D-01

マジックプロトコルを用いたP2P環境での公正なネットワークブラックジャック

Fair Network Blackjack in P2P Environment Using Magic Protocols

畠山貴行 *　　宮崎修一 †
Takayuki Hatakeyama　　Shuichi Miyazaki

1　はじめに

ネットワークの発達に伴い, ネットワーク通信を用いたゲームが普及している. 通常ネットワーク通信を伴うゲームは, サーバが公正にゲームを管理することを前提に設計される. 公平な審判サーバが存在すれば, 多くの場合簡単かつ公平にゲームを行うことができる. しかし, 全てのネットワークゲームにサーバを用意することは多くのリソースやコストを必要とする. また審判サーバが必ずしも信用できるとは限らない. そこで本稿ではサーバを利用せず, プレイヤー同士の, P2P 環境で行われるゲームについて考える.

将棋や囲碁のような完全情報ゲームでは, 全ての情報がオープンになっているため, プレイヤー間で交互に指し手を交換していけばよく, 比較的簡単に P2P 環境でのゲームを設計することが出来る. しかし, トランプや麻雀のような不完全情報ゲームでは, 自分の手札のように他人に見せない情報や, 山札のように全プレイヤーに対して秘密の情報を扱う必要があり, それをどう管理するかが問題となる. 例えば山札を特定のプレイヤーが管理する場合は, そのプレイヤーが山札を覗き見をすることが可能となるし, 他人に見えないのを良いことに, 自分の手札をすり替えることも可能となる. プレイヤーが「相手が不正を行っているかもしれない」と考えることは, 結果としてゲームに対しての楽しみが損なわれてしまう. またオンラインカジノのような場合では, 金銭が絡むため公正にゲームが行われることは必要不可欠である. そのため悪意を持ったプレイヤーが存在したとしても不正が行われることのないプロトコルが必要である.

本研究では, 不正を許さずに P2P 環境でカードゲームを行えるプロトコルを設計する. この実現にあたっては, 暗号を複合的に利用して設計されているマジックプロトコルを利用した. 本稿では, カードゲームの基本となる山札の生成, カードの配布, 二者間での数の大小判定などのプロトコルを, メンタルポーカープロトコル, および金持ちの財産比べプロトコルと呼ばれるマジックプロトコルを利用して設計した. また, これらのプロトコルを組み合わせることにより, 具体例としてブラックジャックを実行するプロトコルを設計し実装した.

*京都大学 大学院情報学研究科
Graduate School of Informatics, Kyoto University
†京都大学 学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

2　関連研究

本節では関連研究として代表的なマジックプロトコルであるメンタルポーカープロトコル, および金持ちの財産比べプロトコルについて説明する.

2.1　メンタルポーカープロトコル

メンタルポーカー [1, 2] は「電話でポーカーをするための方法」であり, 離散対数問題が困難であることを前提として, 安全 (公平) にカードを配布する方法を実現している. 今, ディーラー A が 52 枚のカードを公平にプレイヤー B に 5 枚配布することを考える. このとき, 安全なカード配布とは以下のような要件を満たすものである.

1. ディーラー A は 52 枚のカードを持っており, そのうち 5 枚をプレイヤー B に配布する.
2. A は, どのカードを配布したかを知ることができない.
3. B は, 配布されるカードを選ぶことができない (配布されるカードは無作為に決定される).

ここで本稿では暗号化, 及び復号に用いる鍵を次のように定義している.

$$K_1 = (E_1, D_1).$$

ここで鍵 K_1 は暗号化関数 E_1 と復号関数 D_1 の組を表している. このような暗号化関数によって暗号化されたものを暗号文, 暗号化されていないものを平文と呼ぶ. また x, y は暗号化および復号の対象となる数値表現とする. メンタルポーカープロトコルで用いる鍵の生成方法は 2.3.1 で示す.

メンタルポーカーでは, 上記の機能を実現するために, 以下のような特殊な暗号を仮定している.

1. $D_1(E_1(x)) = x.$
2. $E_2(E_1(x)) = E_1(E_2(x)).$

3. x および $E_1(x)$ が与えられたとき, 鍵 1 を計算して求めるのは非常に難しい.

4. いかなる平文 x および y の組に対しても, $E_1(x) = E_2(y)$ なる鍵 1, 2 の組を見つけるのは非常に難しい.

メンタルポーカープロトコルでは, 2 の条件が特徴的である. これは, 用いられる鍵が平文を複数の鍵で複数回暗号化する場合, 暗号化の順序を入れ替えても得られる結果が等しい可換な性質を持つことを表す. 同様に, 複数回にわたって暗号化された暗号文を復号する場合に, いかなる順序で復号しても等しく平文が得られることを表す. このような暗号系を利用して, 以下のような手順でメンタルポーカーを実現できる.

1. ディーラー A は鍵 $K_A = (E_A, D_A)$ を生成し, 52 枚のカード (数値表現) を E_A を用いて暗号化し, プレイヤー B に渡す.

2. B は, 52 枚のカードのうち 5 枚を選ぶ. 暗号化されているので, 選んだカードが何かは分からない.

3. B は鍵 $K_B = (E_B, D_B)$ を生成し, 選んだ 5 枚のカードを E_B を用いて暗号化する. カードの数値表現を c とすると, この時点で 5 枚のカードは $E_B(E_A(c))$ のように暗号化されている.

4. B は暗号化した 5 枚のカードを A に渡す.

5. A は渡された 5 枚のカードを鍵 K_A を用いて復号する. カードは $E_B(c)$ という状態になり, A は B がどのカードを選んだか分からない.

6. A は復号した 5 枚のカードを B に渡す. B はそれらを鍵 K_B を用いて復号し, 5 枚のカードの内容を知る.

メンタルポーカープロトコルで用いられる鍵 K_A, K_B はそれぞれ A, B 自身以外には中身が知られていないものとする.

2.2 金持ちの財産比べプロトコル

金持ちの財産比べプロトコルとは, 2 人の金持ちがどちらの方が金持ちであるかを, お互いの財産を秘密にして比べるプロトコルである [3, 4]. 金持ちの財産比べプロトコルは公開鍵暗号を用いて構築されたプロトコルである. 公開鍵暗号とは, 鍵 K_X に対して, 暗号化関数 E_X が鍵 K_X を保持している人以外も知っている暗号のことを指す. 金持ちの財産比べプロトコルで用いる鍵の生成方法は 2.3.2 節で述べる. ここで, 金持ちの財産比べプロトコルを以下に示す. 今, A の財産が x 億円, B の財産が y 億円であるとする. ただし x, y はそれぞれ $1 \leq x, y \leq 100$ を満たす整数であるとする.

1. A は, 自分の公開鍵暗号の鍵 $K_A = (E_A, D_A)$ を作成し暗号化関数 E_A を B に送る.

2. B は, ランダムな値 r を選び (r は E_A の定義域から選ぶ),
$$z = E_A(r) - y$$
を計算し, z を A に送る.(E_A の値域は, 整数値の連続な区間とする.)

3. A は,
$$u_i = D_A(z + i)(i = 1, 2, \ldots, 100)$$
を計算する. 次に, A は, 適当な数 p を定め,
$$v_i = E_A(u_i + p)$$
を計算し, B に以下を p とともに送る.
$$v_1, v_2, \ldots, v_x, v_{x+1} + 1, v_{x+2} + 2, \ldots, v_{100} + 1.$$

4. B は, $v_y = E_A(y+p)$ かどうかを調べ, 成立すれば $y \leq x$ と判定し, さもなければ $y > x$ と判定する.

もし, A と B が上記の手順を正しく実行すれば, 明らかに x と y の大小関係を正しく出力できる. また, 秘密を相手に漏らさない点では, B が A に送る情報は z だけであり, r をランダムに選べば明らかに y に関する情報を一切漏らさない. 一方, D_A を知らない B にとって $(u_i, \ldots, u_{100})$ のうち $u_y (= r)$ 以外は未知であるため, $(v_1, v_2, \ldots, v_x, v_{x+1}+1, \ldots, v_{100}+1)$ のうち, $v_y (= E_A(r)$ もしくは $E_A(r)+1)$ 以外は, 乱数と同様に見える. B は r の値を知っているため, 結局 A は B に大小関係の 1 ビット以外を漏らさないことになる.

2.3 暗号技術

本節では, 2.1 節, 2.2 節で述べたプロトコルで用いる鍵の生成方法について述べる.

2.3.1 可換な暗号

メンタルポーカープロトコルを実現するためには 2.1 節で示したような特殊な暗号系が必要である. 今回は以下のような暗号系を利用する [2].

まず暗号鍵, 復号鍵を各プレイヤーが以下のように生成する.

1. 十分に大きな整数 n を, プレイヤー間で了解する.

2. 各プレイヤーは n をもとに暗号鍵, 復号鍵を生成する. プレイヤー 1 人分の手順を以下に示す.

   - n より小さく, $\phi(n)$ と互いに素である整数を 1 つ選び, これを暗号鍵 e とする.

- $d = K^{-1}\ mod\ \phi(n)$ となる d を計算し, これを復号鍵とする.

ここで, $\phi(n)$ は Euler 関数といい, 「n 以下で n と互いに素である自然数の個数」を与える関数である.

生成した暗号鍵, 復号鍵を用いて鍵 K を生成する. K に対する暗号化関数 E, 復号関数 D はそれぞれ次のように定義される.

$$E(M) \equiv M^e (mod\ n)$$
$$D(C) \equiv C^d (mod\ n)$$

ここで M, C はそれぞれ平文と暗号文を表している.

2.3.2 RSA 暗号

2.2 節で示した金持ちの財産比べプロトコルで用いる公開鍵暗号として, RSA 暗号について説明する. RSA 暗号は, Rivest, Shamir, Adleman によって考案された最初の本格的な公開鍵暗号方式である [1, 2, 5]. RSA 暗号は素因数分解の困難さを安全性の根拠としている. RSA 暗号は信頼性の高い公開鍵暗号方式として幅広い分野に実用化されており, 事実上の標準とされている. 以下に RSA 暗号について述べる.

利用者は暗号鍵, 復号鍵を次のように生成する.

1. 2 つの大きな素数 p, q を選択し, これらの積 $n = pq$ を計算する.
2. $p-1$ と $q-1$ の最小公倍数 $L = LCM(p-1, q-1)$ を計算する.
3. 最大公約数 $GCD(e, L) = 1$ を満たす自然数 e を選び, $ed = 1(mod\ L)$ を満たす d を求め, e を暗号鍵, d を復号鍵とする.

生成した暗号鍵, 復号鍵を用いて鍵 K を生成する. K に対する暗号化関数 E, 復号関数 D はそれぞれ次のように定義される.

$$E(M) \equiv M^e (mod\ n)$$
$$D(C) \equiv C^d (mod\ n)$$

ここで M, C はそれぞれ平文と暗号文を表している. RSA 暗号は公開鍵暗号であり, 鍵 K のうち暗号化関数 E が全体へ公開されている.

3 ブラックジャックの設計

3.1 ブラックジャック

ブラックジャックは世界中で広く知られているカードゲームの 1 つである. ブラックジャックはディーラーとプレイヤーの 2 人の間で行われるゲームである. ブラックジャックは次のような手順で行われる.

1. 山札からディーラーとプレイヤーにカードが 2 枚ずつ配布される. この際, ディーラーのカードは 1 枚表向きで配られる.
2. プレイヤーはヒットまたはスタンドを行い, カードの合計を 21 に近づける. スタンドを行うか, もしくはカードの合計が 21 を超えるまでこの動作を繰り返す. カードの合計が 21 を超えた場合プレイヤーの負けでゲームが終了する.
3. プレイヤーがスタンドを行った場合に行われる. ディーラーはカードの合計が 17 以上になるまでカードを引く. カードの合計が 21 を超えるとディーラーの負けでゲームが終了する.
4. ディーラーとプレイヤーのカードの合計を比較し, 勝敗を決定する. カードの合計が 21 に近い方が勝利となる.

ここで 2 の操作で行われるヒットとは山札からカードを 1 枚引くことを表す. またスタンドとは, これ以上ヒットしないと宣言することである.

本稿で設計するブラックジャックは通常のブラックジャックとは少し異なるものとなる. その違いを以下に記す.

- ディーラーのカードは全て裏向きで配られる.
- ディーラーとプレイヤーのカードの合計を比較する際, ディーラーのカードは表向きにされない.

ディーラーに配布されたカードは全て裏向きで操作されるため, プレイヤーはディーラーのカードが何か分からない. なぜこのような違いを行うかについて説明する.

ブラックジャックには使われたカードの枚数を数えるカウンティングという技術が存在する. カウンティングは基本的な戦術ではなく, カジノでは禁止されている行為である. 本論文で上記のような違いを加えることによって, プレイヤーから見えるカードの枚数が少なくなるため, 通常の場合よりプレイヤーにカウンティングを行ないにくくしており, 自然にカウンティングを排除する効果が得られる. また, ヒットやスタンドの判断の際, 通常は見えているカードも伏せるため, 実際のブラックジャックよりも判断が難しくなる. 代わりにディーラーが何枚カードを引いたかをプレイヤーは知ることができるため, カードの枚数からディーラーの手の内を予測するといった楽しみも生まれ, ゲームが面白くなると期待できる.

以下にこのブラックジャックを行うためのプロトコルを述べる.

3.2 山札プロトコル

カードゲームにおいて, 伏せたカードを山にしたものを山札と呼ぶ. 山札プロトコルではメンタルポーカープロトコルを応用し, 山札の作成, カードの配布, 及び山札の公開を行うことができる.

3.2.1 山札の生成

今, n 人のプレイヤー間で, k 枚のカードから山札を作成するものとする. ここで k 毎のカードは, それぞれ $0, 1, \ldots, (k-1)$ の整数で表現し, プレイヤーはそれぞれ $U_0, U_1, \ldots, U_{n-1}$ で表現する. 全プレイヤーは 2.3.1 節で述べた暗号系に基づき鍵を生成する. プレイヤー $U_i (i = 0, 1, \ldots, n-1)$ の鍵をそれぞれ K_i で表す. また山札の中身は, 配列 m(要素数 k) に格納する. ここで山札の生成手順を以下に示す.

1. 初め, 配列 m には, $m[0] = 0, m[1] = 1, \ldots, m[k-1] = k-1$, の順序でカードが並んでいる.
2. プレイヤーは $U_0, U_1, \ldots, U_{n-1}$ の順番で, このカードをシャッフルする. プレイヤー 1 人分のシャッフルは以下の手順で行う.
   - プレイヤーは $m[0], m[1], \ldots, m[k-1]$ をそれぞれ自分の鍵で暗号化し, それらの順番を任意に入れ替える (シャッフル)
   - プレイヤーは上記の処理を行ったあとの配列 m を, 他の全プレイヤーに通知する.
3. 全プレイヤーのシャッフルが終わったら, 山札の生成が完了する. この時点で配列 m の各要素は, $0, 1, \ldots, (k-1)$ のうちいずれかの整数を $E_0(), E_1(), \ldots, E_{n-1}()$ の順で暗号化したものである.

上記の処理で得られる山札の中身は無作為な順番に並んでおり, 全てのプレイヤーが中身を分からない状態となっている. またカードの位置によってあらかじめ中身が決まっている.

3.2.2 カードの配布

山札よりプレイヤーにカードを配布する方法について述べる. 山札は 3.2.1 節で述べた山札プロトコルを用いて生成されたものとする. ここでカードの配布とは, プレイヤー $U_i (i = 0, 1, \ldots, n-1)$ が他のプレイヤーに知られないように山札 $m[a] (a = 0, 1, \ldots, k-1)$ の内容を知ることを指す. 以下にカードの配布を行う手順を示す.

1. プレイヤーたちは $U_{n-1}, \ldots, U_1, U_0$ の順番で $m[a]$ を復号していく. ただし, プレイヤー U_i 自身はこの復号に参加しない. U_i を除く全員の復号が終わると, 元のカードの値に $E_i()$ のみを適用したものが得られる.
2. U_i は, 1 の結果得られた値を自分の鍵で復号して, $m[a]$ に格納されたカードを知ることができる.

上記の操作で得られたカードの値は最後にプレイヤー U_i の鍵で復号を行うため, プレイヤー U_i 以外に知られることはない. これによりカードを中身が他のプレイヤーに知られること無く, 配布されたプレイヤーのみが分かるように配布することができる.

3.2.3 山札の公開

山札を公開する方法について述べる. 山札は 3.2.1 節で述べた山札プロトコルを用いて生成されたものとする. ゲームが行われた後, 山札を公開しプレイヤー全員で山札を確認することができる. このとき, ゲーム中の操作を全て記録しておくと, 実際に用いられたカードと公開された山札とを照らし合わせることでゲーム中の矛盾を発見することができる. この山札の公開方法を以下に示す.

1. 各プレイヤーは山札生成時に生成した自らの鍵を他のプレイヤーに通知する.
2. 各プレイヤーは受け取った鍵と自らの鍵をもとに最終的に作られた山札を復号する.

山札生成時にできた山札を全てのプレイヤーで共有しているため, 他のプレイヤーの鍵を受け取ることで各自で山札の確認を行うことができる.

3.3 勝敗判定プロトコル

勝敗判定プロトコルは, ブラックジャックを実装する際に, 金持ちの財産比べプロトコルを用いて勝敗を決定するプロトコルを表している. ディーラー A とプレイヤー B によってブラックジャックが行われるとする. 今ブラックジャックが行われ, A のカードの合計が x, B のカードの合計が y であるとする. A の公開鍵暗号を用いて金持ちの財産比べプロトコルを行うと x と y の大小関係が $y \leq x$, もしくは $y > x$ であることが分かる. このことを用いて以下に勝敗判定プロトコルを示す.

1. ディーラー A, プレイヤー B の手持ちカードの合計をそれぞれ x, y とする. ただし合計が 21 より大きい場合 0 とする.
2. A, B は A の公開鍵暗号を用いて金持ちの財産比べプロトコルを x, y について実行する.
3. 結果が $y \leq x$ の場合 A の勝ち, $y > x$ の場合 B の勝ちとなる. B は勝敗を A に送信する.

上記のプロトコルによりブラックジャックの勝敗を決定することができる. 金持ちの財産比べプロトコルを用いるため, A, B ともにカードの合計を公開せずに勝敗を決定することができる. そのため, A, B それぞれがカードを相手に公開する必要なくゲームを行うことができる. 金持ちの財産比べプロトコルを用いるため, A と B のカードの合計値 x, y が等しい場合, ディーラー A の勝ちとなる. 本来のブラックジャックではカードの合計が等しい場合引き分けとなるが, 本稿のブラックジャックでは引き分けとせず, ディーラーの勝ちとする. なお B の公開鍵暗号を用いて再度金持ちの財産比べプロトコルを行うことで, 引き分けかどうかの判定を行うことができる.

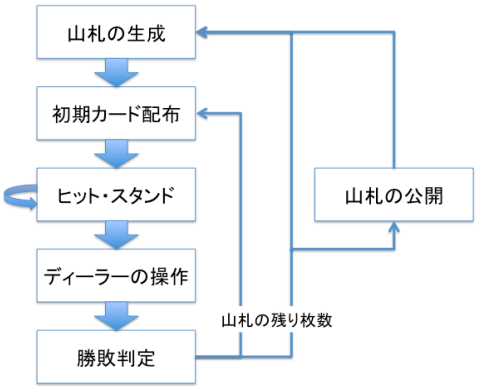

図 1: ブラックジャックの手順

3.4 新たなブラックジャックの手順

本節では, 実際に実装されるブラックジャックの手順を説明する. ブラックジャックは図 1 のような手順で行われる.

1. 山札の生成：この際, ディーラーとプレイヤーはそれぞれ必要な鍵を生成する.
2. 初期カードの配布：ディーラー, プレイヤーにそれぞれカードが 2 枚ずつ配られる. ただし, ディーラーのカードは裏向きで配られる.
3. ヒット · スタンド：プレイヤーがヒットもしくはスタンドを選択する.
4. ディーラーの操作：ディーラーはカードの合計が 17 以上になるまでカードを引く.
5. 勝敗判定：ディーラーとプレイヤーどちらが勝ったかを判定する.
6. 山札の公開：山札が再生成される際, プレイヤーが公開を選択すれば山札を公開する.

4 実装

本節では, 3.4 節で述べたブラックジャックの実装を行う. 開発言語としてプログラミング言語 Python 2.7.2 と Python のゲーム用ライブラリ Pygame を用いてブラックジャックを実装した.

以下にブラックジャックの各手順の操作について記述する. ただしゲームはディーラー A とプレイヤー B の 2 人で行われるものとする.

- 山札の生成
  ディーラー A は非常に大きい素数 n を用意し, プレイヤー B に伝える. n をもとに A, B はそれぞれ 2.3.1 節で述べた可換な暗号を生成する. 生成された暗号をそれぞれ K_A, K_B とする. また A は 2.3.2 節で述べた RSA 暗号を生成する. この暗号を K'_A とする. A, B は鍵 K_A, K_B を用い, 山札の生成プロトコルによって山札を生成する. また K'_A は公開鍵なので A は B に E'_A を伝える.
- 初期カードの配布
  ゲームの始めに山札の配布を行う. まずディーラー A が山札の最初の 2 枚を, カードの配布プロトコルを用いて引く. 次にプレイヤー B が, A が引いたカードの次の 2 枚を, カードの配布プロトコルを用いて引く.
- ヒット · スタンド
  初期カードの配布が行われると, プレイヤー B はヒットもしくはスタンドの操作を行う. まず B はヒットを行うかスタンドを行うかをディーラー A に送信する. ヒットを行う場合, B はカードの配布プロトコルを用いて山札よりカードを 1 枚引く. ヒットの操作が行われると B は合計が 21 を超えたかどうかを送信する. 21 を超えていた場合, 次の操作に移る. B がスタンドを選択するまでこの操作を繰り返す.
- ディーラーの操作
  プレイヤー B がカードを引き終わると, ディーラー A はカードの合計が 17 以上になるまでカードを引く. まず初期カードの合計を確認し, カードを引くかどうかを B に送信する. 合計が 17 未満のとき, A はカード配布プロトコルを用いて山札よりカードを 1 枚引く. A はカードの合計が 17 以上になるまでこの操作を繰り返す.
- 勝敗判定
  ディーラー A のカードの合計が 17 以上になった時, 勝敗判定を行う. 公開鍵 K'_A を用いて勝敗判定プロトコルを実行する. 勝敗が決まったあと, A は山札の残り枚数を確認し, 山札を再生成するかどうかを判断する.
- 山札の公開
  山札の再生成が行われる際, プレイヤー B は山札の公開を行うかどうかをディーラー A に送信する. 山札の公開を行う場合, 山札の公開プロトコルを用いて山札の公開を行う.

実際に実装したブラックジャックのゲーム画面を図 2, 3 に示す. 図 2 は山札を生成し, 初期カードが配布された状態を表す. プレイヤーは Hit もしくは Stand ボタンを押してヒットもしくはスタンドを行う. 図 3 はプレイヤーがスタンドを行い, 山札の再生成が行われる際の図である. ここでは, 勝敗が左下に表示されている. またディーラーのカードは公開されず, プレイヤーはディーラーが 4 枚のカードを引いたことを確認することができる. また, CHECK!の部分をプレイヤーが押すとこれまで用いられた山札が公開される.

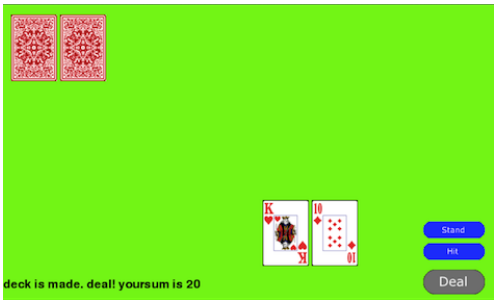

図 2: ゲームの開始

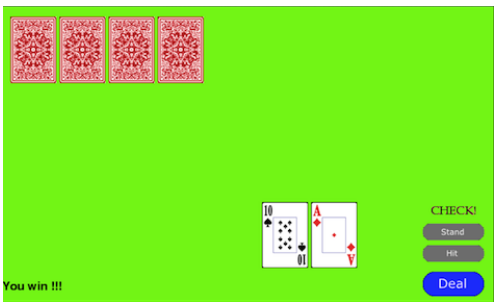

図 3: 山札の確認

5　おわりに

本稿では, マジックプロトコルとして代表的なメンタルポーカープロトコル, および金持ちの財産比べプロトコルを用いて全てのプレイヤーが中身を知ることができない山札の生成プロトコル, カードを配られた本人しかカードの中身を知ることができないカード配布プロトコル, 使用された山札の中身を確認する山札公開プロトコル, および互いの情報を公開せずに勝敗を判定する勝敗判定プロトコルを設計し, 実際にブラックジャックについて実装を行った.

本稿で実装したブラックジャックではディーラーは機械的な操作しか行っていないが, ディーラーにプレイヤーと同じ動きをさせることで対戦型ブラックジャックの実装が可能であると考えられる.

参考文献

[1] 太田和男, 黒澤馨, 渡辺治：情報セキュリティの科学, 講談社 (1995).

[2] Shamir, A., Rivest, R., and Adleman, L.：*Mental poker*, The Mathematical Gardner, pp 37-43 (1981).

[3] 岡本龍明, 山本博資：現代暗号, 産業図書, pp.106-130, 223-238 (1997).

[4] Yao, A.：*Protocols for Secure Computations (extended abstract)*, Proceedings of FOCS'82, IEEE Computer Society, pp 160-164 (1982).

[5] Rivest, R., Shamir, A., and Adleman, L.：*A Method for Obtaining Digital Signatures and PublicKey Cryptosystems*, Communications of the ACM, 21(2), pp 120-126 (1978).