Amoeba-inspired algorithm for cognitive medium access

Song-Ju Kim and Masashi Aono

1 WPI Center for Materials Nanoarchitectonics (MANA), National Institute for Materials Science (NIMS)
1-1 Namiki, Tsukuba, Ibaraki 305-0044, Japan
2 Earth-Life Science Institute, Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152-8550, Japan
3 PRESTO, Japan Science and Technology Agency
4-1-8 Honcho, Kawaguchi-shi, Saitama 332-0012, Japan

a) KIM.Songju@nims.go.jp

Received July 24, 2013; Revised November 25, 2013; Published April 1, 2014

Abstract: The “tug-of-war (TOW) model” is a unique parallel search algorithm for solving the multi-armed bandit problem (BP), which was inspired by the photoavoidance behavior of a single-celled amoeboid organism, the true slime mold Physarum polycephalum [1–4]. “The cognitive medium access (CMA) problem,” which refers to multiuser channel allocations of the cognitive radio, can be interpreted as a “competitive multi-armed bandit problem (CBP) [5, 6].” Unlike the normal BP, the CBP considers a competitive situation in which more than one user selects a channel whose reward probability (probability of which channel is free) varies depending on the number and combination of the selecting users as indicated in a payoff matrix. Depending on the payoff matrix, the CBP provides a hard problem instance in which the users should not be attracted to the Nash equilibrium to achieve the “social maximum,” which is the most desirable state to obtain the maximum total score (throughput) for all the users. In this study, we propose two variants of the TOW model (solid type and liquid type) for the CBP toward developing a CMA protocol using a distributed control in uncertain environments. Using the minimum CBP cases where both the users choose a channel from the two considered channels, we show that the performance of our solid-type TOW model is better than that of the well-known upper confidence bound 1 (UCB1)-tuned algorithm, particularly for the hard problem instances. The aim of this study is to explore how the users can achieve the social maximum in a decentralized manner. We also show that our liquid-type TOW model, which introduces direct interactions among the users for avoiding mutual collisions, makes it possible to achieve the social maximum for general CBP instances.

Key Words: cognitive radio, medium access control, multi-armed bandit problem, bio-inspired computing, parallel search, machine learning
1. Introduction

Recently, various biologically inspired computing algorithms, such as ant colony optimization [7] and bee colony optimization [8], have been studied actively. In this study, we were inspired by a unicellular amoeboid organism, the plasmodium of the true slime mold *Physarum polycephalum*, that exhibits rich spatiotemporal oscillatory behavior and sophisticated computational capabilities [9]. We are interested in how the volume conservation law affects the information processing capabilities of the amoeba, and developed a tug-of-war (TOW) model [1–4, 10].

In the TOW model, a number of branches of the amoeba act as search agents to collect information on light stimuli while conserving the total sum of their resources. The resource conservation law produces nonlocally correlated search movements of the branches. We showed that the nonlocal correlation can be advantageous to manage the “exploration–exploitation dilemma,” which is the trade-off between the accuracy and speed in solving the “multi-armed bandit problem (BP).” In this study, we concentrate on the minimal instances of the BP, i.e., two-armed cases, stated as follows: Consider two slot machines. Both machines have individual reward probabilities $P_A$ and $P_B$. At each trial, a player selects one of the machines and obtains some reward, for example, a coin, with the corresponding probability. The player wants to maximize the total reward sum obtained after a certain number of selections. However, it is supposed that the player does not know these probabilities. The problem is to determine the optimal strategy for selecting the machine that yields maximum rewards by referring to past experiences.

In our previous studies [1–4, 11–17], we showed that the TOW model is more efficient than other well-known algorithms such as the modified $\epsilon$-greedy algorithm and the modified softmax algorithm, and is comparable to the “upper confidence bound 1-tuned (UCBIT) algorithm,” which is known as the best algorithm among non-parameter algorithms [18]. The algorithms for solving the problem are applicable to various fields, such as the Monte-Carlo tree search, which is used in algorithms for the game of GO [19, 20], the cognitive radio [5, 6], and web advertising [21].

In this paper, we present our algorithm that is applied to the cognitive radio and compare the performances of our TOW model and the UCBIT algorithm. Recently, the “cognitive medium access (CMA)” problem is one of the hottest topic in the field of mobile communications [5, 6]. The underlying idea is to allow unlicensed users (i.e., cognitive users) to access the available spectrum when the licensed users (i.e., primary users) are not active. The CMA is a new medium access paradigm in which the cognitive users are not allowed to interfere with the licensed users.

Figure 1 shows the channel model proposed by Lai et al. [5, 6]. There is a primary network consisting of N channels, each with bandwidth B. The users in the primary network are operated in a synchronous time-slotted fashion. It is assumed that at each time slot, channel $i$ is free with probability $P_i$. The cognitive users do not know $P_i$ a priori. At each time slot, the cognitive users attempt to exploit the availability of channels in the primary network by sensing the activity in this channel model. In this setting, a single cognitive user can access only a single channel at any given time. The problem is to derive an optimal access strategy for choosing channels that maximize the expected throughput obtained by the cognitive user. This situation can be interpreted as “the multiuser competitive bandit
problem (CBP)."

We consider the minimum CBP, i.e., 2 cognitive (unlicensed) users (1 and 2) and 2 channels (A and B). Each channel is not occupied by primary (licensed) users with the probability \( P_i \). In the BP context, we assume that the user accessing a free channel can get some reward, for example a coin, with the probability \( P_i \). Table I shows the payoff matrix for users 1 and 2. If two cognitive users select the same channel, i.e., the collision occurs, the reward is evenly split between the selecting users. The CBP is to determine the optimal strategy for selecting the machine that yields the maximum total reward of all users, which is called “social maximum.”

<table>
<thead>
<tr>
<th></th>
<th>user 2: A</th>
<th>user 2: B</th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1: A</td>
<td>( P_A/2 ) (( P_A/2 ))</td>
<td>( P_A ) (( P_B ))</td>
</tr>
<tr>
<td>user 1: B</td>
<td>( P_B ) (( P_A ))</td>
<td>( P_B/2 ) (( P_B/2 ))</td>
</tr>
</tbody>
</table>

In order to develop a unified framework for the design of efficient, low-complexity, and distributed-control, CMA protocols, we have to develop an algorithm that can obtain the maximum total reward (score) in this context. We report the results for the performance of the TOW model and the UCB1T algorithm as a candidate for the CMA in this study.

2. Model

2.1 Advantage of the TOW model’s learning rule

The learning rule of the TOW model is given by the following \( Q_j(t) \) (\( j \in \{A, B\} \)),

\[
Q_j(t) = N_j - (1 + \omega) L_j,
\]

where \( N_j \) and \( L_j \) denote the numbers (accumulated counts) of selections and light stimulations (no coin) of branch (machine) \( j \) until time \( t \), respectively [11, 12]. Here, \( \omega \) represents the weight parameter to be discussed later.

In this subsection, the learning rules of the TOW model are derived from radical discussion, and we can obtain the almost optimal weight parameter \( \omega_0 \). In many popular algorithms such as the \( \epsilon \)-greedy algorithm, an estimate for the reward probability is updated only in a selected arm. In contrast, we consider the case in which the sum of the reward probabilities \( \gamma = P_A + P_B \) is given. Then, we can update both the estimates simultaneously as follows:

\[
A: \quad \frac{N_A - L_A}{N_A} \quad B: \quad \frac{N_A - L_A}{N_A},
\]

\[
A: \quad \frac{N_B - L_B}{N_B} \quad B: \quad \frac{N_B - L_B}{N_B}.
\]

Here, the top and bottom rows provide the estimates based on the selection of \( A \) for \( N_A \) times and the selection of \( B \) for \( N_B \) times, respectively.

Each expected reward based on the abovementioned selections is given as follows:

\[
Q'_A = N_A \frac{N_A - L_A}{N_A} + N_B (\gamma - \frac{N_B - L_B}{N_B})
\]

\[
= N_A - L_A + (\gamma - 1) N_B + L_B,
\]

\[
Q'_B = N_A (\gamma - \frac{N_A - L_A}{N_A}) + N_B \frac{N_B - L_B}{N_B}
\]

\[
= N_B - L_B + (\gamma - 1) N_A + L_A.
\]

These expected rewards \( Q'_j \)’s are not the same as the learning rules of the TOW model, \( Q_j \)’s in Eq. (1). However, what we use substantially in the TOW model is the difference

\[
Q_A - Q_B = (N_A - N_B) - (1 + \omega) (L_A - L_B).
\]
When we transform the expected rewards $Q'_j$ into

$$Q'_A = Q_A'/(2 - \gamma),$$

(5)

$$Q'_B = Q_B'/(2 - \gamma),$$

(6)

we can obtain the difference

$$Q'_A - Q'_B = (N_A - N_B) - \frac{2}{2 - \gamma} (L_A - L_B).$$

(7)

A comparison of the coefficients of Eqs. (4) and (7) reveals that these two differences are always equal when $\omega = \omega_0$ satisfies,

$$\omega_0 = \frac{\gamma}{2 - \gamma}. (8)$$

Eventually, we can obtain the almost optimal weight parameter $\omega_0$ in terms of $\gamma$.

This derivation means that the TOW model has an equivalent learning rule with the system that can update both estimates simultaneously. The TOW model can imitate the system that determines its next moves at time $t + 1$ in referring to the estimate of each arm even if it was not selected at time $t$, as if the two arms were selected simultaneously at time $t$. This unique feature in the learning rule, derived from the fact that the sum of reward probabilities is given in advance, would be one of reasons for the TOW model’s high performance.

Performing Monte Carlo simulations, we confirmed that the performance of the TOW model with $\omega_0$ is comparable (a little lower) to its best performance, i.e., the TOW model with $\omega_{opt}$. To derive the $\omega_{opt}$ accurately, we need to take into account the dynamics of branches [12]. Detailed descriptions on how to derive the $\omega_{opt}$ and the proof for showing that $\omega_{opt}$ is optimal will be presented elsewhere.

In addition, we have confirmed that, because of the unique feature of the learning rule mentioned above, the performance of the TOW model with the parameter $\omega_0$ is better than that of well-known one-parameter algorithms, such as the $\epsilon$-greedy algorithm and the softmax algorithm, in almost all cases [2, 3].

2.2 Solid-type TOW (STOW) model

In a previous study [13, 14], we proposed the solid-type TOW (STOW) model that directly uses the advantage of the learning rule. Unlike the original TOW model (Eq. (1)), a more general form of $Q_k$ (Eq. (9)) is directly linked to a branch variable $X_k$ in this model (see Eq. (12)).

Consider a rigid body like an iron bar, as shown in Fig. 2. Here, variable $X_k$ corresponds to the displacement of branch $k$ from an initial position, where $k \in \{A, B\}$. If $X_k$ is the largest one, we consider that the body selects machine $k$. In the TOW model, the BP is represented in its inverse form, as we introduce “punishment” instead of “reward.” That is, when machine (channel) $k$ is played (chosen), the player is “punished” (failure to send packet) with a probability $1 - P_k$ (no coin).

We used the following estimate $Q_k$ ($k \in \{A, B\}$):

$$Q_k(t) = (N_k - L_k) - \sum_{\tau=0}^{t} \omega_c(\tau) l_k(\tau),$$

(9)

$$\omega_c(\tau) = \frac{2}{z(\tau)} - 1,$$

(10)
\[ z(\tau) = \frac{L_A(\tau)}{N_A(\tau)} + \frac{L_B(\tau)}{N_B(\tau)}. \]  

(11)

Here, \( N_k \) denotes the number of playing machine \( k \) and \( L_k \) represents the number of light stimuli (i.e., punishments) in \( k \), where \( l_k(t) = 1 \) if the light stimulus is applied at time \( t \), otherwise 0.

The displacement \( X_B (= -X_A) \) is determined by the following difference equations:

\[ X_B(t) = Q_B(t) - Q_A(t) + \delta, \]  

(12)

\[ \delta = \frac{n}{|d|} \sin(\pi t + \pi/2), \]  

(13)

\[ d = \frac{L_A}{N_A} - \frac{L_B}{N_B}. \]  

(14)

The body oscillates autonomously according to Eq. (13). The parameter \( a \) is fixed as \( a = 0.35 \) in this study. Consequently, +1 is added to \( X_k \) if a reward (no light stimulus) occurs, or \( -\omega e(t) \) is added to \( X_k \) if the light stimulus is applied on each selected side in addition to the oscillation.

2.3 Liquid-type TOW (LTOW) model

The liquid-type TOW (LTOW) model is used as an n-channel extension of the STOW model. It is assumed that the volume of liquid (yellow) is constant (see Fig. 3 for a 3-channel case).

![Fig. 3. Liquid-type TOW model (3-channel).](image)

The LTOW model for n-channel is described by the following equations for each user:

\[ Q_k(t) = N_k - (1 + \omega) L_k, \]  

(15)

\[ X_k(t) = Q_k(t) - \frac{1}{n-1} \sum_{k' \neq k}^n Q_{k'}(t) + A \cos(2\pi t/n + 2(k-1)\pi/n). \]  

(16)

Here, \( N_k \) and \( L_K \) are the same as those in the STOW model. In this study, we used \( n = 3 \) (\( k = 1, 2, 3 \) or \( A, B, C \)), \( \omega = 0.5 \), and \( A = 0.5 \). There are many possibilities of adding the oscillations for each user. For the sake of simplicity, we used the same oscillations for all users in this study.

2.4 Upper confidence bound 1-tuned (UCB1T) algorithm

The UCB1T algorithm is one of the most popular algorithms that can solve the BP and is widely used across the world for many applications including the CMA [5, 6] and web advertising [21]. The algorithm is as follows [18]:

- Initialization: Play each machine once.
- Loop: Play machine \( k \) that maximizes the following estimate \( R_k(t) \), where \( R_k \) denotes the average reward obtained from machine \( k \), \( N_k \) represents the number of times machine \( k \) has been played thus far, and \( N \) indicates the overall number of plays done thus far.
\[ Q_k(t) = R_k + \sqrt{\frac{2\ln N}{N_k}} \min\left(1/4, V_k(N_k)\right), \]
\[ V_k(s) = \frac{1}{s} \sum_{\tau=1}^{s} r_{k,\tau}^2 - R_{k,s}^2 + \sqrt{\frac{2\ln t}{s}}. \]

Here, \( r_{k,\tau} \) denotes the reward from machine \( k \) at \( \tau \) and \( R_{k,s} \) represents the average reward obtained from machine \( k \) until \( s \).

3. Results: Performance Evaluation

3.1 Easy problem instances

First, we consider problem instances such that \( P_A < P_B \) and \( P_A > P_B/2 \), that is, \( (P_A, P_B) = (0.2, 0.3), (0.3, 0.4), (0.4, 0.5), (0.5, 0.6), (0.6, 0.7), (0.7, 0.8), \) and \( (0.8, 0.9) \). When a user plays a machine that is different from the one that another user plays, we call this state “segregation,” i.e., \((\text{user 1}, \text{user 2}) = (A, B)\) or \((B, A)\). The two users in the segregation state will lose their rewards if they change their machines to play. Thus, the segregation state can be maintained stably as an “equilibrium” (see Table II).

| Table II. Payoff matrix for an easy problem where \((P_A, P_B) = (0.2, 0.3)\). |
|---------------------------------|------------------|------------------|
| user 1: \( A \) | user 2: \( B \) | user 2: \( A \) |
| user 1: \( A \) | 0.1 (0.1) | 0.2 (0.3) | |
| user 1: \( B \) | 0.3 (0.2) | 0.15 (0.15) | |

A state that gives the maximal total score, i.e., the maximal amount of reward obtained by the two users, is called “social maximum” in the context of algorithmic game theory [22]. Here, we designed each of the problem instances such that the segregation state corresponds to the social maximum.

The performance of each algorithm is evaluated in terms of the “score”; the accumulated amount of sending packets in the CMA. Figure 4 shows the user scores of the STOW model (a) and (b) and the UCB1T algorithm (c) and (d) for \( P_A = 0.3 \) and \( P_B = 0.4 \), and \( P_A = 0.7 \) and \( P_B = 0.8 \), respectively. There are 1000 open circles in each figure because we used 1000 samples. An open circle denotes the score of users 1 (horizontal axis) and 2 (vertical axis) until 1000 selections for a sample.

There are two clusters of points in Figs. 4c and d. These clusters give the social maximum as they correspond to the segregation equilibrium such that \((\text{user 1}, \text{user 2}) = (A, B)\) or \((B, A)\). When \( P_A = 0.3 \) and \( P_B = 0.4 \) (Fig. 4c), \((\text{user 1 score, user 2 score}) = (300, 400)\) or \((400, 300)\). When \( P_A = 0.7 \) and \( P_B = 0.8 \) (Fig. 4d), \((\text{user 1 score, user 2 score}) = (700, 800)\) or \((800, 700)\).

On the other hand, there are three or four clusters in the STOW model. The two larger clusters correspond to the segregation equilibrium, and the other smaller clusters correspond to the collision points due to some estimation errors. The STOW model always estimates \( P_A + P_B \) and \( P_A - P_B \) by its own internal variables (Eqs. (11) and (14)). Although this estimate generates the high performance of the STOW model, estimation errors are observed in a small number of samples (around 20 samples).

In the UCB1T algorithm, a mechanism that performs sufficient exploration is included. Because of this mechanism, an accurate judgment regarding which machine has a higher reward probability is possible, and there is “no spilling” (no mistake). Instead, there is a loss due to the sufficient exploration. On the other hand, in the STOW model, the judgment is made during early-stage oscillation, and the body moves toward one side. When early judgment is wrong, it returns to an opposite direction, but it may not return effectively because of the estimation error (Eqs. (11) and (14)) of environmental information \( P_A + P_B \) and \( P_A - P_B \). As a result, in the STOW model, the strategy that permits the presence of “spilling” is adopted, but it reduces the exploration loss by early judgment instead.

Despite the estimation errors in the STOW model, the total scores are comparable to those of the UCB1T algorithm as shown in Fig. 5. When \( P_A = 0.3 \) and \( P_B = 0.4 \), the average total score of the STOW model is higher than that of the UCB1T algorithm, whereas the average total score of the
Fig. 4. User scores of the STOW model and the UCB1T algorithm. An open circle denotes the score of users 1 (horizontal axis) and 2 (vertical axis) until 1000 selections for a sample. a) Scores of the STOW model for $P_A = 0.3$ and $P_B = 0.4$. b) Scores of the STOW model for $P_A = 0.7$ and $P_B = 0.8$. c) Scores of the UCB1T algorithm for $P_A = 0.3$ and $P_B = 0.4$. d) Scores of the UCB1T algorithm for $P_A = 0.7$ and $P_B = 0.8$.

STOW model is lower than that of the UCB1T algorithm when $P_A = 0.7$ and $P_B = 0.8$. Figure 6a also shows that the average total scores of the STOW model are comparable to those of the UCB1T algorithm although the two algorithms adopt completely different strategies from each other.

3.2 Hard problem instances

Secondly, we consider problem instances such that $P_A < P_B$ and $P_A < P_B/2$, that is, $(P_A, P_B) = (0.1, 0.3), (0.2, 0.5), (0.3, 0.7)$, and $(0.4, 0.9)$. In contrast to the first instances in which the segregation states are equilibria and social maxima, these second instances were designed so that the segregation states are social maxima and not equilibria. Instead, the Nash equilibrium $(P_B/2, P_B/2)$ exists (see Table III). The second instances are harder than the first ones, because the users need to avoid the Nash equilibrium to obtain the maximal total score.

Table III. Payoff matrix for a hard problem where $(P_A, P_B) = (0.1, 0.3)$.

<table>
<thead>
<tr>
<th></th>
<th>user 2: $A$</th>
<th>user 2: $B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1: $A$</td>
<td>0.05 (0.05)</td>
<td>0.1 (0.3)</td>
</tr>
<tr>
<td>user 1: $B$</td>
<td>0.3 (0.1)</td>
<td>0.15 (0.15)</td>
</tr>
</tbody>
</table>

Figure 7 shows the user scores of the STOW model (a) and (b) and the UCB1T algorithm (c) and (d) for $P_A = 0.2$ and $P_B = 0.5$, and $P_A = 0.4$ and $P_B = 0.9$, respectively. There are 1000 open circles in each figure because we used 1000 samples. An open circle denotes the score of users 1 (horizontal axis) and 2 (vertical axis) until 1000 selections for a sample.

There is one cluster of points in Figs. 7c and d. The cluster corresponds to the Nash equilibrium such that (user 1, user 2) = $(B, B)$. When $P_A = 0.2$ and $P_B = 0.5$ (Fig. 7c), (user 1 score, user 2 score) = (250, 250). When $P_A = 0.4$ and $P_B = 0.9$ (Fig. 7d), (user 1 score, user 2 score) = (450, 450).

On the other hand, there are three clusters in the STOW model. The largest cluster corresponds to the Nash equilibrium, and the remaining two clusters provide the social maximum as they correspond to the segregation state such that (user 1, user 2) = $(A, B)$ or $(B, A)$. When $P_A = 0.2$ and $P_B = 0.5$ (Fig. 7c), (user 1 score, user 2 score) = (200, 500) or (500, 200). When $P_A = 0.4$ and $P_B = 0.9$
Fig. 5. Average total scores of the STOW model (solid line) and the UCB1T algorithm (dashed line) for a) $P_A = 0.3$ and $P_B = 0.4$, and b) $P_A = 0.7$ and $P_B = 0.8$, respectively.

(Fig. 7d), (user 1 score, user 2 score) = (400, 900) or (900, 400). Because of these two clusters, the average total score of the STOW model is higher than that of the UCB1T algorithm.

Why can these two clusters be formed only for the STOW model? Samples in these clusters succeeded in reaching the social maximum while avoiding the Nash equilibrium. However, at the same time, these samples failed to select the best machine in the BP context because machine $B$ always dispenses more coins for each individual in the hard problem instances.

The estimation errors are observed in only around 20 samples in easy problem instances. In contrast, we can see more error samples in the hard problem instances. This is because it is harder to estimate which machine has a higher reward probability in the hard problem instances, where the difference between $P_A$ and $P_B/2$ decreases (0.05). In these cases, errors occur even when the estimation errors are not very significant. It is noted that some errors occur even in the UCB1T algorithm (see Fig. 7d). We can control the error rate of the STOW model by adjusting the parameter $a$ although this is fixed as $a = 0.35$ in this study. The performance of the STOW model for hard problem instances is enhanced because of the high error rate as the parameter $a$ decreases. In contrast, the performance of the STOW model for easy problem instances degrades.

Figure 6b shows the average total scores of the STOW model (filled circle) and the UCB1T algorithm
4. Direct interaction between users

In the previous section, we confirmed that the STOW model can avoid the Nash equilibrium by using “the estimation error” of environment variables, and achieve the social maximum. The only way to achieve the social maximum with indirect interactions between users is to use the estimation error. In this section, we propose a more positive way to realize the social maximum by using the “direct” interaction between users, since each user only indirectly interacts with another user through the payoff matrix in the STOW model.

The social maximum can be realized when we avoid the collisions by using the abovementioned interactions because the Nash equilibrium always exists in the diagonal components of a payoff matrix (or tensor), and the social maximum always exists in the non-diagonal components of the payoff matrix (or tensor) in the CMA cases. In this study, we adopt “the action-reaction law” as a direct interaction between the users of the STOW model and the LTOW model. If each user chooses channel $k$ and channel $k$ is available for packet transmission (i.e., machine $k$ emits a coin), $+1 (= \Delta X_k(t))$ is added to $X_k$. Conversely, if the user chooses channel $k$ that is unavailable (i.e., machine $k$ emits no coin), $-\omega (= \Delta X_k(t))$ is added to $X_k$. Note that we ignore the effects of the autonomous oscillations in the above discussion. That is, at the same time, each user receives the opposite displacement of the other users because of “the action-reaction law.” This can be expressed by the following equation for $m$ users and $n$ channels.

$$X^i_k(t+1) = X^i_k(t) + \Delta X^i_k(t) - \sum_{j \neq i}^{m} \Delta X^j_k(t) \quad (19)$$

Here, $X^i_k(t)$ denotes the $k$th variable of user $i$ for $k = \{1, 2, \cdots, n\}$ and $i = \{1, 2, \cdots, m\}$. After all, as long as the effects of the autonomous oscillations are ignored, conservation laws $X^i_k + X^j_k = \text{constant}$ hold for $i \neq j$, and collisions can be avoided.
Fig. 8. User scores of the STOW (LTO) model with direct interaction between users (DI). An open circle denotes the score of users 1 (horizontal axis) and 2 (vertical axis) until 1000 selections for a sample. a) Scores of the STOW model with DI for $P_A = 0.2$ and $P_B = 0.5$. b) Scores of the STOW model with DI for $P_A = 0.4$ and $P_B = 0.9$. c) Scores of the LTOW model without DI for $P_A = 0.2$, $P_B = 0.3$, and $P_C = 0.8$. d) Scores of the LTOW model with DI for $P_A = 0.2$, $P_B = 0.3$, and $P_C = 0.8$.

The user scores of the STOW model with direct interaction (DI) for the case of $(P_A = 0.2, P_B = 0.5)$ and the case of $(P_A = 0.4, P_B = 0.9)$ are shown in Figs. 8a and b. Compared with Figs. 7a and b, the large cluster that corresponds to the Nash equilibrium disappears, and the social maximum is realized. However, in the cases where the number of channels is 2, avoidance of collisions is equivalent to the achievement of the social maximum because the social maximum only exists in the non-diagonal components of the payoff matrix.

Here, we consider the case where the number of channels is 3 (see Table IV). There are segregation states that are not the social maximum. Therefore, exploration is needed to achieve the social maximum. In order to maximize a score of a user, each user should choose the channel C because the payoff of the channel C is always higher than that of the other choices irrespective of the other user’s choice. In other words, $(user 1, user 2) = (C, C)$ is the Nash equilibrium (NE). The social maximum (SM) is $(B, C)$ or $(C, B)$. Figures 8c and d show the user scores of the LTOW model (c) and the LTOW model with DI (d) for the case of $(P_A = 0.2, P_B = 0.3, P_C = 0.8)$, respectively. We can confirm that users successfully avoid the NE $(400, 400)$ and achieve the social maximum $(300, 800)$ and $(800, 300)$ in the right figure (the LTOW with DI).

Table IV. Payoff matrix for a hard problem where $(P_A, P_B, P_C) = (0.2, 0.3, 0.8)$.

<table>
<thead>
<tr>
<th></th>
<th>user 2: A</th>
<th>user 2: B</th>
<th>user 2: C</th>
</tr>
</thead>
<tbody>
<tr>
<td>user 1: A</td>
<td>0.1 (0.1)</td>
<td>0.2 (0.3)</td>
<td>0.2 (0.8)</td>
</tr>
<tr>
<td>user 1: B</td>
<td>0.3 (0.2)</td>
<td>0.15 (0.15)</td>
<td>0.3 (0.8) SM</td>
</tr>
<tr>
<td>user 1: C</td>
<td>0.8 (0.2)</td>
<td>0.8 (0.3) SM</td>
<td>0.4 (0.4) NE</td>
</tr>
</tbody>
</table>

5. Conclusion
In order to realize autonomous distributed control of the cognitive MAC protocol, we proposed variants of the TOW model as promising candidates. Here, we considered 2-user and 2-channel cases of the CBP for the sake of simplicity. We showed that the performance of the STOW model is comparable
to that of the UCB1T algorithm, which is known as the best algorithm for the BP, for easy (normal) problem instances. In the case of hard (abnormal) problem instances in which the users should not be attracted to the Nash equilibrium to achieve the social maximum, the STOW model outperformed the UCB1T algorithm. Finally, we proposed the LTOW model by introducing “the action-reaction law” in which the users avoided colliding with each other. The LTOW model achieved the social maximum for general CBP cases. The extension of our models and application to more realistic situations of cognitive MAC are topics of future research, aimed at the design of efficient and low-complexity decentralized CMA protocols.

Acknowledgments

This work was partially done when S.-J.K. and M.A. belonged to RIKEN Advanced Science Institute, which was reorganized and was integrated into RIKEN as of the end of March, 2013. We thank Prof. Masahiko Hara and Dr. Etsushi Nameda for valuable discussions.

References


