A non-cooperative Pareto-efficient solution to a single-shot Prisoner's Dilemma

Wu, Haoyang

5 April 2011

A non-cooperative Pareto-efficient solution to a single-shot Prisoner's Dilemma

Haoyang Wu *

Abstract

The Prisoner's Dilemma is a simple model that captures the essential contradiction between individual rationality and global rationality. Although the single-shot Prisoner's Dilemma is usually viewed simple, in this paper we will propose an algorithmic model and a non-binding scheme to help non-cooperative agents obtain Pareto-efficient payoffs self-enforcingly. The scheme stems from quantum game theory, but is applicable to the macro world immediately.

Key words: Quantum game theory; Prisoner's Dilemma.

1 Introduction

The Prisoner's Dilemma (PD) is perhaps the most famous model in the field of game theory. In the original version of PD, two prisoners are arrested by a policeman. Each prisoner must independently choose a strategy between "Confessing" (denoted as strategy "Defect") and "Not confessing" (denoted as strategy "Cooperate"). The payoff matrix of PD is shown in Table 1. It has been known that even if two agents can communicate and sign a non-binding contract to cooperate before moving, the contract cannot be self-enforcing. As long as two agents are rational, the unique Nash equilibrium shall be (Defect, Defect), which results in a Pareto-inefficient payoff (P, P). That is the dilemma.

Table 1: The payoff matrix of PD, where $T > R > P > S$, and $R > (T+S)/2$. *The first entry in the parenthesis denotes the payoff of agent 1 and the second*

* Wan-Dou-Miao Research Lab, Shanghai, 200051, China.
Email addresses: hywch@mail.xjtu.edu.cn, Tel: 86-18621753457 (Haoyang Wu).

stands for the payoff of agent 2.

agent 2 / agent 1	Cooperate	Defect
Cooperate	(R, R)	(S, T)
Defect	(T, S)	(P, P)

Roughly speaking, there are two sorts of PD: single-shot PD and iterated PD. Nowadays a lot of studies on PD are focused on the latter case. For example, Axelrod [1] investigated the evolution of cooperative behavior in well-mixed populations of selfish agents by using PD as a paradigm. Nowak and May [2] induced a spatial structure in PD, i.e., agents were restricted to interact with his immediate neighbors. Santos and Pacheco [3] found that when agents interacted following scale-free networks, cooperation would become a dominating trait throughout the entire range of parameters of PD. Perc and Szolnoki [4] proposed that social diversity could induce cooperation as the dominating trait throughout the entire range of parameters of PD.

Compared with the iterated PD, the single-shot PD is usually viewed simple. In 1999, Eisert *et al* [5] proposed a quantum-version model of PD. The model showed "quantum advantages" as a result of a novel quantum Nash equilibrium, which help agents reach the Pareto-efficient payoff (R, R). Hence, the agents escape the dilemma. In 2002, Du *et al* [6] gave an experiment to carry out the quantum PD.

In Eisert *et al*'s model, there exists an arbitrator and he must perform quantum measurements to readout the messages of agents. This demand is somewhat not practical for macro disciplines such as politics and economics, because the arbitrator should play a neutral role in the game: His reasonable actions should be to receive agents' strategies and assign payoffs to agents according to Table 1. If he is willing to work using an additional quantum equipment which lets agents to obtain Pareto-efficient payoffs (R, R), then why does not he directly assign the Pareto-efficient payoffs to the agents?

In order to make Eisert *et al*'s model more practical, in Section 2 we will give an amended version of Eisert *et al*'s model. In Section 3, we will propose an algorithmic model to simulate the amended model. Section 4 is the main part of this paper, where we will categorize the single-shot PD into five types, and propose a non-binding scheme to help non-cooperative agents obtain Pareto-efficient payoffs self-enforcingly. Section 5 gives some discussions. The last section draws conclusion.

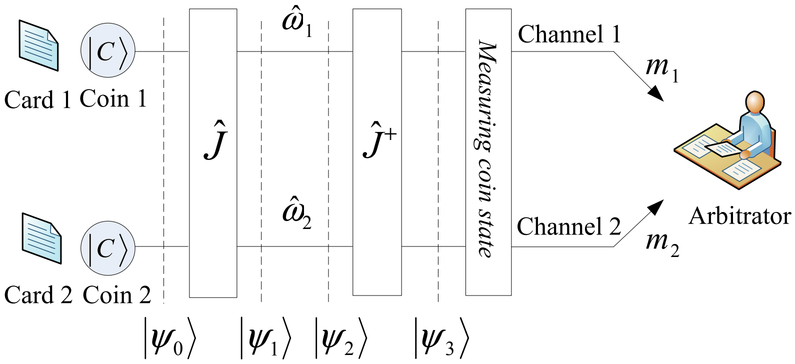

Fig. 1. The amended version of Eisert *et al*'s model . Each agent has a quantum coin and a classical card. Each agent independently performs a local unitary operation on his/her own quantum coin.

2 An amended version of Eisert *et al*'s model

Let the set of two agents be denoted as $N = \{1, 2\}$. Following formula (4) in Ref. [7] and Ref. [8], two-parameter quantum strategies are drawn from the set:

$$\hat{\omega}(\theta, \phi) = \begin{bmatrix} e^{i\phi}\cos(\theta/2) & i\sin(\theta/2) \\ i\sin(\theta/2) & e^{-i\phi}\cos(\theta/2) \end{bmatrix},$$

$\hat{\Omega} = \{\hat{\omega}(\theta, \phi) : \theta \in [0, \pi], \phi \in [0, \pi/2]\}$, $\hat{J} = \cos(\gamma/2)\hat{I} \otimes \hat{I} + i\sin(\gamma/2)\hat{\sigma_x} \otimes \hat{\sigma_x}$ (where γ is an entanglement measure), $\hat{I} = \hat{\omega}(0, 0)$, $\hat{D} = \hat{\omega}(\pi, \pi/2)$, $\hat{C} = \hat{\omega}(0, \pi/2)$.

Without loss of generality, we assume:
1) Each agent $j \in N$ has a quantum coin j (qubit) and a classical card j connected to the arbitrator. The basis vectors $|C\rangle = (1, 0)^T$, $|D\rangle = (0, 1)^T$ of a quantum coin denote head up and tail up respectively.
2) Each agent $j \in N$ independently performs a local unitary operation on his/her own quantum coin. The set of agent j's operation is $\hat{\Omega}_j = \hat{\Omega}$. A strategic operation chosen by agent j is denoted as $\hat{\omega}_j \in \hat{\Omega}_j$. If $\hat{\omega}_j = \hat{I}$, then $\hat{\omega}_j(|C\rangle) = |C\rangle$, $\hat{\omega}_j(|D\rangle) = |D\rangle$; If $\hat{\omega}_j = \hat{D}$, then $\hat{\omega}_j(|C\rangle) = |D\rangle$, $\hat{\omega}_j(|D\rangle) = |C\rangle$. $\hat{I}$ denotes "*Not flip*", $\hat{D}$ denotes "*Flip*".
3) The two sides of a card are denoted as Side 0 and Side 1. The messages written on the Side 0 (or Side 1) of card j is denoted as $card(j, 0)$ (or $card(j, 1)$). $card(j, 0)$ represents "Cooperation", and $card(j, 1)$ represents "Defect".
4) There is a device that can measure the state of two quantum coins and send messages to the designer.

Fig. 1 shows the amended version of Eisert *et al*'s model. Its working steps are shown as follows:
Step 1: The state of each quantum coin is set as $|C\rangle$. The initial state of the

two quantum coins is $|\psi_0\rangle = |CC\rangle$.
Step 2: Let the two quantum coins be entangled by $\hat{J}$. $|\psi_1\rangle = \hat{J}|CC\rangle$.
Step 3: Each agent j independently performs a local unitary operation $\hat{\omega}_j$ on his/her own quantum coin. $|\psi_2\rangle = [\hat{\omega}_1 \otimes \hat{\omega}_2]\hat{J}|CC\rangle$.
Step 4: Let the two quantum coins be disentangled by $\hat{J}^{+}$. $|\psi_3\rangle = \hat{J}^{+}[\hat{\omega}_1 \otimes \hat{\omega}_2]\hat{J}|CC\rangle$.
Step 5: The device measures the state of the two quantum coins and sends $card(j,0)$ (or $card(j,1)$) as the message m_j to the arbitrator if the state of quantum coin j is $|C\rangle$ (or $|D\rangle$).
Step 8: The arbitrator receives the overall message $m = (m_1, m_2)$ and assigns payoffs to the two agents according to Table 1. END.

Obviously, in the amended model the arbitrator does not need to work on an additional quantum equipment as Eisert *et al*'s model requires. Thus, the amended model is more suitable for macro applications.

3 An algorithmic version of the amended Eisert *et al*'s model

3.1 Matrix representations of quantum states

In quantum mechanics, a quantum state can be described as a vector. For a two-level system, there are two basis vectors: $(1,0)^T$ and $(0,1)^T$. In the beginning, we define:

$$|C\rangle = [1,0]^T,\ |D\rangle = [0,1]^T,\ |CC\rangle = [1,0,0,0]^T,$$

$$\hat{J} = \begin{bmatrix} \cos(\gamma/2) & 0 & 0 & i\sin(\gamma/2) \\ 0 & \cos(\gamma/2) & i\sin(\gamma/2) & 0 \\ 0 & i\sin(\gamma/2) & \cos(\gamma/2) & 0 \\ i\sin(\gamma/2) & 0 & 0 & \cos(\gamma/2) \end{bmatrix},\ \gamma \in [0, \pi/2].$$

For $\gamma = \pi/2$,

$$\hat{J}_{\pi/2} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & i \\ 0 & 1 & i & 0 \\ 0 & i & 1 & 0 \\ i & 0 & 0 & 1 \end{bmatrix},\ \hat{J}^{+}_{\pi/2} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & -i \\ 0 & 1 & -i & 0 \\ 0 & -i & 1 & 0 \\ -i & 0 & 0 & 1 \end{bmatrix}.$$

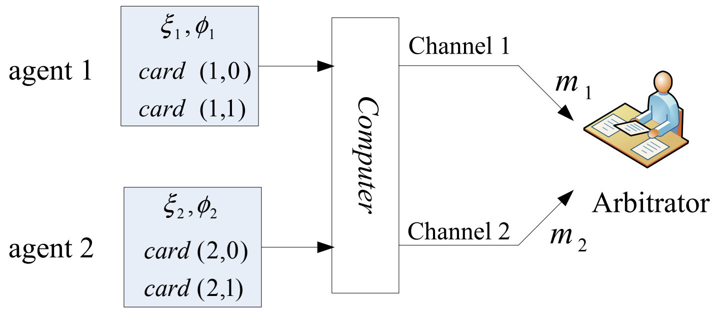

Fig. 2. The inputs and outputs of the algorithmic model.

$$|\psi_1\rangle = \hat{J}|CC\rangle = \begin{bmatrix} \cos(\gamma/2) \\ 0 \\ 0 \\ i\sin(\gamma/2) \end{bmatrix}$$

Following Ref. [11], since only two values in $|\psi_1\rangle$ are non-zero, we only need to calculate the leftmost and rightmost column of $\hat{\omega}_1 \otimes \hat{\omega}_2$ to derive $|\psi_2\rangle = [\hat{\omega}_1 \otimes \hat{\omega}_2]|\psi_1\rangle$.

3.2 An algorithmic model

Following Ref. [11], here we will propose an algorithmic model that simulates the amended Eisert *et al*'s model. Since the entanglement measurement γ is a control factor, it can be simply set as its maximum $\pi/2$. The inputs and outputs of the algorithmic model are shown in Fig. 2. The *Matlab* program is shown in Fig. 3(a)-(d).

Inputs:
1) ξ_j, ϕ_j, $j = 1, 2$: the parameters of agent j's local operation $\hat{\omega}_j$, $\xi_j \in [0, \pi]$, $\phi_j \in [0, \pi/2]$.
2) $card(j, 0), card(j, 1)$, $j = 1, 2$: the messages written on the two sides of agent j's card.

Outputs:
$m_j \in \{card(j, 0), card(j, 1)\}$, $j = 1, 2$: agent j's message that is sent to the arbitrator.

Procedures of the algorithmic model:
Step 1: The computer checks the number of channels that it controls.
Step 2: If only one channel is available to the computer (suppose the channel belongs to agent j), then the computer sends $card(j, 1)$ as message m_j to the

arbitrator. END.
Step 3: Reading two parameters ξ_j and ϕ_j from each agent j (See Fig. 3(a)).
Step 4: Computing the leftmost and rightmost columns of $\hat{\omega}_1 \otimes \hat{\omega}_2$ (See Fig. 3(b)).
Step 5: Computing the vector representation of $|\psi_2\rangle = [\hat{\omega}_1 \otimes \hat{\omega}_2]\hat{J}_{\pi/2}|CC\rangle$.
Step 6: Computing the vector representation of $|\psi_3\rangle = \hat{J}^{+}_{\pi/2}|\psi_2\rangle$.
Step 7: Computing the probability distribution $\langle\psi_3|\psi_3\rangle$ (See Fig. 3(c)).
Step 8: Randomly choosing a "collapsed" state from the set of all four possible states $\{|CC\rangle, |CD\rangle, |DC\rangle, |DD\rangle\}$ according to the probability distribution $\langle\psi_3|\psi_3\rangle$.
Step 9: For each $j \in I$, the computer sends $card(j, 0)$ (or $card(j, 1)$) as message m_j to the arbitrator if the j-th basis vector of the "collapsed" state is $|C\rangle$ (or $|D\rangle$) (See Fig. 3(d)).

4 Five types of PD

Up to now, PD has been generalized to many disciplines such as politics, economics, sociology, biology and so on. Despite these widespread applications, people usually do not care how agents obtain their payoffs. For example, Axelrod [1] used the word "yield" to describe how the agents obtained the payoffs. Nowak and May [2] used the word "get", and Santos and Pacheco [3] used the word "receive" respectively. One may think that such question looks trivial at first sight. However, as we will show in this section, there is an interesting story behind this question. In what follows, we will categorize the PD into five different types.

Type-1 PD:
1) There are two agents and no arbitrator in the game.
2) The strategies of agents are actions performed by agents. The agents' payoffs are determined by the outcomes of these actions and satisfy Table 1.
For example, let us neglect the United Nation and consider two countries (e.g., US and Russia) confronted the problem of nuclear disarmament. The strategy C means "Obeying disarmament", and D means "Refusing disarmament". If the payoff matrix confronted by the two countries satisfies Table 1, the nuclear disarmament game is a type-1 PD.

Type-2 PD:
1) There are two agents and an arbitrator in the game.
2) The strategies of agents are actions performed by agents. The arbitrator observes the outcomes of actions and assign payoffs to the agents according to Table 1.
For example, let us consider a taxi game. Suppose there are two taxi drivers and a manager. Two drivers drive a car in turn, one in day and the other

in night. The car's status will be very good, ok or common if the number of drivers who maintain the car is two, one or zero. The manager observes the car's status and assigns rewards R_2, R_1, R_0 to each driver respectively, where $R_2 > R_1 > R_0$. The whole cost of maintenance is c. Let the strategy C denote "Maintain", and D denote "Not maintain". The payoff matrix can be represented as Table 2. If Table 2 satisfies the conditions in Table 1, the taxi game is a type-2 PD.

Table 2: The payoff matrix of type-2 PD.

agent 1 \ agent 2	C	D
C	$(R_2 - c/2, R_2 - c/2)$	$(R_1 - c, R_1)$
D	$(R_1, R_1 - c)$	(R_0, R_0)

Type-3 PD:
1) There are two agents and an arbitrator in the game.
2) The strategy of each agent is not an action, but a message that can be sent to the arbitrator through a channel. The arbitrator receives two messages and assign payoffs to the agents according to Table 1.
3) Two agents cannot communicate with each other.
For example, suppose two agents are arrested separately and required to report their crime information to the arbitrator through two channels respectively. If the arbitrator assigns payoffs to agents according to Table 1, this game is a type-3 PD.

Type-4 PD:
Conditions 1-2 are the same as those in type-3 PD.
3) Two agents can communicate with each other.
4) Before sending messages to the arbitrator, two agents can construct a non-binding scheme (We will show that this scheme is self-enforcing), which specifies a dynamic game with three stages:
Stage 1 (Initialization): Two agents design an algorithmic model specified in Section 3.2.
Stage 2: (Actions of two agents) For each agent $j \in N$, he independently faces two strategies:
$S(j, 0)$: Participate the algorithmic model, i.e., leave his channel to the computer, and submit $\xi_j, \phi_j, card(j, 0), card(j, 1)$ to the computer.
$S(j, 1)$: Not participate the algorithmic model, i.e., take back his channel, and submit a message m_j to the arbitrator directly.
Stage 3: (Actions of the arbitrator) The arbitrator receives two messages and assigns payoffs to agents according to Table 1.

The algorithmic model is triggered if at least one agent transfers his channel to the computer. From the viewpoints of the arbitrator, nothing is changed.

However, the payoff matrix confronted by two agents is now changed to Table 3. For each entry of Table 3, we give the corresponding explanation as follows:
1) Strategy $(S(1,0), S(2,0))$: This strategy means two agents both participate the algorithmic model and submit parameters to the computer. According to Ref. [11], for each agent $j \in N$, his dominant parameters are $\xi_j = 0$ and $\phi_j = \pi/2$, which result in a Pareto-efficient payoff (R, R).
2) Strategy $(S(1,0), S(2,1))$: This strategy means agent 1 participates the algorithmic model, but agent 2 takes back his channel and submits a message to the arbitrator directly. Obviously, the unique rational message sent by agent 2 is $m_2 = card(2,1)$. On the other hand, the algorithmic model finds that only one channel (transferred from agent 1) is available, so the computer will send $m_1 = card(1,1)$ to the arbitrator. As a result, the arbitrator will assign (P, P) to two agents according to Table 1.
3) Strategy $(S(1,1), S(2,0))$: This case is similar to the above case. The arbitrator will assign (P, P) to two agents.
4) Strategy $(S(1,1), S(2,1))$: This strategy means two agents both take back their channels and send messages to the arbitrator directly. Obviously, the dominant message sent by each agent j is $card(j,1)$, and the arbitrator will assign the Pareto-inefficient payoff (P, P) to agents.

Table 3: The payoff matrix of two agents by using the non-binding scheme, where R, P are defined in Table 1, $R > P$.

agent 1 \ agent 2	$S(2,0)$	$S(2,1)$
$S(1,0)$	(R, R)	(P, P)
$S(1,1)$	(P, P)	(P, P)

From Table 3, it can be seen that $(S(1,0), S(2,0))$ and $(S(1,1), S(2,1))$ are two Nash equilibria, and the former is Pareto-efficient. Since two channels have been controlled by the computer in Stage 1, in the end the Pareto-efficient payoff (R, R) is self-enforcing. In this sense, the two agents escape the dilemma.

Type-5 PD:
Conditions 1-3 are the same as those in type-4 PD.
4) The last condition of type-4 PD does not hold.

5 Discussions

The scheme revises common understanding on the Prisoner's Dilemma. Some readers may doubt its justification. We will discuss some possible doubts as follows.

Q1: The scheme seems to be cooperative because in Stage 1, two agents must agree to construct an algorithmic model, which acts as a correlation between agents.
A1: As we have pointed out, from the viewpoints of the arbitrator, nothing is changed. Thus, the so-called correlation between two agents is indeed unobservable to the arbitrator. Put differently, the arbitrator cannot prevent agents from constructing such algorithmic model. On the other hand, since each agent can freely not participate the algorithmic model when he chooses a strategy in Stage 2, the scheme is non-cooperative.

Q2: If the algorithmic model finds that two channels are available, can it simply send $(card(1,0), card(2,0)$ to the arbitrator instead of running Steps 3-9?
A2: The algorithmic model enlarges the strategy space of each agent from a one-dimensional strategy space $[0,1]$ to a two-dimensional strategy space $[0,\pi]\times[0,\pi/2]$, and generates the Pareto-efficient payoff (R,R) in Nash equilibrium. The new game is non-cooperative. However, the idea in Question 3 restricts the strategy space of each agent from a one-dimensional strategy space $[0,1]$ to a single point $(0,\pi/2)$ in the two-dimensional space $[0,\pi]\times[0,\pi/2]$. In this sense, two agents are required to cooperate and make commitments to do so. This is beyond the range of non-cooperative game.

Remark: The type-1 and type-2 PD are not suitable for the scheme, because the computer cannot perform actions on behalf of agents. The type-3 PD is not suitable either because two agents are separated, therefore no common device can send messages to the arbitrator on behalf of them. The type-5 PD is not suitable for the scheme because Condition 4 in type-4 PD is vital and indispensable for the scheme.

6 Conclusion

In this paper, we categorize the well-known PD into five types and propose a non-binding scheme to help agents escape a special type of PD, i.e., the type-4 PD. One point is important for the novel result: Usually people think the payoff matrices confronted by agents and the arbitrator are the same (i.e., Table 1). However we argue that for the case of type-4 PD, the two payoff matrices can be different: The arbitrator still faces Table 1, but the agents can self-enforcingly change their payoff matrix to Table 3 by virtue of the non-binding scheme, which leads to a Pareto-efficient payoff.

Acknowledgments

The author is very grateful to Ms. Fang Chen, Hanyue Wu (*Apple*), Hanxing Wu (*Lily*) and Hanchen Wu (*Cindy*) for their great support.

References

[1] R. Axelrod, W.D. Hamilton, The evolution of cooperation, Science, 211 (1981) 1390-1396.

[2] M.A. Nowak, R.M. May, Evolutionary games and spatial chaos, Nature, 359 (1992) 826-829.

[3] F.C. Santos, J.M. Pacheco, Scale-free networks provide a unifying framework for the emergence of cooperation, Phys. Rev. Lett., 95 (2005) 098104.

[4] M. Perc and A. Szolnoki, Social diversity and promotion of cooperation in the spatial prisoner's dilemma game, Phys. Rev. E, 77 (2008) 011904.

[5] J. Eisert, M. Wilkens, M. Lewenstein, Quantum games and quantum strategies, Phys. Rev. Lett., 83 (1999) 3077-3080.

[6] J. Du, H. Li, X. Xu, M. Shi, J. Wu, X. Zhou and R. Han, Experimental realization of quantum games on a quantum computer, Phys. Rev. Lett., 88 (2002) 137902.

[7] A.P. Flitney and L.C.L. Hollenberg, Nash equilibria in quantum games with generalized two-parameter strategies, *Phys. Lett. A* **363** (2007) 381-388.

[8] H. Wu, Quantum mechanism helps agents combat "bad" social choice rules. *International Journal of Quantum Information*, 2010 (accepted). http://arxiv.org/abs/1002.4294

[9] S. J. van Enk and R. Pike, Classical rules in quantum games, *Phys. Rev. A* **66** (2002) 024306.

[10] T.D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe and J.L. O'Brien, Quantum computers, Nature, 464 (2010) 45-53.

[11] H. Wu, On amending the sufficient conditions for Nash implementation. *Theoretical Computer Science*, 2011 (under review). http://arxiv.org/abs/1004.5327

```
%***********************************************************
%     Step 3 – Step 9 of the algorithmic model
%***********************************************************

% Defining the array of (ξ_j, φ_j), j=1,2
xi=zeros(2,1);
phi=zeros(2,1);

% Reading agent 1's parameters (ξ_1, φ_1). For example, ω̂_1 = Ĉ = ω̂(0, π/2)
xi(1)=0;
phi(1)=pi/2;

% Reading agent 2's parameters (ξ_2, φ_2). For example, ω̂_2 = Ĉ = ω̂(0, π/2)
xi(2)=0;
phi(2)=pi/2;
```

Fig. 3 (a). Reading each agent *j*'s parameters ξ_j and ϕ_j, $j = 1,2$.

```
% Defining two 2*2 matrices A and B
A=zeros(2,2);
B=zeros(2,2);

% Let A represents the local operation ω̂_1 of agent 1.
A(1,1)=exp(i*phi(1))*cos(xi(1)/2);
A(1,2)=i*sin(xi(1)/2);
A(2,1)=A(1,2);
A(2,2)=exp(-i*phi(1))*cos(xi(1)/2);

% Let B represents the local operation ω̂_2 of agent 2.
B(1,1)=exp(i*phi(2))*cos(xi(2)/2);
B(1,2)=i*sin(xi(2)/2);
B(2,1)=B(1,2);
B(2,2)=exp(-i*phi(2))*cos(xi(2)/2);

% Computing the leftmost and rightmost columns of ω̂_1 ⊗ ω̂_2
C=zeros(4, 2);
for row=1 : 2
        C((row-1)*2+1, 1) = A(row,1) * B(1,1);
        C((row-1)*2+2, 1) = A(row,1) * B(2,1);
        C((row-1)*2+1, 2) = A(row,2) * B(1,2);
        C((row-1)*2+2, 2) = A(row,2) * B(2,2);
end
A=C;

% Now the matrix A contains the leftmost and rightmost columns of ω̂_1 ⊗ ω̂_2
```

Fig. 3 (b). Computing the leftmost and rightmost columns of $\hat{\omega}_1 \otimes \hat{\omega}_2$

```
% Computing |ψ2⟩ = [ω̂1 ⊗ ω̂2] Ĵπ/2 |CC⟩
psi2=zeros(4,1);
for row=1 : 4
        psi2(row)=(A(row,1)+A(row,2)*i)/sqrt(2);
end

% Computing |ψ3⟩ = Ĵ+π/2 |ψ2⟩
psi3=zeros(4,1);
for row=1 : 4
        psi3(row)=(psi2(row) - i*psi2(5-row))/sqrt(2);
end

% Computing the probability distribution ⟨ψ3|ψ3⟩
distribution=psi3.*conj(psi3);
distribution=distribution./sum(distribution);
```

Fig. 3 (c). Computing $|\psi_2\rangle, |\psi_3\rangle, \langle\psi_3|\psi_3\rangle$.

```
% Randomly choosing a “collapsed” state according to the probability distribution ⟨ψ3|ψ3⟩
random_number=rand;
temp=0;
for index=1: 4
   temp = temp + distribution(index);
   if temp >= random_number
      break;
   end
end

% indexstr: a binary representation of the index of the collapsed state
%   ‘0’ stands for |C⟩, ‘1’ stands for |D⟩
indexstr=dec2bin(index-1);
sizeofindexstr=size(indexstr);

% Defining an array of messages for two agents
message=cell(2,1);

% For each agent j∈N, the algorithmic model generates the message m_j
for index=1 : 2 - sizeofindexstr(2)
   message{index,1}=strcat('card(',int2str(index),',0)');
end
for index=1 : sizeofindexstr(2)
   if indexstr(index)=='0'      % Note: ‘0’ stands for |C⟩
      message{2-sizeofindexstr(2)+index,1}=strcat('card(',int2str(2-sizeofindexstr(2)+index),',0)');
   else
      message{2-sizeofindexstr(2)+index,1}=strcat('card(',int2str(2-sizeofindexstr(2)+index),',1)');
   end
end

% The algorithmic model outputs the messages m1,m2 to the arbitrator
for index=1:2
   disp(message(index));
end
```

Fig. 3 (d). Computing the messages m_1, m_2 that agents submit to the arbitrator.