players does not broadcast anything during step 1 of Stage 1, then the protocol would terminate with nobody recovering s. This would result in a lower payoff for everybody including the deviating player. Thus, any rational player is to broadcast a value during that step. Suppose that some P_i broadcasts a fake couple $(\hat{r}_i, \hat{r}'_i)$ while the others follow the prescribed strategy. In order for the protocol not to abort, this forgery must lead to a couple of values $(\widehat{\mathcal{R}}, \widehat{\mathcal{R}}')$ such that: $\widehat{\mathcal{R}} + \widehat{\mathcal{R}}' = r + r'$ (random element uniformly distributed over $\mathbb{F}$). In other words, the deviation of P_i is successful if $(\hat{r}_i, \hat{r}'_i)$ corresponds to one of the $q - 1$ couples $(\widehat{\mathcal{R}}, \widehat{\mathcal{R}}')$ consistent with $r + r'$ and with $\hat{r}_i \neq r_i$.

Remark 6.2.2. We stress that the values $\widehat{\mathcal{R}}$ and $\widehat{\mathcal{R}}'$ are common to all $t^* - 1$ honest players.

Since we use simultaneous broadcast channels, deviating P_i must input $(\hat{r}_i, \hat{r}'_i)$ (with $\hat{r}_i \neq r_i$) before receiving any information about the shares of the honest players. As a consequence, the result of the polynomial reconstructions appears uniformly distributed to P_i. Thus, the value $\widehat{\mathcal{R}} + \widehat{\mathcal{R}}'$ appears uniformly distributed over $\mathbb{F} = \mathrm{GF}(q)$. Therefore, the expected payoff of P_i by performing this deviation is:

$$\frac{1}{q}\, u_i^+ + \left(1 - \frac{1}{q}\right)\, u_i^-$$

Thus, P_i does *not* deviate if:

$$\frac{1}{q}\, u_i^+ + \left(1 - \frac{1}{q}\right)\, u_i^- < u_i \tag{6.1}$$

Now, we have to discuss an important fact. Inequality (6.1) is *not* a restriction to our scheme. Indeed, this relation corresponds to the fact that it is more valuable for P_i to participate in the secret reconstruction process than to abort the protocol and to toss a coin to decide the value of the secret s since this selfish strategy is successful with probability $\frac{1}{q}$. As said in [27] about Fuchsbauer *et al.*'s value U_{random}, if Inequality (6.1) does not hold, then P_i has no incentive in cooperating at all and this player is better out of the group of participants. Thus, it can be assumed without loss of generality that Inequality (6.1) does hold for all n players. As a consequence, any rational player is to follow all instructions in Stage 1.

Assume that player P_i wants to cheat during Stage 2. In order for the protocol not to be terminated abruptly, P_i is to send data to P_{i^+} and P_{i^-} at step 1 and to all other $(t^* - 1)$ players at step 3. A necessary condition for a successful cheating is to have:

$$p_1 = p_2 = \cdots = p_{i^-} = p_{i^+} = \cdots = p_{t^*} = 1$$

Denote $\widetilde{d_i}, \widetilde{b'_i}$ and $\widetilde{d_{i^+} \oplus b'_{i^-}}$ the values sent by the deviating P_i during Stage 2. Let j be any value in $\{1, \ldots, t^*\} \setminus \{i\}$. We first study p_j. Following the protocol's instructions, P_j computes the bit p_j as:

$$\begin{aligned} p_j &= \underbrace{(b'_{i^{--}} \oplus \widetilde{d_i})}_{\text{from } P_{i^-}} \oplus \underbrace{(\widetilde{d_{i^+} \oplus b'_{i^-}})}_{\text{from } P_i} \oplus \underbrace{(\widetilde{b'_i} \oplus d_{i^{++}})}_{\text{from } P_{i^+}} \oplus \underbrace{[\bigoplus_{\substack{l=1 \\ l \neq i^-, i, i^+}}^{t^*} (b'_{l^-} \oplus d_{l^+})]}_{\text{from remaining players including } P_j} \\ &= (b'_{i^+} \oplus d_{i^-}) \oplus (\widetilde{d_i} \oplus \widetilde{b'_i}) \oplus (\widetilde{d_{i^+} \oplus b'_{i^-}}) \oplus [\bigoplus_{\substack{l=1 \\ l \neq i^-, i, i^+}}^{t^*} b_l] \end{aligned}$$

We notice that the value of p_j does not depend on the index j. Therefore, we rename this common value as p. It can be simplified as follows:

$$p = \underbrace{(d_{i^+} \oplus b'_{i^-})}_{\text{known to } P_i} \oplus \underbrace{(\widetilde{d_i} \oplus \widetilde{b'_i}) \oplus (\widetilde{d_{i^+} \oplus b'_{i^-}})}_{\text{chosen by } P_i} \oplus [\bigoplus_{\substack{l=1\\ l\neq i}}^{t^*} b_l] \tag{6.2}$$

Denote w_i the Hamming weight of the vector $(b_1, \ldots, b_{i^-}, b_{i^+}, \ldots, b_{t^*})$. We have two cases to consider:

1. $\widetilde{d_i}, \widetilde{b'_i}$ and $\widetilde{d_{i^+} \oplus b'_{i^-}}$ are such that $p = 1 \oplus [\bigoplus_{\substack{l=1\\ l\neq i}}^{t^*} b_l]$.

2. $\widetilde{d_i}, \widetilde{b'_i}$ and $\widetilde{d_{i^+} \oplus b'_{i^-}}$ are such that $p = \bigoplus_{\substack{l=1\\ l\neq i}}^{t^*} b_l$.

<u>Case 1</u>. $p = 1$ and w_i is even. If the number of honest participating players to send their shares at step 1 of Stage 3 is either at most $t-4$ or at least t, then P_i does not gain anything by deviating. P_i only benefits from not following the protocol's instructions when there are exactly $t-2$ honest participating players. In this case, P_i's expected payoff is u_i^+.

<u>Case 2</u>. $p = 1$ and w_i is odd. If the number of honest participating players to send their shares at step 1 of Stage 3 is either at most $t-3$ or at least $t+1$, then P_i does not gain anything by deviating. P_i only benefits from not following the protocol's instructions when there are exactly $t-1$ honest participating players. Then, P_i's expected payoff is u_i^+.

Reaching this point in our reasoning, we need to reflect upon P_i's strategy. We showed that he only has an incentive to deviate in two specific subcases of Case 2 (assuming that $p = 1$): $w_i = t-2$ and $w_i = t-1$. However, if P_i follows the instructions at Stage 2, then each of the remaining (t^*-1) players would get:

$$p = \bigoplus_{l=1}^{t^*} b_l \tag{6.3}$$

So, P_i would obtain the same benefit in cheating at step 1 of Stage 3 as in the subcases above where the role of $(d_{i^+} \oplus b'_{i^-}) \oplus (\widetilde{d_i} \oplus \widetilde{b'_i}) \oplus (\widetilde{d_{i^+} \oplus b'_{i^-}})$ from Eq. (6.2) would be played by b_i in

Eq. (6.3). Therefore, P_i has no incentive in sending different values than those prescribed by the protocol during Stage 2.

We now focus on potential deviations of P_i during Stage 3. There are three possibilities for P_i to cheat at step 1 of Stage 3 (where $p = 1$):

1. P_i broadcasts some value despite $b_i = 0$.
2. P_i does not broadcast anything despite $b_i = 1$.
3. P_i broadcasts a fake share when $b_i = 1$.

Case 1. The protocol will terminate during this iteration of Stage 3 since the value l_j will be even for $j \neq i$ (step 2.b cannot be accessed). In this situation, P_i is better sending a fake share value. Since $p = 1$, w_i is odd. We are in the same situation as in Case 2 of Stage 2 where P_i sends a fake share.

As said above, if $w_i \leq t - 3$ then no player learns s and P_i's expected payoff is u_i^-.

If $w_i \geq t + 1$ then everybody recovers s:

- P_i interpolates $t - 2$ of the w_i shares he got from the other players (he runs step 3).
- Each P_j ($j \neq i$) is to execute step 4. Since there is at most one incorrect value amongst the $(t + 1)$ elements he chooses, this potential error is to be corrected by the GRS decoder and P_j recovers s.

In this situation, the expected payoff of the cheating P_i is u_i.

If $w_i = t - 1$, then only P_i will recover s, since the remaining players will reconstruct an incorrect polynomial at step 3. In such a situation, P_i gets at most u_i^+. We need to compute the following three probabilities:

$$\Pr[w_i \geq t + 1 | \{p = 1\} \cap \{b_i = 0\}]$$

$$\Pr[w_i = t - 1 | \{p = 1\} \cap \{b_i = 0\}]$$

$$\Pr[w_i \leq t - 3 | \{p = 1\} \cap \{b_i = 0\}]$$

Let λ be any element of $\{0, \ldots, t^* - 1\}$.

$$\begin{aligned}\Pr[w_i = \lambda | \{p = 1\} \cap \{b_i = 0\}] &= \frac{\Pr(\{w_i = \lambda\} \cap \{\bigoplus_{l=1}^{t^*} b_l = 1\} \cap \{b_i = 0\})}{\Pr[\{\bigoplus_{l=1}^{t^*} b_l = 1\} \cap \{b_i = 0\}]} \\ &= \frac{\Pr[\{w_i = \lambda\} \cap \{w_i \text{ is odd}\}]}{\Pr[w_i \text{ is odd}]} \\ &= \begin{cases} 0 & \text{if } \lambda \text{ is even} \\ \frac{\Pr[w_i = \lambda]}{\Pr[w_i \text{ is odd}]} & \text{if } \lambda \text{ is odd} \end{cases}\end{aligned}$$

We can now compute the probabilistic values we need using the fact that t is even.

$$\Pr[w_i \geq t - 1 | \{p = 1\} \cap \{b_i = 0\}] = \frac{\sum_{\lambda = t+1, \lambda \text{ odd}}^{t^*-1} \Pr[w_i = \lambda]}{\Pr[w_i \text{ is odd}]}$$

$$\Pr[w_i \leq t - 3 | \{p = 1\} \cap \{b_i = 0\}] = \frac{\sum_{\lambda = 0, \lambda \text{ odd}}^{t-3} \Pr(w_i = \lambda)}{\Pr[w_i \text{ is odd}]}$$

Based on this analysis, P_i is ***not*** to cheat if:

$$u_i^+ \Pr[w_i = t - 1] + u_i \sum_{\lambda = t+1, \lambda \text{ odd}}^{t^*-1} \Pr[w_i = \lambda] + u_i^- \sum_{\lambda = 0, \lambda \text{ odd}}^{t-3} \Pr[w_i = \lambda] \leq u_i \Pr[w_i \text{ is odd}]$$

The previous inequality is equivalent to:

$$u_i^+ \alpha^{t-1} (1 - \alpha)^{t^* - t} \binom{t^* - 1}{t - 1} + u_i^- \sum_{\substack{\lambda = 0 \\ \lambda \text{ odd}}}^{t-3} \alpha^{\lambda} (1 - \alpha)^{t^* - (\lambda+1)} \binom{t^* - 1}{\lambda}$$

$$\leq$$

$$u_i \sum_{\substack{\lambda = 0 \\ \lambda \text{ odd}}}^{t-1} \alpha^{\lambda} (1 - \alpha)^{t^* - (\lambda+1)} \binom{t^* - 1}{\lambda}$$

Since t is even, the sum on the right hand side ends when $\lambda = t - 3$. We get:

$$(u_i^+ - u_i)\,\alpha^{t-1}\,(1-\alpha)^{t^*-t}\binom{t^*-1}{t-1} \leq (u_i - u_i^-) \sum_{\substack{\lambda=0 \\ \lambda \text{ odd}}}^{t-3} \alpha^{\lambda}\,(1-\alpha)^{t^*-(\lambda+1)}\binom{t^*-1}{\lambda}$$

We divide both sides of the previous inequality by $\alpha^{t-1}\,(1-\alpha)^{t^*-t}(u_i - u_i^-)$. Defining $\Lambda := \frac{1-\alpha}{\alpha}$, we obtain:

$$\frac{u_i^+ - u_i}{u_i - u_i^-}\binom{t^*-1}{t-1} \leq \sum_{\substack{\lambda=0 \\ \lambda \text{ odd}}}^{t-3} \binom{t^*-1}{\lambda}\Lambda^{t-(\lambda+1)} \tag{6.4}$$

The right hand side of Inequality (6.4) is a polynomial of degree $t-2$ in Λ with a positive leading coefficient as soon as $t-3 \geq 1$ (i.e. $t \geq 4$). Since $\Lambda \xrightarrow[\alpha \to 0_+]{} +\infty$, we deduce that there exists a value $\alpha_{i,1}$ such that for all $\alpha \leq \alpha_{i,1}$, Inequality (6.4) does hold. In such a situation, P_i does ***not*** cheat as indicated in Case 1.

Case 2. As in Case 1, the protocol will terminate during this iteration of Stage 3. In this case, both t and w_i are even. We are in the same situation as in Case 1 of Stage 2 where P_i remains silent.

If $w_i \leq t-4$ then no player learns s and P_i's expected payoff is u_i^-. If $w_i \geq t$ then everybody runs step 3 and recovers s since all the shares are genuine. The expected payoff of the cheating P_i is u_i. When $w_i \leq t-2$, P_i will be the only player to recover s since all other participants will run step 2.a. In this situation, P_i gets at most u_i^+. We are interested in the following three probabilities:

$$\Pr[w_i \geq t | \{p = 1\} \cap \{b_i = 1\}]$$

$$\Pr[w_i = t-2 | \{p = 1\} \cap \{b_i = 1\}]$$

$$\Pr[w_i \leq t-4 | \{p = 1\} \cap \{b_i = 1\}]$$

Let λ be any element of $\{0, \dots, t^*\}$.

$$\begin{aligned}
\Pr[w_i = \lambda | \{p = 1\} \cap \{b_i = 1\}] &= \frac{\Pr[\{w_i = \lambda\} \cap \{\bigoplus_{l=1}^{t^*} b_l = 1\} \cap \{b_i = 1\}]}{\Pr[\{\bigoplus_{l=1}^{t^*} b_l = 1\} \cap \{b_i = 1\}]} \\
&= \frac{\Pr[\{w_i = \lambda\} \cap \{w_i \text{ is even}\}]}{\Pr[w_i \text{ is even}]} \\
&= \begin{cases} 0 & \text{if } \lambda \text{ is odd} \\ \frac{\Pr[\{w_i = \lambda\}]}{\Pr[w_i \text{ is even}]} & \text{if } \lambda \text{ is even} \end{cases}
\end{aligned}$$

We can now compute the probabilistic values we need.

$$\Pr[w_i \geq t | \{p = 1\} \cap \{b_i = 1\}] = \frac{\sum_{\lambda = t, \lambda \text{ even}}^{t^*-1} \Pr[w_i = \lambda]}{\Pr[w_i \text{ is even}]}$$

$$\Pr[w_i \leq t - 4 | \{p = 1\} \cap \{b_i = 1\}] = \frac{\sum_{\lambda = 0, \lambda \text{ even}}^{t-4} \Pr[w_i = \lambda]}{\Pr[w_i \text{ is even}]}$$

Based on this analysis, P_i is ***not*** to cheat if:

$$u_i^+ \Pr[w_i = t - 2] + u_i \sum_{\substack{\lambda = t \\ \lambda \text{ even}}}^{t^*-1} \Pr[w_i = \lambda] + u_i^- \sum_{\substack{\lambda = 0 \\ \lambda \text{ even}}}^{t-4} \Pr[w_i = \lambda]$$

$$\leq$$

$$u_i \Pr[w_i \text{ is even}]$$

The previous inequality is equivalent to:

$$u_i^+ \alpha^{t-2} (1 - \alpha)^{t^* - (t-1)} \binom{t^* - 1}{t - 2} + u_i^- \sum_{\lambda = 0, \lambda \text{ even}}^{t-4} \alpha^\lambda (1 - \alpha)^{t^* - (\lambda+1)} \binom{t^* - 1}{\lambda}$$

$$\leq$$

$$u_i \sum_{\lambda = 0, \lambda \text{ even}}^{t-1} \alpha^\lambda (1 - \alpha)^{t^* - (\lambda+1)} \binom{t^* - 1}{\lambda}$$

Since t is even, the sum on the right hand side ends when $\lambda = t-2$. We get:

$$(u_i^+ - u_i)\,\alpha^{t-2}\,(1-\alpha)^{t^*-(t-1)}\binom{t^*-1}{t-2} \leq (u_i - u_i^-) \sum_{\substack{\lambda=0 \\ \lambda \text{ even}}}^{t-4} \alpha^{\lambda}\,(1-\alpha)^{t^*-(\lambda+1)}\binom{t^*-1}{\lambda}$$

We divide both sides of the previous inequality by $\alpha^{t-2}\,(1-\alpha)^{t^*-(t-1)}(u_i - u_i^-)$. Since $\Lambda = \frac{1-\alpha}{\alpha}$, we obtain:

$$\frac{u_i^+ - u_i}{u_i - u_i^-}\binom{t^*-1}{t-2} \leq \sum_{\substack{\lambda=0 \\ \lambda \text{ even}}}^{t-4} \binom{t^*-1}{\lambda}\Lambda^{t-(\lambda+2)} \tag{6.5}$$

The right hand side of Inequality (6.5) is a polynomial of degree $t-2$ in Λ with a positive leading coefficient. Since $\Lambda \xrightarrow[\alpha\to 0_+]{} +\infty$, we deduce that there exists a value $\alpha_{i,2}$ such that for all $\alpha \leq \alpha_{i,2}$, Inequality (6.5) hold. In such a situation, P_i does ***not*** cheat as indicated in Case 2.

Case 3. The current case corresponds to Case 1 of Stage 2 where P_i sends a fake share. We are essentially in the same situation as in Case 2. Given t is even, we have: $w_i \geq t$ (everybody recovers s) or $w_i \leq t-4$ (nobody recovers s) or $w_i = t-2$. In the latter subcase, P_i will recover s while the other players will get a wrong value with large probability. The expected payoff of P_i is at most u_i^+ in that specific subcase. As a consequence, if Inequality (6.5) holds (i.e. we set $\alpha_{i,3} := \alpha_{i,2}$) then P_i does ***not*** cheat as indicated in Case 3.

At this point of our proof, we showed that for any $\alpha \leq \alpha_i^* := \min(\alpha_{i,1}, \alpha_{i,2})$, player P_i will follow the protocol's instructions until step 1 of Stage 3 is included. Since steps 2.a, 3 and 4 of Stage 3 are run independently by each player, the only remaining way for P_i to deviate would occur when step 2.b is executed. In the check phase, each single deviation will cause the protocol to stop without anybody learning the secret while, in the renewal phase, a single deviation either may cause every player to recover a wrong secret or may cause the protocol to stop with nobody learning s. Hence, in both cases, no single rational player P_i has any incentive to deviate during step 2.b.

As a consequence, for any $\alpha \leq \alpha^* := \min(\alpha_1^*, \dots, \alpha_n^*)$, $\Pi(\alpha)$ is a Nash equilibrium. Using the same argument as the proof in Theorem 3.2 from [31], we could demonstrate that $\Pi(\alpha)$ survives iterated deletion of weakly-dominated strategies.

□

Explicitly computing the largest possible value for α^* is not trivial. Fortunately, we can obtain an explicit bound more easily. We define the values $\beta := \max_{i=1,\ldots,n} \frac{u_i^+ - u_i}{u_i - u_i^-}$ and $\widetilde{\alpha} := \left[(\frac{t^*}{t-3} - 1)\sqrt{\beta} + 1 \right]^{-1}$.

Theorem 6.2.3. *Assume that $t \geq 4$. For any $\alpha \leq \widetilde{\alpha}$, $\Pi(\alpha)$ induces a Nash equilibrium surviving iterated deletion of weakly dominated strategies.*

Proof. Assume that $t \geq 4$. Consider $\alpha \leq \widetilde{\alpha}$. This inequality implies:

$$\beta \left(\frac{t^* - t + 3}{t-3} \right)^2 \leq \Lambda^2 \tag{6.6}$$

This inequality leads to:

$$\beta \, \frac{(t^* - t + 2)\,(t^* - t + 1)}{(t-1)\,(t-2)} \leq \Lambda^2.$$

Therefore, we get: $\beta \binom{t^*-1}{t-1} \leq \binom{t^*-1}{t-3} \Lambda^2$ and we deduce:

$$\frac{u_i^+ - u_i}{u_i - u_i^-} \binom{t^*-1}{t-1} \leq \sum_{\substack{\lambda = 0 \\ \lambda \text{ odd}}}^{t-3} \binom{t^*-1}{\lambda} \Lambda^{t-(\lambda+1)},$$

which means that Inequality (6.4) is verified.

Let's start from Inequality (6.6) again. It also implies:

$$\beta \, \frac{(t^* - t + 3)\,(t^* - t + 2)}{(t-2)\,(t-3)} \leq \Lambda^2$$

Therefore, we get: $\beta \binom{t^*-1}{t-2} \leq \binom{t^*-1}{t-4} \Lambda^2$ and we further deduce:

$$\frac{u_i^+ - u_i}{u_i - u_i^-} \binom{t^*-1}{t-2} \leq \sum_{\substack{\lambda = 0 \\ \lambda \text{ even}}}^{t-4} \binom{t^*-1}{\lambda} \Lambda^{t-(\lambda+2)},$$

which means that Inequality (6.5) is verified. It ends our demonstration since this result is valid for every player P_i $(i \in \{1, \ldots, n\})$. □

6.3 Round Complexity

We use the same notations as in the previous section. The following result gives an upper bound on the round complexity of our protocol.

Theorem 6.3.1 (Upper bound). *Assume that $t \geq 4$. Let α^* be as in Theorem 6.2.1. For any $\alpha \leq \alpha^*$, the expected round complexity of $\Pi(\alpha)$ is:*

$$\frac{1}{\displaystyle\sum_{j=\frac{t}{2}}^{\lceil\frac{t^*}{2}\rceil} \alpha^{2j-1}(1-\alpha)^{t^*-(2j-1)}\binom{t^*}{2j-1}},$$

which is $O(\frac{\alpha^{-t^*}}{t^*})$.

Proof. Since $\alpha \leq \alpha^*$, all t^* active players follow the protocol's instructions. As in the proof of Theorem 6.2.1, we assume that the active players are $P_1, \ldots, P_{t^*}$.

Based on Eq. (6.3), the secret s is to be recovered when $p = 1$ and at least $t-1$ of the b_i's are equal to 1. Denote w the Hamming weight of $(b_1, \ldots, b_{t^*})$. Since the b_i's are chosen uniformly at random and independently, we have:

$$\begin{aligned}
\Pr[s \text{ is recovered}] &= \sum_{\lambda=t-1}^{t^*} \Pr[\{p=1\} \cap \{w=\lambda\}] \\
&= \sum_{\substack{\lambda = t-1 \\ \lambda \text{ odd}}}^{t^*} \Pr[w=\lambda] \\
&= \sum_{\substack{\lambda = t-1 \\ \lambda \text{ odd}}}^{t^*} \alpha^{\lambda}\,(1-\alpha)^{t^*-\lambda}\binom{t^*}{\lambda}.
\end{aligned}$$

Since $\Pi(\alpha)$ is a Nash equilibrium for small values of α, we can assume: $\alpha \leq \frac{1}{2}$. Thus, $1-\alpha \geq \alpha$ and we get the lower bound:

$$\Pr(s \text{ is recovered}) \geq \sum_{\substack{\lambda = t-1 \\ \lambda \text{ odd}}}^{t^*} \alpha^{t^*}\binom{t^*}{\lambda} \geq t^*\,\alpha^{t^*}.$$

In other words, the expected round complexity is $O(\frac{\alpha^{-t^*}}{t^*})$. □

We now show a lower bound on the round complexity of our protocol. The demonstration of the following theorem relies on the Chernoff's bound on the tail of the binomial distribution [15].

Theorem 6.3.2 (Lower bound). *Assume that* $t \geq 4$. *For any* $\alpha \leq \min(\alpha^*, \frac{t-2}{t^*-1})$, *the expected round complexity of* $\Pi(\alpha)$ *is:*

$$\Omega\left(\left(\frac{t-2}{t^*-1}\,\alpha^{-1}\right)^{t-2} \quad \frac{e^{\alpha(t^*-1)-t+2}}{1-\alpha}\right).$$

Proof. In order to bound the value Pr[s is recovered], we expand this expression as follows.

$$\begin{aligned}
\Pr[s \text{ is recovered}] &= \sum_{\substack{\lambda = t-1 \\ \lambda \text{ odd}}}^{t^*} \quad \alpha^{\lambda}\,(1-\alpha)^{t^*-\lambda} \left[\binom{t^*-1}{\lambda} + \binom{t^*-1}{\lambda-1}\right] \\
&= (1-\alpha) \quad \sum_{\substack{\lambda = t-1 \\ \lambda \text{ odd}}}^{t^*-1} \quad \alpha^{\lambda}\,(1-\alpha)^{t^*-1-\lambda}\binom{t^*-1}{\lambda} \\
&\qquad + \\
&\quad \alpha \quad \sum_{\substack{\lambda = t-2 \\ \lambda \text{ even}}}^{t^*-1} \quad \alpha^{\lambda}\,(1-\alpha)^{t^*-1-\lambda}\binom{t^*-1}{\lambda}.
\end{aligned}$$

Since t is even, the indices for both sums can start from $t-2$.

$$\begin{aligned}
\Pr[s \text{ is recovered}] = (1-\alpha) &\quad \sum_{\substack{\lambda = t-2 \\ \lambda \text{ odd}}}^{t^*-1} \quad \alpha^{\lambda}\,(1-\alpha)^{t^*-1-\lambda}\binom{t^*-1}{\lambda} \\
&\qquad + \\
\alpha &\quad \sum_{\substack{\lambda = t-2 \\ \lambda \text{ even}}}^{t^*-1} \quad \alpha^{\lambda}\,(1-\alpha)^{t^*-1-\lambda}\binom{t^*-1}{\lambda}.
\end{aligned}$$

Consider the following values:

$$\forall \ell \in \mathbb{N}\ \forall u\{0,\ldots,\ell\}\ \forall \theta \in [0,1] \quad B(u,\ell,\theta) := \sum_{\lambda=u}^{\ell} \theta\,(1-\theta)^{\ell-\lambda}\binom{\ell}{\lambda}.$$

As said before, we can always assume that $\alpha \leq \frac{1}{2}$. We get the following bounds:

$$\alpha\, B(t-2,t^*-1,\alpha) \leq \Pr[s \text{ is recovered}] \leq (1-\alpha)\, B(t-2,t^*-1,\alpha).$$

As recalled in [61], since $\alpha \leq \frac{t-2}{t^*-1}$, we have the Chernoff bound:

$$B(t-2,t^*-1,\alpha) \leq \left(\frac{\alpha\,(t^*-1)}{t-2}\right)^{t-2} e^{t-2-\alpha(t^*-1)},$$

which leads to the claimed lower bound on the expected round complexity. □

6.4 Remark on the Case When t Is Odd

Since the beginning of Sect. 6.1, we only considered the case when t is even. When the threshold t is odd, we can essentially use the same protocol with the exception of step 4 in Stage 2 and step 2.a in Stage 3 which become:

Stage 2 (update)

4. If $p_j = 1$ for some P_j, then the whole protocol goes back to the first step of Stage 2. Otherwise, all participants proceed to Stage 3.

Stage 3 (update)

2.a. If l_i is odd for at least $t^* - 1$ players P_i, then the protocol stops.

Similar security and efficiency theorems to those presented in the past two sections hold. The only analytical difference lies in the fact that we now have $p = 0$.

6.5 Discussion

Equilibrium. Our solution concept is based on Nash equilibria surviving iterated deletions of weakly dominated strategies. We are aware that the notion of iterated deletion exhibits several problems [38] and that several new concepts have been proposed (mainly using computational versions of Nash equilibria [27, 38]). The purpose of this construction is not to advocate in favor of a specific type of equilibrium. Its primary goal is to present a new construction combining the advantages of several schemes for a model widely studied in the literature.

Communication Channels. Our rational protocol requires the presence of simultaneous broadcast channels which is a commonly-used model for rational SSS. In [38], Kol and Naor manage to remove the need of simultaneity for the broadcast channels. However, this is at the expense of increasing the round complexity by a multiplicative t and the removal is based on permutations to relocate the meaningful encryption key. Thus, this process is related to the use of their *meaningful/meaningless* encryption primitive. In [27], Fuchsbauer *et al.* only use point-to-point channels. However, authentication needs to use the *verifiable random functions*, which are computational primitives.

Share Consistency. We use an error-correcting code to check the consistency of the shares. Another possibility would have been to use information theoretically secure MACs. The issue is that these MACs encounter a small authentication error probability while our coding-based approach does not.

Computation Efficiency. Several rational protocols (such as [31]) require the dealer to participate in every round of the secret reconstruction phase. This is a bottleneck for the efficiency of those constructions. Like [27, 38], our scheme does not require the presence of an online dealer. Furthermore, our share renewal process does not rely on either complex MPC protocols (contrary to [1, 2, 29]) or complicated hardware such as envelopes and ballot boxes (contrary to [34, 43]).

Backward Induction Attack. In [38], Kol and Naor emphasized that techniques from [1, 29, 31] were susceptible to backward induction attacks resulting in all players remaining silent from the beginning of the secret reconstruction process. This attack requires an exponential number of rounds to succeed. Such a large running time only occurs with negligible probability. Nonetheless, our scheme is immune against this threat since we only use information theoretical tools (one-time pads) to authenticate data. In particular, our immunity does not require the existence of any additional cryptographic primitive contrary to [38] where meaningful/meaningless encryption schemes were used.

6.6 Conclusion

In this chapter, we have presented a new protocol for rational threshold secret sharing based on symmetric polynomials. To the best of our knowledge, it is the first time that such polynomials have been used for rational secret sharing. Our protocol requires simultaneous broadcast channels and works for any $t \geq 4$. This construction does not require the presence of the dealer during the share reconstruction phase and it provides information theoretical security. It is immune against the backward induction attack and it leads to a Nash equilibrium surviving the iterated deletion of weakly dominated strategies.

On the negative side, our scheme is only secure against single strategy deviations. One line to follow for our future research is to extend this scheme to handle the case of coalition of enemies (c-resilience for $c \geq 2$). Using bivariate polynomials of degree $t - 2\,c$ is a possible approach as the corresponding GRS codes can correct up to c errors. It is not hard to see that the information theoretical security provided by the one-time pads still holds (for any coalition of size at most $\frac{t-1}{2}$). Furthermore, the check phase would still be consistent as every player P_j (testing the validity of λ sent by P_i) would get $t^* - c \geq t - 2\,c + 1$ correct broadcast elements at the end of step 1. The tricky point with this approach is to perform the probabilistic analysis of a group of c cheaters. Indeed, the probabilistic formulas exposed in the security proof cannot be simplified as easily since we have to handle a set of c bits which may not have been chosen by the cheaters as stated in the algorithm. In the case $c = 1$, we obtained simple conditional probabilities. This is no longer the case for coalitions of size $c \geq 2$ as some cheaters may choose their bit as they feel best for themselves.

As said in Sect. 6.5, our solution concept is based on Nash equilibria surviving iterated deletions of weakly dominated strategies and several others approaches in designing rational protocols have recently been proposed. Since the cryptographic community is still in search of a proper framework for rational protocols, it would be interesting to study the benefits of our approach in different equilibrium contexts.

7. AN EFFICIENT RATIONAL SECRET SHARING SCHEME BASED ON THE CHINESE REMAINDER THEOREM

In this chapter, we propose an efficient protocol for rational t-out-of-n secret sharing based on the Chinese Remainder Theorem without an on-line dealer. Instead, we borrow the idea from *Joint Random Secret Sharing* to allow active players to jointly form his "one-time" share related to a random element at the beginning of each iteration, merely by interactions among the group of $t^*(t^* \geq t)$ participants. Under some computational assumptions related to the discrete logarithm problem and RSA, this construction leads to a $(t-1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles. Our protocol does not rely on a simultaneous channel, instead, it only requires a synchronous broadcast channel and pairwise secure channels. Compared with the protocol in [27] that works in almost the same model as ours, our protocol has smaller share size even when $(t-1)$-resilience to coalitions is required. More explicitly, our shares are $O(k)$ bits long while those from [27] need $(n-t+1)\,(2n|s|+O(k))$ bits. In fact, what we demonstrate in this chapter is an updated version of the publication [62], which is a joint work with C. Tartary and H. Wang. The main idea of our construction is described as follows.

In the share distribution phase, the dealer uses the modified version of the Asmuth-Bloom Secret Sharing Scheme [3] proposed by Kaya and Selçuk [36] (with further modification which is necessary for our needs) to generate n shares for the secret s. Suppose that there are t^* players active in the reconstruction phase, say $P_1, \ldots, P_{t^*}$. This phase proceeds with several rounds. At the beginning of each iteration, the "one-time" shares for $(s+d)$ mod p_0 are generated (jointly by the active players) using the technique from *Joint Random Secret Sharing*, where $d = d^{(1)} + \cdots + d^{(t^*)}$ and each $d^{(i)}$ is chosen by player P_i independently and uniformly at random from $\mathbb{Z}_{p_0}$, the domain of the secret. If $d \equiv 0$ mod p_0, then all the "one-time" shares are valid for recovering s, and in this sense, the current iteration is called the *valid* iteration. Otherwise, the current iteration is *invalid*, which is designed only for catching possible cheaters. Each communicated message carries a commitment that is perfect binding and computational hiding (under the computational assumption on the hardness of computing discrete logarithm). Thus, at every point of our protocol, there is a unique legal message that each player can send (except with negligible probability).

Then, all the active players are required to open their "one-time" shares related to a random element $(s + d) \bmod p_0$. After each player P_i has received the "one-time" shares from all the other active players, he is required to open $h^{y^{(i)}} \bmod Q$, which provides a unique way for the participants to jointly identify whether $d \equiv 0 \bmod p_0$ and consequently identify the valid iteration. If the current iteration is invalid, all the players are asked to restart a new iteration; otherwise, the secret s is recovered and the protocol terminates immediately after this iteration. In this way, no player can identify the valid iteration before he opens his "one-time" share. Furthermore, each player can identify the valid iteration only after it has occurred, that is, once a player learns that the current iteration is valid, each player has already got the real secret. Due to this, we do not need simultaneous broadcast channels. Our protocol is efficient in that the round complexity and computation complexity are both polynomial (in the security parameter k). However, our protocol relies on the assumption that no player knows auxiliary information about the secret s, which has been proved to be inherent in the non-simultaneous channels model [2].

Like classical secret sharing protocols, our protocol contains two phases: share distribution phase and secret reconstruction phase. The dealer is available only in the initial share distribution phase, during which he is assumed to be honest. We assume the existence of synchronous (but non-simultaneous) broadcast channels for all participating players and the presence of secure channels between any pair of these players and the dealer. We stress that all n players are assumed to be computationally bounded and we only consider probabilistic polynomial time (PPT for short) deviations. In the following, let k be a security parameter.

7.1 Initial Share Distribution Phase

This is the only phase in which the dealer is active. His goal is to distribute s over $\mathcal{P} := \{P_1, \ldots, P_n\}$ using the the Asmuth-Bloom Secret Sharing Scheme with threshold t. As mentioned above, we adopt the modified version of the Asmuth-Bloom Secret Sharing Scheme proposed by Kaya and Selçuk [36] and make further modifications (mainly on the parameters settings) to meet our needs. This initial share phase is comprised of two stages.

Initial Share Phase

1. Parameters Setup

To share a secret s, the dealer chooses $p_0 (> s)$ and publishes it. This value p_0 should also be lower bounded by a value depending on players' utilities and discussed later in this chapter.

1. The dealer chooses and publishes a set of prime integers $p_1, \dots, p_n$ of bit length at least k (here k is a security parameter) such that the following requirements are satisfied:

   (a) $p_0 < p_1 < \cdots < p_n$;

   (b) $\prod_{i=1}^{t} p_i > (n+1)p_0^2 \prod_{i=1}^{t-1} p_{n-i+1}$;

   (c) $q_j = 2p_j + 1$ is prime for any $1 \leq j \leq n$.

2. For any $1 \leq i \leq n$, let G_i be a subgroup of $\mathbb{Z}_{q_i}^*$ of order p_i and denote g_i a generator of G_i. Let $Q = \prod_{i=1}^{n} q_i$. He computes $g = (\sum_{i=1}^{n} g_i \cdot Q_i' \cdot \frac{Q}{q_i}) \bmod Q$ along with $x = (\sum_{i=1}^{t} g_i \cdot Q_i' \cdot \frac{Q}{q_i}) \bmod Q$, where Q_i' is the inverse of $\frac{Q}{q_i}$ in $\mathbb{Z}_{q_i}^*$, for $1 \leq i \leq n$. Let e be the order of x in $\mathbb{Z}_Q^*$. Then $\prod_{i=1}^{t} p_i$ divides e. Let $h := x^{\frac{e}{\prod_{i=1}^{t} p_i}} \in \mathbb{Z}_Q^*$. Then the order of h is $\prod_{i=1}^{t} p_i$. The dealer publishes g, h along with a sorted list $\{h^{i \cdot p_0} \bmod Q | 1 \leq i \leq \lfloor \frac{\prod_{j=1}^{t} p_j}{p_0} \rfloor\}$.

3. The dealer chooses and publishes an RSA modulus N of length at least k whose factorization is unknown to any of the n players.

2. Share Distribution Phase

To share a secret $s \in \mathbb{Z}_{p_0}$ among a group of n players $\{P_1, \dots, P_n\}$, the dealer executes the following steps.

1. He sets $M := \left\lfloor \frac{\prod_{i=1}^{t} p_i}{n+1} \right\rfloor$. He computes $y = s + A_0 \cdot p_0$ for some positive integer A_0 generated randomly subject to the condition that $0 < y < M$, calculates $y_i = y \bmod p_i$ and finally sends the share y_i to player P_i secretly, for $1 \leq i \leq n$.

2. He computes $E(y) := g^y \bmod QN$ and broadcasts $E(y)$.

Remark 7.1.1. The value g is the unique integer in $\mathbb{Z}_Q$ satisfying $g_i \equiv g \bmod q_i$, for all $1 \leq i \leq n$. Besides, the order of g in $\mathbb{Z}_{QN}^*$ is at least $\prod_{j=1}^{n} p_j$ and for each $1 \leq i \leq n$, we have:

$$E(y) \bmod q_i = (g^y \bmod QN) \bmod q_i = g^y \bmod q_i = g_i^{y_i} \bmod q_i$$

Hence, during the whole protocol, we use $(E(y) \bmod q_i)$ as a commitment to y_i, which is perfect binding but is computational hiding. In other words, the committer cannot commit himself to two values y_i and y_i' by the same commitment value, and under the assumption that computing discrete logarithm is intractable in $\mathbb{Z}_{q_i}$, no PPT player learns y_i from $E(y) \bmod q_i$ except with negligible probability in k. This allows players to check the consistency of the received data. Since the dealer is assumed to be honest, $E(y)$ is only used to detect the players' possible malicious behavior during the reconstruction process described in the next section.

7.2 Secret Reconstruction Phase

We assume that $t^*(\geq t)$ players participate in the secret reconstruction phase. For ease of description, we can assume without loss of generality that those players are $P_1, \ldots, P_{t^*}$. The reconstruction phase proceeds in a series of iterations, each of which consists of multiple communication rounds among those players. First we propose two sub-protocols to be called upon within the reconstruction phase.

7.2.1 Share Update Phase

This is done by the players participating in the secret reconstruction process, namely, by $P_1, \ldots, P_{t^*}$. In this phase, each participating P_i (sorted in index increasing order) plays a similar role to the dealer's (initial share phase) to share a random element $d^{(i)} \in \mathbb{Z}_{p_0}$ and finally get his "one-time" share for $(s + d^{(1)} + \cdots + d^{(t^*)}) \bmod p_0$.

In [36], in order to prevent the dealer from distributing inconsistent shares, the range-proof technique proposed in [13] is used to allow the dealer to convince each player that some committed integer lies in a particular interval. This range proof is statistically zero-knowledge in the random-oracle model. Besides, under some computational assumptions related to the discrete logarithm problem and RSA, a cheating dealer can only succeed with negligible probability (in k). We refer to [13, 36] for further details.

Here, in order to prevent each player P_i from distributing inconsistent shares for his random chosen $d^{(i)}$, we need to apply this range-proof technique. Throughout this chapter, we will use $\mathsf{RngPrf}(E(y), M)$ to denote the Cao-Liu's non-interactive range proof that a secret integer y committed with $E(y)$ lies in the interval $[0, M)$ [13]. In the following share update phase, we will use

$\mathsf{RngPrf}(E(y), M)$ as a black box and we refer to [13] for additional information.

Share Update Phase

1. Each P_i selects a random element $d^{(i)} \in \mathbb{Z}_{p_0}$ uniformly and independently. He computes $y^{(i)} = A_i \cdot p_0 + d^{(i)}$, where A_i is a positive integer chosen randomly conditioned on $0 < y^{(i)} < M$. Then he computes $y_j^{(i)} := y^{(i)} \bmod p_j$ along with $E(y^{(i)}) := g^{y^{(i)}} \bmod QN$, and he finally sends $y_j^{(i)}$ to player P_j secretly through a secure channel for each $j \neq i$. In addition, P_i broadcasts $E(h^{y^{(i)}} \bmod Q) := g^{(h^{y^{(i)}} \bmod Q)} \bmod QN$, $E(y^{(i)})$ and $\mathsf{RngPrf}(E(y^{(i)}), M)$.

2. If player P_i only receives partial messages (hereinafter, partial messages including the case of no message at all), then he outputs a random guess of the secret and terminates the protocol. Otherwise, he checks whether $g_i^{y_i^{(j)}} \equiv E(y^{(j)}) \bmod q_i$ and he checks the correctness of $\mathsf{RngPrf}(E(y^j), M)$ for $1 \leq j \neq i \leq t^*$. If all the checks are successful, then P_i computes $d_i = \sum_{l=1}^{t^*} y_i^{(l)} \bmod p_i$. Otherwise, he outputs a random guess of the secret and stops the protocol.

   Let $d := d^{(1)} + \cdots + d^{(t^*)}$. Note that $d_1, \ldots, d_{t^*}$ are the shares for $d \bmod p_0$.

3. Each P_i computes $\widetilde{y_i} := (y_i + d_i) \bmod p_i$ as his "one-time" share for the current iteration. The commitment for $\widetilde{y_i}$ is $E(\widetilde{y_i}) := E(y) \prod_{l=1}^{t^*} E(y^{(l)}) \bmod q_i$, which can be locally computed by each player.

Proposition 7.2.1. *Let $Y := y + y^{(1)} + \cdots + y^{(t^*)} = (s+d) + (A_0 + \cdots + A_{t^*}) \cdot p_0$. After the share update phase, $\{\widetilde{y_1}, \ldots, \widetilde{y_{t^*}}\}$ are valid shares for $(s+d)$ mod p_0 as long as all the players follow the protocol honestly. In addition, all the commitments are correctly checked.*

Proof. First we prove the correctness of our scheme, namely, any t players can recover the correct secret. It is easy to see that $Y < \prod_{i=1}^{t} p_i$, since $y < M$, each $y^{(i)} < M, m \leq n$ and $M = \left\lfloor \frac{\prod_{i=1}^{t} p_i}{n+1} \right\rfloor$. (Hence the motivation of setting the parameter $M = \left\lfloor \frac{\prod_{i=1}^{t} p_i}{n+1} \right\rfloor$ is to guarantee that $Y < \prod_{i=1}^{t} p_i$, which is necessary for the correctness of the Asmuth-Bloom's secret sharing scheme on which our secret sharing scheme is based). Now, since $y_i = y \bmod p_i$ and $y_i^{(l)} = y^{(l)} \bmod p_i$, for each

$1 \leq i, l \leq t^*$, it follows that

$$\begin{aligned}\widetilde{y_i} &= (y_i + d_i) \bmod p_i \\ &= (y_i + \sum_{l=1}^{t^*} y_i^{(l)}) \bmod p_i \\ &= Y \bmod p_i\end{aligned}$$

In other words, each $\widetilde{y_i}$ satisfies $\widetilde{y_i} = Y \bmod p_i$ and $Y < \prod_{j=1}^{t} p_j$. Hence, Y and consequently $(s+d) \bmod p_0$ can be reconstructed from t shares by the Chinese Remainder Theorem (or by executing the combine phase). With respect to the correctness of the verification data, it suffices to observe that

$$\begin{aligned}E(y)\prod_{l=1}^{t^*} E(y^{(l)}) \bmod q_i &= g^y \prod_{l=1}^{t^*} g^{y^{(l)}} \bmod QN \bmod q_i \\ &= g_i^y \prod_{l=1}^{t^*} g_i^{y^{(l)}} \bmod q_i \\ &= g_i^{y_i} \prod_{l=1}^{t^*} g_i^{y_i^{(l)}} \bmod q_i \\ &= g_i^{(y_i + \sum_{l=1}^{t^*} y_i^{(l)})} \bmod q_i \\ &= g_i^{\widetilde{y_i}} \bmod q_i\end{aligned}$$

Now we are proceeding to prove that any coalition C of $t-1$ players almost learns no information about $(s+d) \bmod p_0$, that is, when the players in C share all the information they collect during the share update phase, every candidate for $(s+d) \bmod p_0$ is approximately equally likely. Remember that all the "one-time" shares are obtained when each player P_i shares his random-chosen $d^{(i)}$. Let $M_C = \prod_{P_i \in C} p_i$, $y^{(0)} = y$ and $d^{(0)} = s$. It suffices to prove that for any $0 \leq i \leq t^*$, for any candidate $d'^{(i)} \in Z_{p_0}$, the probability of the event $d^{(i)} = d'^{(i)}$ is approximately equals $\frac{1}{p_0}$. Let $y'^{(i)}$ be the unique solution for $y^{(i)}$ in Z_{M_C}, that is $y'^{(i)} \equiv y_j^{(i)} \bmod p_j$ for all $P_j \in C$. Since $M = \left\lfloor \frac{\prod_{l=1}^{t} p_l}{n+1} \right\rfloor$ and $\prod_{l=1}^{t} p_l > (n+1)p_0^2 \prod_{l=1}^{t-1} p_{n-l+1}$, the value of the ratio

$$\frac{M}{M_C} > \frac{M}{\prod_{l=1}^{t-1} p_{n-l+1}} \approx \frac{\prod_{l=1}^{t} p_l}{(n+1)\prod_{l=1}^{t-1} p_{n-l+1}}$$

is greater than p_0^2. Hence $y'^{(i)}+jM_C$ is smaller than M for all $j < p_0$. Besides, since $\gcd(p_0, M_C) = 1$, all the $y'^{(i)} + jM_C \bmod p_0$ are pairwise distinct for $0 \leq j < p_0$, and so cover all the integers between 0 and $p_0 - 1$. For each value of $d'^{(i)}$, there are either $\left\lfloor \frac{M}{p_0 M_C} \right\rfloor$ or $\left\lfloor \frac{M}{p_0 M_C} \right\rfloor + 1$ possible candidate $y^{(i)}$ consistent with $d'^{(i)}$ (that is, there are either $\left\lfloor \frac{M}{p_0 M_C} \right\rfloor$ or $\left\lfloor \frac{M}{p_0 M_C} \right\rfloor + 1$ many j's such that $y'^{(i)} + jM_C \equiv d'^{(i)} \bmod p_0$ and $y'^{(i)} + jM_C < M$), depending on the value of $d'^{(i)}$. In other words, from the perspective of players in C, for any two different integers u, v in Z_{p_0}, the probabilities $\Pr[d^{(i)} = u]$ and $\Pr[d^{(i)} = v]$ approximately equal. Since $\frac{M}{p_0 M_C} > p_0$, all the candidate for $d^{(i)}$ are approximately equally likely when p_0 is large enough.

Aside from those $t-1$ shares for $d^{(i)}$, the only additional information the coalition C can obtain are $E(y^{(i)})$, $E(h^{y^i} \bmod Q)$, and $\mathbf{RngPrf}(E(y^{(i)}), M)$. However, since the underlying commitment scheme is computationally hiding and the range-proof technique is statistically zero knowledge, the coalition learns no information about each $y^{(i)}$ (and so each $d^{(i)}$) in the computational sense. Hence the PPT coalition C knows nothing about $(s + d^{(1)} + \cdots + d^{(t^*)}) \bmod p_0 = (s + d) \bmod p_0$. □

Remark 7.2.2. This proposition means that every subset of at least t players uniquely determines $(s+d) \bmod p_0$ (Correctness), while for any subset of $t-1$ players, every candidate for s or for each $d^{(i)}$ is (approximately) equally likely, and consequently each candidate for each $(s + d) \bmod p_0$ is (approximately) equally likely (Privacy).

Proposition 7.2.3. [36] *During the share update phase, any player P_i can not distribute inconsistent shares for $d^{(i)}$ without being detected except with probability negligible in k. In other words, if all checks are successful, then each of the shares $y_1^{(i)}, \ldots, y_{t^*}^{(i)}$ is a residue of some integer less than M except with negligible probability which is introduced by the error probability of* RngPrf.

Proof. Suppose that P_i distributes inconsistent shares, that is, there exist two subsets V and U with $|U|, |V| = t$ such that,

$$(\sum_{j\in U} y_j^{(i)} M'_{U,j} M_{U\setminus\{j\}}) \bmod M_U \neq (\sum_{j\in V} y_j^{(i)} M'_{V,j} M_{V\setminus\{j\}}) \bmod M_V,$$

where $M_{U\setminus\{j\}} = \prod_{l\in U-\{j\}} p_l$ and $M'_{U,j}$ is the inverse of $M_{U\setminus\{j\}}$ in $Z^*_{p_j}$. Similarly, $M_{V\setminus\{j\}} = \prod_{l\in V-\{j\}} p_l$ and $M'_{V,j}$ is the inverse of $M_{V\setminus\{j\}}$ in $Z^*_{p_j}$. Hence,

$$y^{(i)} = [(\sum_{j=1}^{t} y_j^{(i)} M'_{[t],j} M_{[t]\setminus\{j\}}) \bmod M_{[t]}] > M,$$

where $[t] := \{1, \ldots, t\}$. Therefore P_i can not provide a valid range proof $\mathrm{RngPrf}(E(y^{(i)}, M)$ except with negligible probability. On the other hand, if P_i tries to use a different $y'^{(i)} \neq y^{(i)}$ in the commitment $E(y'^{(i)})$ and generates a valid proof $\mathrm{RngPrf}(E(y'^{(i)}), M)$, then $E(y'^{(i)}) \bmod q_j \neq g_j^{y_j^{(i)}} \bmod q_j$ for some j. □

Remark 7.2.4. Let $T = y^{(1)} + \cdots + y^{(t^*)}$. Since the "one-time" shares $\widetilde{y_1}, \ldots, \widetilde{y_{t^*}}$ are the shares for $(s + d) \bmod p_0$, they are the shares for s if and only if $d \equiv 0 \bmod p_0$. This is equivalent to $T \equiv 0 \bmod p_0$. In this sense, the iteration in which $T \equiv 0 \bmod p_0$ is called the *valid* iteration. It is called an *invalid* iteration otherwise.

Remark 7.2.5. The goals of the Share Update Phase are twofold. On one hand, it makes our protocol proceed with several iterations: all except the last one are invalid iterations, which are designed to catch possible cheaters. During the valid iteration, all active players get the real secret. In addition, no one will know in advance whether the current iteration is going to be the last iteration. On the other hand, since during each iteration all the "one-time" shares are revealed, if the current round is invalid, the players should proceed to the next round with totally new shares, which are provided by the share update phase. Hence, "one-time" shares are shares used only once (that is, used only in the current iteration) and they become obsolete in later iterations.

7.2.2 *Combiner Phase*

In this phase, each player P_i uses the reconstruction algorithm from the the Asmuth-Bloom Secret Sharing Scheme to recover $(s + d) \bmod p_0$.

Combiner Phase

1. Let U be a collection of t shares that player P_i chooses in the reconstruction phase and let V be the corresponding collection of the indices of the players to whom those t shares belong. Let M_V denote $\prod_{j \in V} p_j$.

2. Let $M_{V \setminus \{j\}}$ denote $\prod_{\ell \in V, \ell \neq j} p_\ell$ and let $M'_{V,j}$ be the multiplicative inverse of $M_{V \setminus \{j\}}$ in $\mathbb{Z}^*_{p_j}$. Player P_i computes $Y^{(i)} := \sum_{j \in V} \widetilde{y_j} \cdot M'_{V,j} \cdot M_{V \setminus \{j\}} \bmod M_V$. Finally, let $S^{(i)} := Y^{(i)} \bmod p_0$.

7.2.3 Overview of the Reconstruction Phase.

In order for the reader to get an easier understanding of the reconstruction phase, we first give its general view. The full description is in Sect. 7.2.4.

The reconstruction phase proceeds with a sequence of *invalid/valid* iterations such that the last iteration is valid and each iteration has two stages. During the first stage, players first interact to get their "one-time" shares for $(s+d) \bmod p_0$, where $d = d^{(1)} + \cdots + d^{(t^*)}$ and each $d^{(i)}$ is chosen randomly by P_i in Stage 1. During Stage 2, each player P_i is required to open the value $h^{y^{(i)}} \bmod Q$, where y^i is chosen by P_i in Stage 1. Thus, the players can jointly identify the status of the current iteration since $d^{(i)} = y^{(i)} \bmod p_0$: if $d \bmod p_0 \neq 0$, that is, if $(\prod_{j=1}^{m}(h^{y^{(j)}} \bmod Q)) \bmod Q$ does not appear in the sorted list published by the dealer, then the current iteration is invalid and all the players are asked to restart a new iteration; otherwise, it is valid, the secret s is recovered and the protocol terminates immediately after this iteration.

The iterations have the following properties:

- invalid iteration: no information about s is revealed since all the revealed shares are the shares for $(s+d) \bmod p_0$, a random value other than s. At the beginning of the subsequent iteration, all the shares are updated, which guarantees that the "one-time"'shares revealed in the current iteration are useless for the next iteration.

- valid iteration: every player recovers s provided that every participant follows the protocol.

The key in this process is the fact that nobody knows before the opening of the "one-time" shares whether the current iteration will be valid. Furthermore, when a given player realizes that the valid iteration occurs, each other player can compute the secret as well. That is why we do not need simultaneous channels.

7.2.4 Secret Reconstruction Phase

Our reconstruction protocol proceeds with multiple iterations, each of which contains two stages for each of these t^* players. It is assumed without lost of generality that in each step, each P_i executes his strategy in index increasing order. For each of these t^* participants P_i, his strategy σ_i is as follows.

Secret Reconstruction Phase

Stage 1.

1. Player P_i executes the share update phase to get his "one-time" share $\widetilde{y_i}$ for the value $(s + d) \bmod p_0$, where $d = d^{(1)} + \cdots + d^{(t^*)}$ and each $d^{(i)}$ is chosen independently and uniformly at random by P_i .
2. Player P_i broadcasts his "one-time" share $\widetilde{y_i}$ obtained at the previous step. If P_i does not receive t^* shares (including his own), or if he detects that $g_j^{\widetilde{y_j}} \bmod q_j \neq E(\widetilde{y_j}) \bmod q_j$ for some j, he outputs a random guess of the secret and aborts the protocol abruptly.
3. Otherwise, P_i chooses randomly t data from $\{\widetilde{y_1}, \ldots, \widetilde{y_{t^*}}\}$ and executes the Combiner Phase.

The second stage is used to identify the status (valid/invalid) of the current round since $\{\widetilde{y_1}, \cdots, \widetilde{y_{t^*}}\}$ are the shares for s if and only if $h^{y^{(1)}+\cdots+y^{(t^*)}} \bmod Q$ appears in the sorted list published by the dealer.

Stage 2.

1. Player P_i broadcasts $h^{y^{(i)}} \bmod Q$. If he does not receive t^* messages (including his own), or if he detects that $g^{(h^{y^{(j)}} \bmod Q)} \bmod QN \neq E(h^{y^{(j)}} \bmod Q)$ for some j, P_i outputs $S^{(i)}$ he obtains in the Combiner Phase and aborts the whole protocol.
2. Otherwise, P_i checks by binary search whether $(\prod_{j=1}^{m}(h^{y^{(j)}} \bmod Q)) \bmod Q = h^{y^{(1)}+\cdots+y^{(t^*)}} \bmod Q$ appears in the sorted list published by the dealer. If yes, he outputs $S^{(i)}$ and stops the whole protocol; Otherwise, P_i goes back to Stage 1 and starts another iteration.

7.3 Security of our Rational Secret Sharing Scheme

Before we demonstrate the security proofs, we need to give several definitions of some variants of Nash equilibrium that our protocol is supposed to achieve. In the following, each player is regarded as a polynomial-time probabilistic Turning (PPT) machine and each utility function is considered as a polynomial of the security parameter k .

First we define what it means for two strategies to be equivalent by recalling the corresponding definitions in [27]. We assume that t^* players $P_1, \ldots, P_{t^*}$ are involved in the game. Let C be subset of $\{1, \ldots, t^*\}$. As often carried out in multiparty computation, security will be demonstrated by simulating the views of the different participants.

Definition 7.3.1. [27] Let $P_C := \{P_i : i \in C\}$, $P_{-C} := \{P_i : i \in \{1, \dots, t^*\} \setminus C\}$ and the strategy vector of the t^* players be $\sigma = (\sigma_1, \dots, \sigma_{t^*})$. Define the random variable $\mathsf{View}^{\sigma}_{-C}$ as follows:

> Let Trans denote the messages sent by P_C *not including any message sent by P_C after they write to their output tapes.* $\mathsf{View}^{\sigma}_{-C}$ includes the information given by the dealer to P_{-C}, the random coins of P_{-C} and the (partial) transcript Trans.

Fix a strategy ρ_C and an algorithm T. Define the random variable $\mathsf{View}^{T,\rho_C}_{-C}$ as follows:

> When the t^* players interact, P_C follows ρ_C and P_{-C} follows σ_{-C}. Let Trans denote the messages sent by P_C. Algorithm T, given the entire view of P_C, outputs an arbitrary truncation Trans' of Trans (defining a cut-off point and deleting any messages sent after that point). $\mathsf{View}^{T,\rho_C}_{-C}$ includes the information given by the dealer to P_{-C}, the random coins of P_{-C}, and the (partial) transcript Trans'.

Strategy ρ_C *yields an equivalent play with respect to* σ, denoted $\rho_C \approx \sigma$, if there exists a PPT algorithm T such that for all PPT distinguishers D:

$$\left|\Pr[D(1^k, \mathsf{View}^{T,\rho_C}_{-C}) = 1] - \Pr[D(1^k, \mathsf{View}^{\sigma}_{-C}) = 1]\right| \leq \epsilon(k)$$

where $\epsilon(\cdot)$ is a negligible function.

Definition 7.3.2. [27] A strategy σ is said to be an *r-resilient computational strict Nash equilibrium*, if:

1. σ induces an r-resilient computational Nash equilibrium.

2. For any coalition C of size at most r and for any probabilistic polynomial time strategy σ'_C with $\sigma'_C \not\approx \sigma$, there is a positive polynomial $p(\cdot)$ such that for any $i \in C$, it holds that $U_i(k, \sigma_C, \sigma_{-C}) \geq U_i(k, \sigma'_C, \sigma_{-C}) + \frac{1}{p(k)}$ for infinitely many values of k, namely, $U_i(k, \sigma_C, \sigma_{-C}) - U_i(k, \sigma'_C, \sigma_{-C})$ is non-negligible.

Definition 7.3.3. [27] For any coalition C, strategy ρ_C is *δ-close* to strategy σ_C if ρ_C is as follows:

ρ_C: With probability $1-\delta$, players in C play according to σ_C.
With probability δ, players in C follow an arbitrary (possibly correlated)
PPT strategy σ'_C (called the **residual strategy** of ρ_C).

Definition 7.3.4. [27] σ induces an *r-resilient computational Nash equilibrium that is stable with respect to trembles* if:

1. σ induces an r-resilient computational Nash equilibrium;

2. There is a noticeable function δ such that for any coalition C with size at most r, and any vector of PPT strategies ρ_{-C} that is δ-close to σ_{-C}, any PPT strategy ρ_C, there exists a PPT strategy $\sigma'_C \approx \sigma$ such that $U_i(k, \rho_C, \rho_{-C}) \leq U_i(k, \sigma'_C, \rho_{-C}) + \epsilon(k)$, where $\epsilon(\cdot)$ is negligible.

Remark 7.3.5. Intuitively, the strategy profile (σ_C, σ_{-C}) is stable with respect to trembles if σ_C remains a best response even if P_{-C} plays any PPT strategies other than σ_{-C} with some small but noticeable probability δ.

Let $U_i^r(k) := \frac{1}{|S|} \cdot U_i^+(k) + (1 - \frac{1}{|S|}) \cdot U_i^-(k)$, which is player P_i's expected utility obtained by outputting a random guess for the secret (assuming that the other players abort without any outputs, or with wrong outputs). Let $U_i^*(k) := \frac{1}{p_0} \cdot U_i^+(k) + (1 - \frac{1}{p_0}) \cdot U_i^r(k)$, $1 \leq i \leq n$. Based on the security requirements of [13], we make the following assumption:

Λ: The discrete logarithm problem over finite fields is intractable.
The RSA modulus N is hard to factor; the resulting RSA encryption
scheme and Schnorr signature is secure.

Now we are well prepared to demonstrate the security proofs.

Theorem 7.3.6. *Assuming that Λ holds, σ induces a $(t-1)$-resilient computational Nash equilibrium as long as $U_i(k) - U_i^*(k)$ is non-negligible, for $1 \leq i \leq n$.*

Proof. By Proposition 7.2.1, our protocol is a valid secret sharing scheme. All the active players will be expected to recover the real secret in $\frac{1}{\Pr[d \equiv 0 \bmod p_0]} = p_0$ iterations, as long as they stick to σ.

Now we prove that σ induces a $(t-1)$-resilient computational Nash equilibrium. Let C be any coalition of size at most $t-1$. Assume that all the players not in C stick to their prescribed strategies. We focus on PPT deviations of players in C. There are several possible cases: (1) some player P_i in C deviates during the share update phase; (2) some player P_i in C lies about his "one-time" share or only sends partial messages in Stage 1 - Step 2; (3) some player P_i in C either opens a fake $h^{y^{(i)}}$ mod Q or broadcasts nothing in Stage 2.

Suppose that (1) happens. There are two possible deviations. **Case 1.** P_i only sends (or broadcasts) partial messages in Stage 2 of the share update phase. However, this will be detected and cause the protocol to terminate. In this case, the only profitable thing he can do is to output a random guess of the secret, which will earn him at most $U_i^r(k)$. Obviously, it is a worse outcome to P_i, since $U_i^r(k) < U_i(k)$. Hence, P_i will send all data, fake or real, as required. **Case 2.** P_i distributes inconsistent shares for his randomly chosen $d^{(i)}$ to some player P_j not in C. Under the assumption Λ, no cheating P_i can convince any other P_j to accept $\mathsf{RngPrf}(E(y^{(i)}, M)$ except with negligible probability $\epsilon'(k)$. Once $\mathsf{RngPrf}(E(y^{(i)}, M)$ is rejected, which happens with probability $1 - \epsilon'(k)$, the protocol terminates immediately and the best that player P_i can do is to output a random guess of the secret. Thus, the expected utility P_i can get by distributing inconsistent shares is at most $\epsilon'(k) \cdot U_i^+(k) + (1 - \epsilon'(k)) \cdot U_i^r(k) = \epsilon'(k) \cdot (U_i^+(k) - U_i^r(k)) + U_i^r(k) < \epsilon(k) + U_i(k)$, where $\epsilon(k) = \epsilon'(k)\,(U_i^+(k) - U_i^r(k))$ is a negligible function in k, since we assume that $U_1, \ldots, U_n$ are polynomials in k. In other words, using this type of deviation, P_i can only increase his payoff by a negligible amount (if at all). Thus, given our computational setting, no rational player P_i is to deviate by distributing inconsistent shares.

Now we consider the possible deviations in Step 2 of Stage 1. There are two possible cases. **Case 1.** P_i does not broadcast anything at all. **Case 2.** P_i cheats about his "one-time" share. However, either of these deviations will be detected and cause the protocol to terminate. Hence, we do not distinguish between these two cases. If (d mod p_0) $= 0$ (i.e., the current iteration is valid), which happens with probability $\frac{1}{p_0}$, then all the players in C will output the real secret and hence P_i will get at most $U_i^+(k)$. If (d mod p_0) $\neq 0$ (i.e., the current iteration is invalid), which happens with probability $1 - \frac{1}{p_0}$, then the best thing P_i can do is to output a random guess of the secret earning at most $U_i^r(k)$. Thus, the expected payoff of P_i with this type of deviation is at most $\frac{1}{p_0} \cdot U_i^+(k) + (1 - \frac{1}{p_0}) \cdot U_i^r(k) = U_i^*(k)$. It is less than $U_i(k)$ by our assumption. Hence, as a rational player, P_i will not deviate in Step 2 of Stage 1.

Finally, we study what happens if some player in C does not broadcast anything at all or broadcasts a fake value in Stage 2. Either deviation will be detected and cause the protocol to terminate abruptly. Since we assume that players execute every step of the protocol in ascending order, we can assume without loss of generality that $C = \{P_{t^*-t+2}, \ldots, P_{t^*}\}$. Since all the players in C share their information, for any $t^*-t+2 \leq i \leq t^*$, after receiving the message from the players not in C, P_i can first check whether $h^{y^{(1)}+\cdots+y^{(t^*)}} \bmod Q$ appears in the sorted list published by the dealer (from which he knows nothing about d except whether $d \bmod p_0 = 0$) to identify whether the current round is valid or not, then determines what to do in this stage. Note that we have proved that in the computational and rational setting, any player will execute the reconstruction phase honestly up to the end of Stage 1. Therefore, if the current iteration is valid, each $S^{(j)}$ obtained by P_j in the Combiner Phase is indeed the real secret. In this case, regardless of what P_i will do, each player will output the real secret, which will earn $U_i(k)$ to P_i. On the other hand, if the current round is invalid, no one has recovered the real secret yet and either type of deviations will cause the protocol to terminate abruptly resulting in a payoff at most $U_i^r(k)$ to P_i. Hence, P_i is never better off by this deviations. □

Theorem 7.3.7. *Assuming that Λ holds, σ induces a $(t-1)$-resilient computational strict Nash equilibrium provided that $U_i(k) - U_i^*(k)$ is non-negligible, for $1 \leq i \leq n$.*

Proof. Suppose that C is any subset of $\{1, \ldots, t^*\}$ of size at most $t-1$. Let $P_C := \{P_i | i \in C\}$ and $P_{-C} := \{P_i | i \in \{1, \ldots, t^*\} \setminus C\}$. Since all the players in P_C act in unison, we can regard P_C as a whole. By Theorem 7.3.6, it is sufficient to prove that for any PPT strategy $\rho_C \not\approx \sigma$, there is a positive polynomial $p(\cdot)$ such that for any $i \in C$, $U_i(k, \sigma) \geq U_i(k, \rho_C, \sigma_{-C}) + \frac{1}{p(k)}$ for infinitely many values of k, that is, $U_i(k, \sigma) - U_i(k, \rho_C, \sigma_{-C})$ is positive and non-negligible.

Let $\mathsf{Deviate}$ be the event that P_C deviates from σ_C before he can compute his output, that is, before entering the Stage 2 of the valid iteration. Since $\rho_C \not\approx \sigma$, $\Pr[\mathsf{Deviate}]$ is non-negligible by definition. Now, consider the interaction of ρ_C with σ_{-C}. Let Valid be the event that P_C deviates from σ_C before entering Stage 2 during the valid iteration and let $\mathsf{Invalid}$ be the event that P_C deviates from σ_C during an invalid iteration. Let Caught be the event that P_C is caught cheating. Then, for each $i \in C$, we have:

$$\begin{aligned} & U_i(k, \rho_C, \sigma_{-C}) \\ \leq\ & U_i^+(k) \cdot \Pr[\mathsf{Valid}] + U_i^+(k) \cdot \Pr[\mathsf{Invalid} \wedge \overline{\mathsf{Caught}}] \\ & + U_i^r(k) \cdot \Pr[\mathsf{Invalid} \wedge \mathsf{Caught}] + U_i(k) \cdot \Pr[\overline{\mathsf{Deviate}}] \end{aligned}$$

$$\begin{aligned}
&= U_i^+(k) \cdot (\Pr[\mathsf{Valid}|\mathsf{Deviate}] + \Pr[\overline{\mathsf{Caught}}|\mathsf{Invalid}] \cdot \Pr[\mathsf{Invalid}|\mathsf{Deviate}]) \cdot \Pr[\mathsf{Deviate}] \\
&\quad + U_i^r(k) \cdot \Pr[\mathsf{Caught}|\mathsf{Invalid}] \cdot \Pr[\mathsf{Invalid}|\mathsf{Deviate}] \cdot \Pr[\mathsf{Deviate}] + (1 - \Pr[\mathsf{Deviate}])U_i(k) \\
&= U_i^+(k) \cdot \left[\frac{1}{p_0} + \epsilon(k)(1 - \frac{1}{p_0})\right] \cdot \Pr[\mathsf{Deviate}] \\
&\quad + U_i^r(k) \cdot (1 - \epsilon(k)) \cdot (1 - \frac{1}{p_0}) \cdot \Pr[\mathsf{Deviate}] + U_i(k) - U_i(k) \cdot \Pr[\mathsf{Deviate}] \\
&= U_i(k) + (U_i^*(k) - U_i(k)) \cdot \Pr[\mathsf{Deviate}] + \eta(k)
\end{aligned}$$

where $\eta(k) = \epsilon(k) \cdot (1 - \frac{1}{p_0}) \cdot (U_i^+(k) - U_i^r(k)) \cdot \Pr[\mathsf{Deviate}]$ is negligible. It follows that

$$U_i(k, \sigma) = U_i(k) \geq U_i(k, \rho_C, \sigma_{-C}) + (U_i(k) - U_i^*(k)) \cdot \Pr[\mathsf{Deviate}] - \eta(k).$$

Since both $U_i(k) - U_i^*(k)$ and $\Pr[\mathsf{Deviate}]$ are positive and non-negligible, $U_i(k, \sigma) - U_i(k, \rho_C, \sigma_{-C})$ is positive and non-negligible, which completes this proof. □

Remark 7.3.8. In the proof for Theorem 7.3.7, it is shown that for any PPT strategy ρ_C, we have:

$$U_i(k, \sigma) = U_i(k) \geq U_i(k, \rho_C, \sigma_{-C}) + (U_i(k) - U_i^*(k)) \cdot \Pr[\mathsf{Deviate}] - \eta(k),$$

where $\eta(\cdot)$ is a negligible function.

Theorem 7.3.9. *Assuming that Λ holds, σ induces a computational Nash equilibrium that is stable with respect to trembles provided that $U_i(k) - U_i^*(k)$ is non-negligible, for $1 \leq i \leq n$.*

Proof. This proof is based on [27]. Let δ be a parameter which we will specify at the end of the proof. Note that δ may depend on k. Since we assume that players execute every step of the protocol in an index increasing order, we can assume without loss of generality that $C = \{t^* - t + 2, \ldots, t^*\}$. It is sufficient to show that for any $i \in C$, for any vector of PPT strategies ρ_{-C} that is δ-close to σ_{-C}, and any PPT strategy ρ_C, there exists a PPT strategy $\sigma'_C \approx \sigma$ such that $U_i(k, \rho_C, \rho_{-C}) \leq U_i(k, \sigma'_C, \rho_{-C}) + \epsilon(k)$, where $\epsilon(\cdot)$ is negligible. As before, let $P_C = \{P_i | i \in C\}$ and $P_{-C} = \{P_i | i \in (\{1, \ldots, t^*\} \setminus C)\}$. First, we construct a strategy σ'_C for the players in P_C as follows.

1. Set Detect:=0.
2. In each iteration:
    (a) Receive the messages from P_{-C} in each possible step. If P_C detects that some player P_j in P_{-C} has deviated from σ_j, set Detect:= 1.
    (b) If Detect= 1, execute the remaining steps according to ρ_C; otherwise σ_C.
3. If Detect= 0, determine the output according to σ_C, otherwise, output whatever ρ_C outputs.

Observe that when σ'_C interacts with σ_{-C}, Detect is never set to be 1. Hence $\sigma'_C \approx \sigma$ and $U_i(k, \sigma'_C, \sigma_{-C}) = U_i(k, \sigma_C, \sigma_{-C}) = U_i(k)$ for any $i \in C$. Now, we want to show that $U_i(k, \rho_C, \rho_{-C}) \leq U_i(k, \sigma'_C, \rho_{-C}) + \eta(k)$ for any $i \in C$, where $\eta(\cdot)$ is negligible. Let $\widetilde{\rho_{-C}}$ denote the residual strategy of ρ_{-C}. In an interaction where P_C follows strategy ρ_C, let Detected be the event that P_C is detected deviating from σ_C before entering stage 2 of the valid iteration while no player in P_{-C} is detected cheating so far. Also, let $\Pr_{\mathsf{Detected}}(\alpha)$ be the probability of Detected when P_{-C} follows strategy α. Since no player in P_{-C} will be detected cheating when P_{-C} execute σ_{-C}, $\Pr_{\mathsf{Detect}ed}(\sigma_{-C})$ equals the probability of P_C being detected deviating from σ_C before entering Stage 2 of the valid iteration.

Claim 1. $\Pr[\mathsf{Deviate}]=\Pr_{\mathsf{Detected}}(\sigma_{-C}) + \epsilon(k) \cdot \Pr[\mathsf{Deviate}]$, for some negligible function ϵ.

Proof. Since the players in C will never be detected deviating from σ_C if they follow σ_C, and any deviation from σ_C can be detected except with negligible probability $\epsilon(k)$,

$$\begin{aligned}
\Pr_{\text{detected}}(\sigma_{-C}) &= \Pr[\textbf{deviate} \wedge \textbf{caught}] \\
&= \Pr[\textbf{caught}|\textbf{deviate}] \cdot \Pr[\textbf{deviate}] \\
&= (1-\epsilon(k)) \cdot \Pr[\textbf{deviate}].
\end{aligned}$$

□

Claim 2. For any $i \in C$,

$$U_i(k, \rho_C, \widetilde{\rho_{-C}}) - U_i(k, \sigma'_C, \widetilde{\rho_{-C}}) \leq \Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) \cdot (U_i^+(k) - U_i^r(k)) + \epsilon(k),$$

where $\epsilon(\cdot)$ is negligible.

Proof. Consider the interaction of σ'_C with $\widetilde{\rho - C}$. If no player in P_{-C} is detected cheating, then $\sigma'_C = \rho_C$. In this case, $U_i(k, \rho_C, \widetilde{\rho_{-C}}) - U_i(k, \sigma'_C, \widetilde{\rho_{-C}}) = 0$. However, if some player in P_C is detected cheating, then **detected** is well defined in the interaction of σ'_C with $\widetilde{\rho_{-C}}$, since σ'_C runs a copy of ρ_C as a sub-routine. When **detected** does not happen, there are two possibilities:

- Neither the players in P_C nor the players in P_{-C} has being detected cheating before entering stage 2 of the valid iteration. In this case, $\sigma'_C = \rho_C = \sigma_C$ except with negligible probability. Hence, the output of each player is unchanged whether P_C is running σ'_C or ρ_C except with negligible probability $\epsilon'(k)$. In this case, $U_i(k, \rho_C, \widetilde{\rho_{-C}}) - U_i(k, \sigma'_C, \widetilde{\rho_{-C}}) \leq \epsilon'(k) \cdot (U_i^+(k) - U_i^r(k))$.

- Some player in P_{-C}, say P_j, is the first one to be detected *cheating before entering in stage 2 of the valid iteration.* However, this implies that before P_j is detected cheating, P_C actually executes σ_C except with negligible probability. Since σ'_C "switches" to ρ_C immediately after P_j is detected cheating, the outputs of each player are identical whether P_C runs σ'_C or ρ_C except with negligible probability $\epsilon''(k)$. Thus $U_i(k, \rho_C, \widetilde{\rho_{-C}}) - U_i(k, \sigma'_C, \widetilde{\rho_{-C}}) \leq \epsilon''(k) \cdot (U_i^+(k) - U_i^r(k))$.

When **detected** happens, the maximum difference in the utilities is $(U_i^+(k) - U_i^r(k))$. It completes the proof of the claim since all the utilities are polynomial in k.

□

Claim 3. $\Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) \leq \Pr_{\mathsf{Detected}}(\sigma_{-C}) + \epsilon(k)$ for some $\epsilon(\cdot)$ negligible.

Proof. Suppose that **detected** happens when interacting with $\widetilde{\rho_{-C}}$. Consider all the view of P_C obtained before **detected** happens. Since any deviation from σ_C will be detected except with negligible probability, the entire view is the same as the one obtained before P_C deviates from σ_C except with negligible probability. On the other hand, since P_C is detected cheating first and in every communication round, any deviation (from σ_{-C}) will be detected except with negligible probability, the initial segment of $\widetilde{\rho_{-C}}$(containing all the steps before P_C deviates from σ_C) coincides with the initial segment of σ_{-C} except with negligible probability. Therefore, except with negligible probability $\epsilon(k)$, P_C will also deviate (from σ_C) and consequently be detected deviating when interacting with σ_{-C}, where nobody in P_{-C} will deviate. That is, $\Pr_{\mathbf{detected}}(\sigma_{-C}) + \epsilon(k) \geq \Pr_{\mathbf{detected}}(\widetilde{\rho_{-C}})$, for some negligible function $\epsilon(\cdot)$. □

By Remark 7.3.8, we know that for any PPT strategy ρ_C,

$$U_i(k, \sigma) = U_i(k) \geq U_i(k, \rho_C, \sigma_{-C}) + (U_i(k) - U_i^*(k)) \cdot \Pr[\mathsf{Deviate}] - \eta(k),$$

where $\eta(\cdot)$ is a negligible function. Now, we get:

$$\begin{aligned} U_i(k, \rho_C, \rho_{-C}) &= (1-\delta) \cdot U_i(k, \rho_C, \sigma_{-C}) + \delta \cdot U_i(k, \rho_C, \widetilde{\rho_{-C}}) \\ &\leq (1-\delta) \cdot [U_i(k) + (U_i^*(k) - U_i(k)) \cdot \Pr[\mathsf{Deviate}] + \eta(k)] \\ &\quad +\delta \cdot U_i(k, \rho_C, \widetilde{\rho_{-C}}) \end{aligned}$$

Besides,

$$\begin{aligned} U_i(k, \sigma_C', \rho_{-C}) &= (1-\delta) \cdot U_i(k, \sigma_C', \sigma_{-C}) + \delta \cdot U_i(k, \sigma_C', \widetilde{\rho_{-C}}) \\ &= (1-\delta) \cdot U_i(k) + \delta \cdot U_i(k, \sigma_C', \widetilde{\rho_{-C}}) \end{aligned}$$

It follows that

$$\begin{aligned} & U_i(k, \rho_C, \rho_{-C}) - U_i(k, \sigma_C', \rho_{-C}) \\ \leq\ & (1-\delta) \cdot (U_i^*(k) - U_i(k)) \cdot \Pr[\mathsf{Deviate}] + \delta \cdot [U_i(k, \rho_i, \widetilde{\rho_{-C}}) - U_i(k, \sigma_C', \widetilde{\rho_{-C}}] + \eta(k) \\ \overset{\text{by Claim 2}}{\leq}\ & (1-\delta) \cdot (U_i^*(k) - U_i(k)) \cdot \Pr[\mathsf{Deviate}] \\ & +\delta \cdot \Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) \cdot (U_i^+(k) - U_i^r(k)) + \delta \cdot \epsilon(k) + \eta(k) \\ \overset{\text{by Claim 1}}{=}\ & (1-\delta) \cdot (U_i^*(k) - U_i(k)) \cdot (\Pr_{\mathsf{Detected}}(\sigma_{-C}) + \epsilon'(k) \cdot \Pr[\mathsf{Deviate}]) \\ & +\delta \cdot (U_i^+(k) - U_i^r(k)) \cdot \Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) + \delta \cdot \epsilon(k) + \eta(k) \\ \overset{\text{by Claim 3}}{\leq}\ & (1-\delta) \cdot (U_i^*(k) - U_i(k)) \cdot \Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) \\ & +\delta \cdot (U_i^+(k) - U_i^r(k)) \cdot \Pr_{\mathsf{Detected}}(\widetilde{\rho_{-C}}) + \eta'(k), \end{aligned}$$

where $\eta'(\cdot)$ is some negligible function. Hence, there exists $\delta > 0$ (may depend on k) such that the above expression is negligible in k for each $i \in C$. □

Efficiency. The expected number of iterations of our protocol is p_0. Note that the requirements for p_0 are that $p_0 > \frac{2[U_i^+(k) - U_i^r(k)]}{U_i(k) - U_i^r(k)}$, for $1 \leq i \leq n$. Since all the utility functions are polynomial in k and $U_i(k) - U_i^r(k)$ is assumed to be non-negligible, p_0 can be chosen to be a prime less than some polynomial in k. Since all the computations are based on modular arithmetic, they can be executed in polynomial time. Besides, RngPrf can also be verified in polynomial time. All these considerations imply that our protocol is efficient.

Comparison to Fuchsbauer *et al.*'s Scheme. The protocol from [27] provides good point of comparison to ours since both techniques have similar features:

- Both of them induce a $(t-1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles.
- Neither of them relies on simultaneous channels.
- Both of them assume that no player knows any auxiliary information about the secret s. This property has been proved to be inherent to the non-simultaneous channels model [2].
- Both protocols run in time polynomial in k (security parameter) and they have almost the same round complexity.

However, our protocol has smaller share size even when $(t-1)$-resilience to coalitions is required. Our shares are $O(k)$ bits long while those from [27] need $(n-t+1)\,(2n|s| + O(k))$ bits. The latter share length leads to practical efficiency issues when $n-t+1$ is large or when Fuchsbauer *et al.*'s technique is used as a building block within more general rational MPC protocols.

7.4 Conclusion

In this chapter, we have presented an efficient protocol for t-out-of-n rational secret sharing based on the Chinese Remainder Theorem in non-simultaneous channels. Our technique leads to a $(t-1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles while having much smaller share size than the protocol proposed by Fuchsbauer *et al.* [27]. Our protocol is efficient since on one hand the expected round complexity is a polynomial in k, on the other hand, all the computations can be executed in polynomial time in k.

8. TRANSFORMATIONS FROM ANY LINEAR SECRET SHARING SCHEME TO A RATIONAL SECRET SHARING SCHEME

In this chapter, we propose three transformations from any classical linear secret sharing (LSSS for short) to a rational secret sharing scheme (RSSS for short) with a mediator. The RSSS obtained induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies, relies on no cryptographic assumption, provides information theoretical security and hence is automatically immune against backward induction. The first transformation works for any LSSS with an arbitrary adversary structure $\mathcal{A}$ and the resulting RSSS has expected round complexity relying on participants' utilities. The second transformation works for any LSSS with a Q^2 adversary structure $\mathcal{A}$ and the resulting RSSS has expected round complexity $O(1)$. In both of the above cases, we assume that knowledge of participants' utilities is public. The third transformation works for any LSSS with a Q^3 adversary structure and requires no knowledge of participants' utilities. The RSSS obtained has expected round complexity $O(1)$. The first transformation requires pairwise secure channels along with a simultaneous broadcast channel, while the second and the last transformations only require pairwise secure channels and if the LSSS to be transformed is $\mathcal{A}$-error detectable, the resulting RSSS induces an $\mathcal{A}$-resilient strict Nash equilibrium.

To the best of our knowledge, this research marks the first instance that the problem on transformation from any linear secret sharing scheme to a rational secret sharing scheme is considered and the first time that rational secret sharing schemes for non-threshold access structures are constructed. As an independent interest, our constructions generalize and improve some of the results proposed by Abraham *et.al.* in [1], where they proposed RSSSs over t-out-of-n threshold access structures, which induce $(t-1)$-resilient Nash equilibrium surviving iterated elimination of weakly dominated strategies. However, their constructions make use of digital signatures, which makes them suspectable to backward induction. As a comparison, our constructions are unconditionally secure.

8.1 Motivation

It is well-known that for any given access structure Γ, there always exists an LSSS realizing it. Suppose that a dealer shares a secret s among n participants via some LSSS and wishes to ensure that every subset $A \in \Gamma$ can later recover the secret s while any subset not in Γ get no information about the secret s. The dealer is no longer available after the process of share distribution. If all the participants were honest, the dealer's goal could be trivially achieved. Unfortunately, it can be the case that some participants will choose to be dishonest if deviating from the protocol will provide them with some advantage. For example, if the secret is the key of a safe containing treasures, it is reasonable to assume that each participant firstly prefers to be the only one learning s and secondly prefers to learn s with as few other participants as possible. In this situation, some participants may form a coalition and cheat about their shares in the secret reconstruction process, which may prevent the non-coalition participants from learning the secret. As a result, the dealer's goal can not be achieved. To partially solve this problem, we attempt to transform the original LSSS to an RSSS that, when rationally played, yields the secret to all participants at a Nash equilibrium or one of its variants.

A useful way to realize this transformation is to involve a mediator, which is a trusted third party, and then design a strategy for each participant along with the mediator such that on one hand every participant has no motivation to deviate from his strategy provided that all the others stick to theirs, and on the other hand, if each participant sticks to his strategy, the secret will be revealed to all participants. The purpose of the mediator is to distribute round r shares for each active participant at the beginning of the r^{th} round, which makes the reconstruction phase proceed with several rounds. Following Forges [26], we may view a mediator as a communication device: each participant sends a message (input) and the mediator computes a function of all the messages ever obtained and sends each participant some information (output).

In this chapter, we use the notations u_i^+, u_i^-, and u_i which were first defined in Chapter 6. (Please refer to P 86.) Besides, we consider coalitions in $\mathcal{A}$ and assume for simplicity that during the process of share reconstruction phase, there is at most one coalition which contains a subset of active players and all the players in this coalition share all information they jointly have. Thus, all the players in some coalition are assumed to share a single output.

8.2 Transformations from an LSSS to an RSSS

In the following, without loss of generality, we assume that $s \neq 0$ and this assumption is known to all participants. For ease of exposition, we assume that the secret is equally likely to be any nonzero field element. For any given monotone access structure Γ over $\mathcal{P} = \{P_1, \ldots, P_n\}$, we fix an MSP $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ computing Γ, where $\mathbb{F} = \mathrm{GF}(q)$, M has size $m \times e$ and the columns of M are assumed to be linearly independent. Let $\mathcal{A}$ denotes the corresponding adversary structure. As in Chapter 3, in the following, without loss of generality, we always assume that ψ preserves order, i.e., $\psi(i) \leq \psi(j)$ whenever $i < j$, where it is assumed that $P_1 < \cdots < P_n$. Let $m_i = |\{j : \psi(j) = P_i\}|$, that is, m_i is the number of rows assigned to P_i, $1 \leq i \leq n$. Let $\lambda_i = m_1 + \cdots + m_i$ for $1 \leq i \leq n$ and $\lambda_0 = 0$. Then for any $1 \leq i \leq n$, $S_i^0 = (s^0_{\lambda_{i-1}+1}, \ldots, s^0_{\lambda_{i-1}+m_i})$ is the share for P_i. It is quite natural to regard $(S_1^0, \ldots, S_n^0)$ as an n-tuple that belongs to $\mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$. Let C be the linear block code generated by M^T as in Chapter 3.

In this section, we are dedicated to transforming $\mathcal{M}$ to an RSSS that, when rationally played, opens the secret to all the participants in an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies. Note that in a rational model, each participant is self-interested, i.e., he does nothing other than maximizing his utility, which solely depends on who outputs the secret and who does not. The initial share distribution phase is given by $\mathcal{M}$. While in the reconstruction phase of our protocol, we view the interactions among the subset of active participants as a game augmented with a mediator. Let $\sigma = (\sigma_1, \ldots, \sigma_n)$ denote the recommended joint strategy of participants, where σ_i is participant P_i's recommended strategy, $1 \leq i \leq n$. The reconstruction phase proceeds with several rounds, among which only the last round is the valid round, since only in the last round the shares are distributed for the secret s while in any other round the shares are distributed for 0. And the key point is that no participant can tell in advance which round is going to be the last round.

8.2.1 Transformation for Any LSSS with an Arbitrary Adversary Structure

In this subsection, we construct a transformation for any LSSS with an arbitrary access structure. We require pairwise secure channels along with a simultaneous broadcast channel. As in Chapter 5, by simultaneous broadcast channel, we mean that all participants can simultaneously send messages and so no participant can see what the others broadcast before sending his own message.

They are necessary to guarantee that no participant can tell whether the current round is the last round before sending his own message, which further guarantees that coalition members can never output the secret while preventing non-coalition members from outputing it. Assume that an authorized subset of participants want to recover the secret by playing a game. For ease of description, we can assume without loss of generality that those participants are $P_1, \ldots, P_{t^*}$. Now we describe a strategy for the mediator, who is a trusted third party, and a recommended strategy for each participant as follows. The game stops if the mediator or at least one player aborts.

The Mediator's Strategy

1. In the 0^{th} round, it collects the share from each participant P_i, $1 \leq i \leq t^*$. If it does not receive an appropriate message from some participant, it aborts abruptly. Otherwise, let $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_{t^*}^0})$ be the sequence of all the information it received. It checks whether there exists a vector $\eta \in \mathbb{F}^e$ such that $\eta \cdot (M_{\psi^{-1}(P_i)})^T = \widetilde{S_i^0}$ for $1 \leq i \leq t^*$. If not, it aborts abruptly; otherwise, it proceeds to the next round.
2. In the $r^{th}(r > 0)$ round, it chooses a random binary variable c^r with $\Pr[c^r = 1] = \alpha$ along with a random column vector $\omega^r \in \mathbb{F}^e$ whose first coordinate is zero, computes $S_i^r := c^r \cdot \widetilde{S_i^0} + \omega^r \cdot (M_{\psi^{-1}(P_i)})^T$ and sends to each participant P_i his round r share S_i^r, for $1 \leq i \leq t^*$. Note that if $c^r = 0$, $S_1^r, \ldots, S_{t^*}^r$ are the shares for 0; otherwise, if $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_{t^*}^0}) = (S_1^0, \ldots, S_{t^*}^0)$, $S_1^r, \ldots, S_{t^*}^r$ are also the shares for s.

Participant P_i's Recommended Strategy σ_i

1. In the 0^{th} round, send his initial share S_i^0 to the mediator.
2. In the $r^{th}(r > 0)$ round, broadcast S_i^r simultaneously. If he fails to receive complete information from some participant P_j, abort abruptly with no output; otherwise, check whether there exists a vector $\xi \in \mathbb{F}^e$ such that:

$$\xi \cdot (M_{\psi^{-1}(P_i)})^T = \widetilde{S_i^r}, \text{ for } 1 \leq i \leq t^*, \qquad \mathbf{(8.1)}$$

where $\{\widetilde{S_1^r}, \ldots, \widetilde{S_{t^*}^r}\}$ denotes the set of all the messages he received in the previous step, including his own. If not, abort immediately; otherwise, reconstruct a value, denoted as Y_i^r. If $Y_i^r \neq 0$, output it and abort; otherwise, proceed to the next round.

We are to show that $(\sigma_1, \ldots, \sigma_{t^*})$ induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies provided that α is suitably chosen.

Theorem 8.2.1. *If $\alpha \leq \min_{1\leq i\leq n}\frac{u_i-u_i^-}{u_i^+-u_i}$, then $(\sigma_1,\ldots,\sigma_{t^*})$ induces an $\mathcal{A}$-resilient Nash equilibrium which survives iterated deletion of weakly dominated strategies. It has expected running time $O(\frac{1}{\alpha})$.*

Proof. It is obvious that if every participant follows his strategy, with probability 1 everyone will eventually learn (and thus output) the secret and this is expected to happen in $O(\frac{1}{\alpha})$ rounds.

Consider what happens if some coalition $A \in \mathcal{A}$ of participants deviates from σ_A. There are two cases. Case 1: some participants in A either lie about or do not send their initial shares at all to the mediator in the 0^{th} round. Case 2: some participants in A either lie or do not broadcast their round r shares (received from the mediator in the r^{th} round for some $r > 0$). Now we are going to prove that under our assumption on α, each of these deviations would never bring to any participant in A more utility provided that the non-coalition participants stick to their strategy σ_{-A}, that is, σ induces an $\mathcal{A}$-resilient Nash equilibrium.

If some participant $P_i \in A$ sends nothing to the mediator in the 0^{th} round, the game stops and nobody outputs the secret, which obviously results in a worse outcome for participant P_i. If some participant $P_i \in A$ cheats about his initial share, we distinguish between two cases as follows. If there does not exist a vector $\eta \in \mathbb{F}^e$ such that $\eta \cdot (M_{\psi^{-1}(P_j)})^T = S_i^0$ for each $1 \leq j \leq t^*$. Then the game stops with no participant outputting the secret, which obviously is a worse outcome for P_i; otherwise, we can reduce this case to the situation where some coalition participant P_i lies about his round r share for some $r > 0$, which will be discussed below.

Now suppose that some $P_i \in A$ does not send his round r share to some participant not in A, for some $r > 0$. If $c^r = 0$, which happens with probability $1 - \alpha$, the game will terminate immediately and nobody may output the secret. However, if $c^r = 1$, which happens with probability α, the best outcome participant P_i can expect is to gain u_i^+. Thus, P_i gains from sending nothing only if $\alpha u_i^+ + (1-\alpha)u_i^- > u_i$, which is not the case by our assumption.

Suppose the participants in A send $X = \{\widetilde{S_i^r} : P_i \in A\}$ instead of $\{S_i^r : P_i \in A\}$ in the r^{th} round, for some $r > 0$. We say that X is a compatible response if there exists a vector $\zeta \in \mathbb{F}^e$ whose first coordinate is $c^r \cdot s$ such that

$$\zeta \cdot (M_{\psi^{-1}(P_i)})^T = \begin{cases} \widetilde{S_i^r}, & P_i \in A; \\ S_i^r, & P_i \in \{P_1,\ldots,P_{t^*}\} \setminus A. \end{cases}$$

Now we consider two subcases.

- X is a compatible response. Then sending X has the same result as sending $\{S_i^r : P_i \in A\}$. In fact, in this case, $\{\widetilde{S_i^r} : P_i \in A\} \bigcup \{S_i^r : P_i \notin A\}$ are also the shares for $c^r \cdot s$, that is, either with probability α all active participants output the secret, or with probability $1 - \alpha$ all active participants recover 0 as the secret and the game continues to the next stage.

- X is not a compatible response. If $c^r = 1$, which happens with probability α, each participant P_i in the coalition A outputs the secret and gains at most u_i^+. If $c^r = 0$, which happens with probability $1 - \alpha$, no participant in A outputs the secret so far and the game stops. To see why the game stops, we observe that any participant $P_j \notin A$ will check whether the equality (8.1) satisfies: if it satisfies, then the first coordinate should be nonzero (since now X is not a compatible response), that is, P_j recovers a nonzero value and so the game stops; if it does not satisfy, then P_j aborts according to his strategy and so the game stops. Therefore, $P_i \in A$ gains from sending an incompatible response only if $\alpha u_i^+ + (1 - \alpha)u_i^- > u_i$, which is not the case by our assumption.

Finally, the statement that $(\sigma_1, \ldots, \sigma_{t^*})$ survives iterated deletion of weakly dominated strategies follows directly from the sufficient condition given by Halpern and Teague [31] for a randomized protocol with unbounded running time to survive the iterated deletion process. This completes the proof. □

Remark 8.2.2. This transformation works for any linear secret sharing scheme over any access structure. However, the expected round complexity of the resulting rational secret sharing scheme relies on participants' utilities or at least some bounds on them. It can be large for some particular choices of utilities. In addition, the obtained RSSS relies on a simultaneous broadcast channel, which is expensive to be implemented in practice. However, if we begin with an LSSS with a Q^2 adversary structure, we can remove these two limitations at the cost of requiring all n players are involved in the secret reconstruction process.

8.2.2 Transformation for Any LSSS with a Q^2 Access Structure

Observe that if $\mathcal{A}$ is Q^2, then the non-coalition participants have enough shares from which they can reconstruct the secret even if the coalition participants do not send their round r shares for some $r > 0$. However, if the coalition participants send incorrect values, then for the non-coalition participants, there may be multiple subsets of shares from which multiple values can be recovered. This problem can be avoided if the mediator sends each participant some extra information he can use to verify the truth of the value sent by any other participant and hence can locate the possible errors. Hereinafter, we assume all participants in $\mathcal{P}$ are involved in the secret reconstruction phase, which is typically the case for rational multiparty computation. And in this subsection, we are dedicated to extending any LSSS with a Q^2 access structure to an RSSS, in which the verification process uses Rabin and Ben-Or's *information checking protocol* (ICP) [53]. Only pairwise secure channels are required in this subsection.

Beginning with any LSSS with a Q^2 access structure, we obtain a rational secret sharing scheme, which is described in the next page.

Remark 8.2.3. Observe that if P_i modifies the value of s_l^r to $\widetilde{s_l^r}$, for some $\lambda_{i-1}+1 \leq l \leq \lambda_i, r > 0$, he must also modify the value $y_{l,j}^r$ to $\widetilde{y_{l,j}^r}$ such that $c_{l,j}^r = b_{l,j}^r \widetilde{s_l^r} + \widetilde{y_{l,j}^r}$; otherwise, P_i will be caught cheating. And the probability of being able to guess an appropriate $\widetilde{y_{l,j}^r}$ is less than β.

Remark 8.2.4. From the above protocol, we can see why we do not need a simultaneous channel in this construction. Once coalition members can tell in advance whether the current round, say the r^{th} round, is the last round before sending their round r shares, it will be rational for them to keep silent or send incorrect round r shares. However, this can never prevent non-coalition members from learning the secret except with probability less than β, since non-coalition members form an authorized subset.

Recall that $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ is the tuple consisting of the messages the mediator received in the 0^{th} round. It can be different from $\mu = (S_1^0, \ldots, S_n^0)$, the tuple of the original shares for s given by the dealer. As in Chapter 3, let C be the linear block code generated by M^T over $\mathbb{F}$. The following lemma suggests a sufficient condition for $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ to be the shares for s.

Lemma 8.2.5. *Let $\boldsymbol{e} := (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n) = \vartheta - \mu$, where $\boldsymbol{e}_i = \widetilde{S_i^0} - S_i^0$, $1 \leq i \leq n$. If $\boldsymbol{e}$ is a block codeword of C and* $\operatorname{supp}(\boldsymbol{e}) := \{P_i | \boldsymbol{e}_i \neq \boldsymbol{0}\} \in \mathcal{A}$*, then $\widetilde{S_1^0}, \ldots, \widetilde{S_n^0}$ are also the shares for the secret s.*

The Mediator's Strategy

1. Choose a field $\mathbb{F}' = \mathrm{GF}(q^h)$ for some integer h such that q^h is larger than $\frac{1}{\beta}$, where β is a security parameter determined below.
2. In the 0^{th} round, collect the initial share from each participant P_i, $1 \leq i \leq n$. If it does not receive an appropriate message from some participants, abort abruptly; otherwise, check whether ϑ is a codeword of C, where $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ be the sequence of all the information it collects. If not, abort abruptly; otherwise, proceed to the next round.
3. In the r^{th} round with $r > 0$,
   (a) Choose a random binary variable c^r with $\Pr[c^r = 1] = \frac{1}{2}$ along with a random vector $\omega^r \in \mathbb{F}^e$ such that the first entry of ω^r is 0.
   (b) For $1 \leq i \leq n$, compute $S_i^r := (s^r_{\lambda_{i-1}+1}, \ldots, s^r_{\lambda_i}) = c^r \cdot \widetilde{S_i^0} + \omega^r \cdot (M_{\psi^{-1}(P_i)})^T$ and send to P_i his round r share S_i^r. Note that if $c^r = 0$, $S_1^r, \ldots, S_n^r$ are the shares for 0; otherwise, if $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ is a codeword of the block code C generated by M^T, $S_1^r, \ldots, S_n^r$ are the shares for s by Lemma 8.2.5.
   (c) For $1 \leq i \leq n$, send to P_i random sequences $(y^r_{\lambda_{i-1}+1,j}, \ldots, y^r_{\lambda_i,j})$ of elements in $\mathbb{F}'$, $j \in \{1, \ldots, n\} \setminus \{i\}$. Besides, for each $j \in \{1, \ldots, n\} \setminus \{i\}$, send to P_j m_i pairs $(b^r_{\lambda_{i-1}+1,j}, c^r_{\lambda_{i-1}+1,j}), \ldots, (b^r_{\lambda_i,j}, c^r_{\lambda_i,j})$ of elements in $\mathbb{F}'$ such that $c^r_{l,j} = b^r_{l,j} s^r_l + y^r_{l,j}$ and $b^r_{l,j} \neq 0$ for all $\lambda_{i-1} + 1 \leq l \leq \lambda_i$.

Participant P_i's Recommended Strategy σ_i'

1. In the 0^{th} round, send his initial share S_i^0 to the mediator.
2. In the r^{th} round with $r > 0$,
   (a) Receive the round r messages from the mediator.
   (b) For $1 \leq j \neq i \leq n$, send the sequence $(y^r_{\lambda_{i-1}+1,j}, \ldots, y^r_{\lambda_i,j})$ along with his round r share $S_i^r = (s^r_{\lambda_{i-1}+1}, \ldots, s^r_{\lambda_{i-1}+p_i})$ to participant P_j.
   (c) Verify the truth of the data from any other participants and reconstruct a value, say Y_i^r, by ignoring those round r shares that fail the verifications. Note that here if some participant P_j sends nothing or incomplete data to P_i, P_i may discard all the data from him and in this case, we say P_j fails the verification.
   (d) If $Y_i^r \neq 0$, output Y_i^r and abort. In the case that $Y_i^r = 0$, if all the verifications in previous step success, proceed to the next round; otherwise, abort with no output.

Proof. It suffices to show that $\mathbf{e}_1, \ldots, \mathbf{e}_n$ are the shares for 0, since the underlying secret sharing scheme is linear. Since $\mathbf{e} \in C$, $\mathbf{e}_1, \ldots, \mathbf{e}_n$ are reasonable shares for some element, say a, in $\mathbb{F}$. Let $O = \{P_i | \mathbf{e}_i = \mathbf{0}\}$, where $\mathbf{0}$ denotes zero vector of proper length. Since $\mathcal{A}$ is Q^2 and $\mathrm{supp}(\mathbf{e}) \in \mathcal{A}$, we have $O \in \Gamma$. Obviously, the value recovered by using the those shares from participants in O is 0, that is, $a = 0$. □

Theorem 8.2.6. *Assume that $\beta < \min_{1 \leq i \leq n} \frac{u_i - u_i^-}{2u_i^+ - u_i - u_i^-}$ and $\mathcal{A}$ is Q^2, then $(\sigma_1', \ldots, \sigma_n')$ induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies. It has expected round complexity $O(1)$.*

Proof. It is obvious that if every participant follows his strategy, with probability 1 everyone will eventually learn (and thus output) the secret and this may be expected to happen in 2 rounds.

Consider what happens if some coalition $A \in \mathcal{A}$ of participants deviates from the recommended strategy. There are two cases. Case 1: some participants in A either lie about or do not send their initial shares at all to the mediator in the 0^{th} round. Case 2: some participants in A either lie or send incomplete round r messages to some non-coalition participants. We proceed to prove that each of these deviations would never bring to coalition participants more utilities provided that the non-coalition participants stick to their recommended strategies.

If some participants in A send nothing to the mediator in the 0^{th} round, the game stops and nobody outputs the secret, which obviously results in a worse outcome for all participants, including those in A. If some participants in A cheat about their initial share, we distinguish between two cases as follows. Case 1: $\vartheta \notin C$. Then the game stops with no participant outputting the secret, which obviously is a worse outcome for all participants, including those in A. Case 2: $\vartheta \in C$. It follows immediately from Lemma 8.2.5 that the entries of ϑ are shares for s too. As a result, sending ϑ has the same result as sending μ, whose entries are the original shares obtained from the dealer. That is, this deviation can never increase coalition players' utilities.

Now suppose that some player $P_i \in A$ sends incomplete round r messages to some non-coalition participants, for some $r > 0$. If $c^r = 0$, which happens with probability $\frac{1}{2}$, the game terminates at the r^{th} round and nobody has gained the secret so far. However, if $c^r = 1$, which happens with probability $\frac{1}{2}$, each participant outputs the real secret, since each $P_j \notin A$ can recover the secret by ignoring those round r shares from participants in A. Thus, P_i may get less from sending partial messages since $\frac{1}{2}u_i^- + \frac{1}{2}u_i < u_i$. In other words, P_i has an incentive to send all of his round r

messages, fake or not, to each participant not in A in the r^{th} round for any $r > 0$.

Finally, suppose that some $P_i \in A$ cheats about his round r share. When $c^r = 1$, the non-coalition participants will output the secret unless some coalition $A \subseteq \mathcal{A}$ lie about their round r shares and not all of the lies are detected. However, the probability that a single lie is not detected is β and the probability that it is detected is $1 - \beta$; if there are more lies, the probability of detecting at least one of them is even greater. In fact, suppose that there are altogether l lies for some integer l. Then the probability of detecting at least one lie is $1 - \beta^l$ and the probability of detecting all the lies is $(1 - \beta)^l$. Thus, when $c^r = 1$, with probability $1 - (1 - \beta)^l$, each $P_i \in A$ gains at most u_i^+, and with probability $(1 - \beta)^l$, P_i gains u_i. When $c^r = 0$, with probability $1 - \beta^l$, at least one lie will be detected and the game will stop at the r^{th} round with no one learning the secret. If none of the lies is detected, which happens with probability β^l, the game will continue to the next stage and the most that P_i can hope to get is u_i^+. Thus, P_i's expected utility from being a member of a coalition $A \in \mathcal{A}$ is at most

$$\begin{aligned} & \frac{1}{2}((1 - (1 - \beta)^l)u_i^+ + (1 - \beta)^l u_i + \beta^l u_i^+ + (1 - \beta^l)u_i^-) \\ = \; & \frac{1}{2}[(1 - (1 - \beta)^l + \beta^l)u_i^+ + (1 - \beta)^l u_i + (1 - \beta^l)u_i^-] \\ < \; & \frac{1}{2}[2\beta u_i^+ + (1 - \beta)u_i + u_i^-], \end{aligned}$$

where the inequality is due to the fact that $1-(1-\beta)^l+\beta^l \leq 2\beta$, since $1-(1-\beta)^l+\beta^l$ is monotone decreasing with respect to l provided that $\beta \leq \frac{1}{2}$. So P_i would prefer it's coalition not to lie if $2\beta u_i^+ + (1 - \beta)u_i + u_i^- < 2u_i$, that is, if $\beta < \frac{u_i - u_i^-}{2u_i^+ - u_i - u_i^-}$. This completes the proof. □

The rational secret sharing scheme obtained from either of the two transformations induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies. However, as pointed by Kol and Naor [38], the solution concept *Nash equilibrium surviving iterated deletion of weakly dominated strategies* is not sufficiently strong, since some bad strategies may survive this deletion process. Take the RSSS obtained from the second transformation as an example. If some coalition $A \in \mathcal{A}$ sent fake initial shares to the mediator in the 0^{th} round in a way that ϑ is still a codeword of C, then this deviation would not be detected and would not decrease any participant's utility. To put it in another way, for any $A \in \mathcal{A}$ the following bad strategy $\widetilde{\sigma'_A}$: coalition members follow their strategy σ'_A in the r^{th} round for each $r > 0$ but deviate in 0^{th} round in the particular way described above, while any non-coalition participant P_j sticks to his strategy σ'_j, can never be a weakly dominated strategy and consequently survives iterated deletion of weakly dominated strategies. However, if we required the existence of on-line dealer, it is easy to see from the proof of Theorem 8.2.6 that σ' induces an $\mathcal{A}$-resilient strict Nash equilibrium. Now we would like to claim that even in the case that the dealer is not available after the share distribution phase (which is the case in our model), the obtained RSSS still induces an $\mathcal{A}$-resilient strict Nash equilibrium provided that $\mathcal{A}$ is Q^2 and the LSSS to be transformed is $\mathcal{A}$-error detectable. Note that as mentioned in Chapter 4, the solution concept *strict Nash equilibrium* is stronger than *Nash equilibrium* surviving iterated deletion of weakly dominated strategies. In particular, the bad strategy $(\widetilde{\sigma'_A}, \sigma'_{-A})$ is not an A-resilient strict Nash equilibrium while it is an A-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies.

Let Γ be an access structure and let $\mathcal{M} = (\mathbb{F}, M, \varepsilon, \psi)$ be an MSP computing Γ. As in Chapter 3, let C be the linear block code generated by M^T. As before, C is called the linear block code corresponding to this $\mathcal{M}$. Recall that we say that $\mathcal{M}$ is $\mathcal{A}$-error detectable if C is $\mathcal{A}$-error detectable, that is, for any nonzero word $\omega = (\mathbf{w}_1, \ldots, \mathbf{w}_n) \in \mathbb{F}^{m_1} \times \cdots \times \mathbb{F}^{m_n}$ with $\mathrm{supp}(\omega) := \{P_i | \mathbf{w}_i \neq 0\} \in \mathcal{A}$, ω can not be a block codeword of C. Here $m_i = |\psi^{-1}(P_i)|$, $1 \leq i \leq n$.

Proposition 8.2.7. $(\sigma'_1, \ldots, \sigma'_n)$ *induces an $\mathcal{A}$-resilient strict Nash equilibrium provided that $\mathcal{M}$ is $\mathcal{A}$-error detectable.*

Proof. The assumption that $\mathcal{M}$ is $\mathcal{A}$-error detectable directly implies that if $\vartheta \neq \mu$, then ϑ can not be a codeword of C. Hence, any modification about the original shares sent to the mediator in the 0^{th} round may be detected and hence cause the protocol to abort immediately with nobody outputting the secret, which is clearly a worse outcome for each participant in some coalition. As showed in the proof for Theorem 8.2.6, any other possible deviation from the participants in some coalition may also result in a decrease in each coalition member's utility, which completes the proof. □

Remark 8.2.8. Let $\mathcal{M}$ be an MSP computing Γ, where the adversary structure is denoted by $\mathcal{A}$. By Theorem 3.3.6, $\mathcal{M}$ is $\mathcal{A}$-error detectable if and only if $\mathcal{A}$ is Q^2 and the rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}$. Thus, our protocol induces a strict Nash equilibrium provided that rows of $(M^{\perp})_B$ are linearly independent for any $B \in \mathcal{A}$.

8.2.3 Transformation for Any LSSS with a Q^3 Access Structure

Now we would like to point out that the obtained RSSS from either of the two transformations we have proposed so far assumes that knowledge of participants' utilities (or at least some bounds on them) is public. However, this assumption may be problematic, since in reality a player does not necessarily know his utility, let alone let others know. Furthermore, even if he knows, it is unclear how others can know it. Actually, all the existing rational secret sharing schemes without a mediator share this limitation. Under the assumption of the existence of the mediator, Abraham *et al.* [1] proposed a *utility independent* rational protocol for any Q^3 threshold access structure. Here, by *utility independence*, we mean that the protocol works for all possible values of utilities satisfying the aforementioned learning preference assumptions: $u_i > u_i^-, 1 \leq i \leq n$. In this subsection, we demonstrate how to transform any LSSS with a Q^3 access structure to a utility independent RSSS with expected round complexity $O(1)$. We borrow the technique proposed by Kurosawa [40]. For notation simplicity, we only deal with the case when $m = n$ and $\psi(i) = P_i$ for $i = 1, \ldots, n$ and our construction can be extended directly to general case.

The Mediator's Strategy

1. In the 0^{th} round, collect the initial share from each participant P_i, $1 \leq i \leq n$. If it does not receive an appropriate message from some participant, abort abruptly; otherwise check whether ϑ is a codeword of C, where $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ be the sequence of all the information it collects. If not, abort abruptly; otherwise, proceed to the next round.

2. In the r^{th} round with $r > 0$,

   (a) Choose a random binary variable c^r with $\Pr[c^r = 1] = \frac{1}{2}$ along with a random vector $\omega^r \in \mathbb{F}^e$ such that the first entry of ω^r is 0.

   (b) For $1 \leq i \leq n$, compute $S_i^r := (c^r \cdot \vartheta + \omega^r) \cdot (M_{\psi^{-1}(P_i)})^T$. Note that if $c^r = 0$, $S_1^r, \ldots, S_n^r$ are the shares for 0; otherwise, if $\vartheta = (\widetilde{S_1^0}, \ldots, \widetilde{S_n^0})$ is a codeword of C, $S_1^r, \ldots, S_n^r$ are the shares for s.

   (c) For each $1 \leq i \leq n$, redistribute S_i^r among all participants: choose a random vector $\rho_i^r \in \mathbb{F}^{e-1}$, compute $(s_{i,1}^r, \ldots, s_{i,n}^r) := (S_i^r, \rho_i^r) \cdot M^T$, and send (S_i^r, ρ_i^r) along with $s_{i,i}^r$ to P_i, while send $s_{i,j}^r$ to P_j, $1 \leq j \neq i \leq n$.

Participant P_i's Recommended Strategy δ_i

1. In the 0^{th} round, send his initial share S_i^0 to the mediator.
2. In the r^{th} round with $r > 0$,
   (a) Receive the round r messages (S_i^r, ρ_i^r) and $s_{l,i}^r$, $1 \leq l \leq n$, from the mediator.
   (b) For $1 \leq j \neq i \leq n$, send (S_i^r, ρ_i^r) along with $s_{l,i}^r$, $1 \leq l \leq n$, to P_j.
   (c) Let $\mathrm{Index}_i^r := \{j : P_i$ collects complete data from P_j at the r^{th} round $\}$. Note that $i \in \mathrm{Index}_i^r$. Let $\{(\widetilde{S_j^r}, \widetilde{\rho_j^r}), \widetilde{s_{1,j}^r}, \ldots, \widetilde{s_{n,j}^r}\}$ be the messages he collects from P_j, $j \in \mathrm{Index}_i^r$, during the previous step and let $X_j^r := (\widetilde{s_{j,1}^r}, \ldots, \widetilde{s_{j,n}^r})$, where $\widetilde{s_{j,l}^r}$ can be the special symbol $\perp$ if $j \notin \mathrm{Index}_i^r$ and in this case $P_j \in \mathrm{supp}(X_j^r - (\widetilde{S_j^r}, \widetilde{\rho^r}) \cdot M^T)$.
   (d) For each $j \in \mathrm{Index}_i^r$, if $\mathrm{supp}(X_j^r - (\widetilde{S_j^r}, \widetilde{\mathbf{r}^r}) \cdot M^T) \in \mathcal{A}$, set $\mathbf{share}_j^r := \widetilde{S_j^r}$; otherwise, set $\mathbf{share}_j^r := \perp$.
   (e) Reconstruct a value, say Y_i^r, by using those shares from
   $\{\mathbf{share}_j^r : j \in \mathrm{Index}_i^r$ and $\mathbf{share}_j^r \neq \perp\}$.
   (f) If $Y_i^r \neq 0$, output Y_i^r and abort. In the case $Y_i^r = 0$, if $\mathrm{Index}_i^r = \{1, \ldots, n\}$ and $X_j^r = (\widetilde{S_j^r}, \widetilde{\mathbf{r}^r}) \cdot M^T$ for all $1 \leq j \leq n$, proceed to the next round; otherwise, abort with no output.

Kurosawa first proposed and proved the following result in [40], which is essential for the proof of Theorem 8.2.10.

Lemma 8.2.9 ([40] Lemma 4.1)**.** *Suppose that $\mathcal{A}$ is Q^3. For $1 \leq j \leq n$, for each $r > 0$ such that the participants are still playing in the r^{th} round, either* $\boldsymbol{share}_j^r = S_j^r$ *or* $\boldsymbol{share}_j^r = \perp$. *Furthermore,* $\boldsymbol{share}_j^r = S_j^r$ *if P_j follows his recommended strategy honestly.*

Theorem 8.2.10. *Suppose that $\mathcal{A}$ is Q^3. $\delta = (\delta_1, \ldots, \delta_n)$ induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies. It has expected round complexity $O(1)$.*

Proof. It is obvious that if every participant follows his strategy, with probability 1 everyone will eventually learn (and thus output) the secret s and this will be expected to happen in 2 rounds.

The argument for $\mathcal{A}$-resiliency proceeds much as that for Theorem 8.2.6. We will only focus on case 2. Suppose that some coalition participants only send part of their round r messages to some non-coalition participants, for some $r > 0$. If $c^r = 0$, which happens with probability $\frac{1}{2}$, the game terminates at the r^{th} round and nobody may output the secret. However, if $c^r = 1$, which happens with probability $\frac{1}{2}$, each participant will output the real secret, since each $P_j \notin A$ can recover the secret by ignoring those round r shares from participants in A. Thus, by this deviation each $P_i \in A$ may gain $\frac{1}{2}u_i^- + \frac{1}{2}u_i$, which is strictly less than u_i. Thus, P_i will send all of his round r messages, fake or not, to each non-coalition participant in the r^{th} round for any $r > 0$. Now we continue to consider what happens if some coalition participants send fake round r messages to some non-coalition participant, for some $r > 0$. However, we claim that this kind of deviation can never bring more utility to any participant P_i in A. To show this, we distinguish between two cases as follows. When $c^r \neq 0$, every participant outputs the secret by Lemma 8.2.9. When $c^r = 0$, if $\text{Index}_i^r = \{1, \ldots, n\}$ and $X_j^r = (\widetilde{S_j^r}, \widetilde{\rho^r}) \cdot M^T$ for all $1 \leq j \leq n$, the game proceeds to the next round; otherwise, it terminates at the r^{th} round with no one outputting the secret. In all, these deviations can never increase coalition players' utilities, that is, δ induces an $\mathcal{A}$-resilient Nash equilibrium.

□

Remark 8.2.11. The main advantage of the RSSSs obtained from the third transformation is that they do not rely on knowledge of participants' utilities, i.e., they are utility independent. In other words, the obtained RSSSs work for all choices of numerical utility as long as the participants do not strictly prefer not learning the secret to learning the secret.

We would like to stress that we can do better if the the underlying LSSS with a Q^3 adversary structure $\mathcal{A}$ is $\mathcal{A}$-error detectable.

Proposition 8.2.12. *Let $\mathcal{M}$ be a monotone span program computing Γ. If $\mathcal{M}$ is $\mathcal{A}$-error detectable and $\mathcal{A}$ is Q^3, then δ induces an $\mathcal{A}$-resilient strict Nash equilibrium.*

Proof. Suppose that $\mathcal{M}$ is $\mathcal{A}$-error detectable. On one hand, ϑ is a codeword in C only if $\vartheta = \mu$, i.e., any modification to the shares sent to the mediator in 0^{th} round can be detected and cause the game to abort and no one outputs the secret, which is a worse outcome to the participants in some coalition $A \in \mathcal{A}$. On the other hand, if $X_j^r = (\widetilde{s_{j,1}^r}, \ldots, \widetilde{s_{j,n}^r}) = (\widetilde{S_j^r}, \widetilde{\rho^r}) \cdot M^T$ for each $1 \leq j \leq n$, then $(\widetilde{s_{j,1}^r}, \ldots, \widetilde{s_{j,n}^r}) = (s_{j,1}^r, \ldots, s_{j,n}^r)$ and $(\widetilde{S_j^r}, \widetilde{\rho^r}) = (S_j^r, \rho^r)$, since $(\widetilde{s_{j,1}^r}, \ldots, \widetilde{s_{j,n}^r}) - (s_{j1}^r, \ldots, s_{jn}^r)$ is a codeword of C and its support is in $\mathcal{A}$. Consequently, any fake information sent in the r^{th} round with $r > 0$ can eventually be detected. Now suppose some participants in a coalition $A \in \mathcal{A}$

send incomplete or fake messages in the r^{th} round for some $r > 0$. If $c^r = 0$, which happens with probability $\frac{1}{2}$, the game stops at the r^{th} round and no one outputs the secret; if $c^r = 1$, which happens with probability $\frac{1}{2}$, everyone outputs the secret by Lemma 8.2.9. Hence, the expected utility any participant P_i in some coalition may gain by sending fake information is $\frac{1}{2}u_i^- + \frac{1}{2}u_i$, which is strictly less than u_i. The rest of the proof follows directly from the proof in Theorem 8.2.10. □

8.3 Conclusion

In this chapter, we propose three transformations from a traditional LSSS to an RSSS with a mediator. The RSSS obtained induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies, relies on no cryptographic assumption, provides information theoretical security and hence is immune against backward induction attacks. If we begin with an $\mathcal{A}$-error detectable LSSS with a Q^3 adversary structure, the RSSS obtained induces an $\mathcal{A}$-resilient strict Nash equilibrium. Those transformations assume the existence of an online mediator. It should be interesting to consider whether it is possible to remove the online mediator and how to remove it if it is possible at all. In particular, if we begin with any $\mathcal{A}$-error correctable LSSS and provided that there is an efficient decoding algorithm, then we can obtain a utility independent RSSS without a mediator.

9. GENERAL CONCLUSION AND OPEN QUESTIONS

This thesis talked about two related cryptographic primitives: (classical) secret sharing and rational secret sharing. Secret sharing was introduced in order to facilitate the distributed storage of private data in an unreliable environment. Secret sharing schemes allow a dealer to distribute a secret among a set of finite players in a way that each authorized set can later recover the secret by combining their shares, while any unauthorized set can not. This purpose can be achieved for certain only under the assumption that each player is honest. However, in reality (as in game theory), players are self-interested, that is, they may choose to be dishonest if deviating from the protocol will provide them more benefit. One approach to bridging game theory and secret sharing is to use tools from game theory to design secret sharing schemes such that rational players will be motivated to follow. The resulting schemes are called rational secret sharing schemes. There are several criteria to evaluate a rational protocol, such as security models (unconditionally secure or computationally secure), channel models, coalition-resilience abilities, equilibrium types achieved, utility dependence and efficiencies (including expected round complexity, computational complexity and share bitsize).

In this thesis we proposed two rational secret sharing protocols. The first one uses the techniques of proactive secret sharing scheme and one-time pad. The main advantage is that it is unconditionally secure. It induces a Nash equilibrium surviving iterated deletion of weakly dominated strategies in the simultaneous broadcast channels model. Since the cryptographic community is still in search of a proper framework for rational protocols, it would be interesting to study the benefits of our approach in different equilibrium contexts. The second one is based on the Chinese Remainder Theorem and it induces a $(t-1)$-resilient computational strict Nash equilibrium that is stable with respect to trembles in the non-simultaneous broadcast channels model. The trade off is that its security relies on some computational assumptions related to discrete logarithm and RSA.

On one hand, these two schemes share with almost all of the existing rational protocols the common limitation: the solution concept it induces is not sufficiently strong, especially when there are multiple strict Nash equilibria that are stable with respect to trembles or multiple Nash equilibria surviving iterated deletion of weakly dominated strategies, and consequently the game may not end in a Nash equilibrium at all. Micali and Shelat [43] proposed a protocol, which induces a solution concept stronger than strict Nash equilibrium and Nash equilibrium surviving iterated deletion of weakly dominated strategies, since in their protocol after the process of iterated deletion of weakly dominated strategies, surviving strategy of each player in their protocol is unique. However, the security of their construction relies on special communication channels like ordinary envelopes (as a way of temporarily and perfectly hiding a secret value), which may be quite expensive to be implemented in practice. Hence, it should be interesting to construct rational protocols which induce a stronger (than Nash equilibrium and its variants) solution concept in standard communication networks.

On the other hand, most of the existing rational protocols including ours relies on the assumption that knowledge on players' utilities (or at least some bounds on them) is public. However, as mentioned previously, this assumption may be problematic in practice. Thus, it makes sense to consider the problem of designing utility independent rational protocol. In the literature of rational secret sharing, there are two utility independent protocols for threshold access structures, both of which require an on-line trusted party (a dealer or a mediator) and digital signatures. It should be not hard to remove an online trusted party. As far as computational security is concerned, it is inherent for utility independent rational protocols, that is, there does not exist an unconditional secure rational secret sharing protocol while preserving utility independence. Hence, in order to achieve information-theoretic security, the requirement for utility independence should be relaxed in a *reasonable* way.

In addition, we proposed three transformations from a traditional LSSS to an RSSS with a mediator. The RSSS obtained induces an $\mathcal{A}$-resilient Nash equilibrium surviving iterated deletion of weakly dominated strategies, relies on no cryptographic assumption, provides information theoretical security and hence is immune against backward induction attacks. All of those transformations assume the existence of an online mediator. It should be interesting to consider whether it is possible to remove the online mediator and how to remove it if it is possible at all. In particular, if we begin with any $\mathcal{A}$-error correctable LSSS with a Q^3 adversary structure and provided that there is an efficient decoding algorithm, then we can obtain a utility independent RSSS without a mediator. However, we have no idea about whether there exists and how can we find an efficient decoding algorithm for a particular LSSS over an arbitrary monotone access structure.

BIBLIOGRAPHY

[1] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In *Proceedings of the 25th ACM Symposium on Principles of Distributed Computing*, pages 53–62, 2006.

[2] G. Asharov and Y. Lindell. Utility dependence in correct and fair rational secret sharing. In *Advances in Cryptology-CRYPTO 2009*, pages 559–576, 2009.

[3] C. Asmuth and J. Bloom. A modular approach to key safeguarding. *IEEE Transactions on Information Theory*, 29(2):208–210, 1983.

[4] P. Béguin and A. Cresti. General short computational secret sharing schemes. In *Advances in Cryptology-EUROCRYPT 1995*, pages 194–208, 1995.

[5] A. Beimel. *Secure schemes for secret sharing and key distribution.* PhD thesis, Department of Computer Science, Technion, 1996.

[6] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computations. In *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing*, pages 1–10, 1998.

[7] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptolog-CRYPTO 1988*, pages 27–36, 1988.

[8] M. Bertilsson and I. Ingemarsson. A construction of practical secret sharing schemes using linear block codes. In *Advances in Cryptology-AUSCRYPT 1992*, pages 67–79, 1992.

[9] G. R. Blakley. Safeguarding cryptographic keys. In *American Federation of Information Processing Societies, 1979 National Computer Conference*, pages 313–317, 1979.

[10] F. Boudot. Efficient proofs that a committed number lies in an interval. In *Advances in Cryptology-EUROCRYPT 2000*, pages 431–444, 2000.

[11] E. F. Brickell. Some ideal secret sharing schemes. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 6:105–113, 1989.

[12] E. F. Brickell and D. M. Davenport. On the classification of ideal secret sharing schemes. *Journal of Cryptology*, 4(2):123–134, 1991.

[13] Zhengjun Cao and Lihua Liu. Boudot's range-bounded commitment scheme revised. In *Proceedings of the 9th International Conference on Information and Communications Security*, pages 230–238, 2007.

[14] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the 20th AnnualACM Symposium on the Theory of Computing*, pages 11–19, 1998.

[15] H. Chernoff. A measure of asymptotic efficiency for tests of a hypotheisis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952.

[16] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *Proceedings of the 26th Annual Symposium on the Foundations of Computer Science*, pages 383–395, 1985.

[17] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In *Advances in Cryptology-EUROCRYPT 1999*, pages 311–326, 1999.

[18] R. Cramer, I. Damgård, and U. M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *Advances in Cryptology-EUROCRYPT 2000*, pages 316–334, 2000.

[19] L. Csirmaz. The size of a share must be large. In *Advances in Cryptology-EUROCRYPT 1994*, pages 13–22, 1994.

[20] P. D'Arco and D. R. Stinson. On unconditionally secure robust distributed key distribution centers. In *Advances in Cryptology-ASIACRYPT 2002*, pages 346–363, 2002.

[21] Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures (extended abstract). In *Advances in Cryptology-CRYPTO 1991*, pages 457–469, 1991.

[22] S. Fehr. Efficient construction of dual span program. In *Manuscript*, 1999.

[23] S. Fehr and U. Maurer. Linear vss and distributed commitments based on secret sharing and pairwise checks. In *Advances in Cryptology-CRYPTO 2002*, pages 565–580, 2002.

[24] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proceedings of the 28th IEEE Symposium on Foundations of Computer Science*, pages 427–437, 1987.

[25] P. Feldman and S. Micali. An optimal probabilistic algorithm for synchronous byzantine agreement. In *Proceedings of the 16th International Colloquium on Automata, Languages and Programming*, pages 341–378, 1989.

[26] F. M. Forges. An approach to communication equilibria. *Econometrica*, 54(6):1375–1385, 1986.

[27] G. Fuchsbauer, J. Katz, and D. Naccache. Efficient rational secret sharing in standard communication networks. In *Proceedings of the 7th IACR Theory of Cryptography Conference*, pages 419–436, 2010.

[28] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity for all languages in np have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991.

[29] S. D. Gordon and J. Katz. Rational secret ssharing, revisited. In *Proceedings of the 5th Conference on Security and Cryptography for Networks*, pages 229–241, 2006.

[30] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13rd ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[31] J. Y. Halpern and V. Teague. Rational secret sharing and multiparty computation: extended abstract. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 623–632, 2004.

[32] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing or: how to cope with perpetual leakage. In *Advances in Cryptology-CRYPTO 1995*, pages 339–352, 1995.

[33] M. Ito, A. Saito, and T. Nishizek. Secret sharing schemes realizing general access structure. In *Proceedings of the IEEE Global Telecommunication conf.*, pages 99–102, 1987.

[34] S. Izmalkov, S. Micali, and M. Lepinski. Rational secure computation and ideal mechanism design. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science*, pages 585–595, 2005.

[35] M. Karchmer and A. Wigderson. On span programs. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 102–111, 1993.

[36] K. Kaya and A. A. Selçuk. Secret sharing extensions based on the chinese remainder theorem. In *eprint.iacr.org/2010/096*, 2010.

[37] G. Kol and M. Naor. Cryptography and game theory: designing protocols for exchanging information. In *Proceedings of the 5th Theory of Cryptography Conference*, pages 320–339, 2008.

[38] G. Kol and M. Naor. Games for exchanging information. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 423–432, 2008.

[39] H. Krawczyk and D. R. Stinson. Secret sharing made short. In *Advances in Cryptology-CRYPTO 1993*, pages 136–146, 1993.

[40] K. Kurosawa. General error decodable secret sharing scheme and its application. *IACR Cryptology ePrint Archive*, page 263, 2009.

[41] M. Liu and Z. F. Zhang. *Secret sharing schemes and secure multiparty computation.* Publishing House of Electronics Industry, 2008.

[42] R. J. McEliece and D. V. Sarwate. On sharing secrets and reed-solomon codes. *Communications of the ACM*, 24(9):583–584, 1981.

[43] S. Micali and A. Shelat. Purely rational secret sharing (extended abstract). In *Proceedings of the 6th Theory of Cryptography Conference*, pages 54–71, 2009.

[44] M. Naor and A. Wool. Access control and signatures via quorum secret sharing. *IEEE Transactions on Parallel and Distributed Systems*, 9(9):909–922, 1998.

[45] V. Nikov and S. Nikova. On a relation between verifiable secret sharing schemes and a class of error-correcting codes. In *Proceedings of International Workshop on Coding and Cryptography*, pages 275–290, 2005.

[46] V. Nikov, S. Nikova, and B. Preneel. On the size of monotone span programs. In *Proceedings of the 4th Conference on Security in Communication Networks*, pages 249–262, 2004.

[47] V. Nikov, S. Nikova, B. Preneel, and J. Vandewalle. Applying general access structure to proactive secret sharing schemes. In *Proceedings of the 23rd Symposium on Information Theory in the Benelux*, pages 197–206, 2002.

[48] S. J. Ong, D. Parkes, A. Rosen, and S. Vadhan. Fairness with an honest minority and a rational majority. In *Proceedings of the 6th Theory of Cryptography Conference*, pages 36–53, 2009.

[49] M. J. Osborne and A. Rubinstein. *A course in game theory.* MIT Press, 1994.

[50] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology-CRYPTO 1991*, pages 129–140, 1991.

[51] M. O. Rabin. Randomized byzantine generals. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, pages 403–409, 1983.

[52] T. Rabin. Robust sharing of secrets when the dealer is honest or cheating. *Journal of the ACM*, 41(6):1089–1109, 1994.

[53] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 73–85, 1989.

[54] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[55] A. N. Reyzin. Rational secret sharing. 2007.

[56] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[57] D. R. Stinson and R. Wei. Unconditionally secure proactive secret sharing scheme with combinatorial structures. In *Proceedings of the 6th Annual International Conference on selected Area in Cryptography*, pages 200–214, 1999.

[58] C. Tartary, H. Wang, and Y. Zhang. An efficient and information theoretically secure rational secret sharing scheme based on symmetric bivariate polynomials. *International Journal of Foundations of Computer*, pages 1395–1416, 2011.

[59] T. Tassa. Generalized oblivious transfer by secret sharing. *Design Codes and Cryptography*, 58(1):11–21, 2011.

[60] M. Tompa and H. Woll. How to share a secret with cheaters. *Journal of Cryptology*, 1(2):133–138, 1988.

[61] T. Worsch. Lower bounds for (sums of) binomial coefficients. In *Technical Report 31/94, Universität für, Informatik*, 1994.

[62] Y. Zhang, C. Tartary, and H. Wang. An efficient rational secret sharing scheme based on the chinese remainder theorem. In *Proceedings of the 16th Australasian Conference on Information Security and Privacy*, pages 259–275, 2011.