QFlip: An Adaptive Reinforcement Learning Strategy for the FlipIt Security Game

Lisa Oakley and Alina Oprea

Khoury College of Computer Sciences, Northeastern University, Boston MA, USA

Abstract. A rise in Advanced Persistent Threats (APTs) has introduced a need for robustness against long-running, stealthy attacks which circumvent existing cryptographic security guarantees. FlipIt is a security game that models attacker-defender interactions in advanced scenarios such as APTs. Previous work analyzed extensively non-adaptive strategies in FlipIt, but adaptive strategies rise naturally in practical interactions as players receive feedback during the game. We model the FlipIt game as a Markov Decision Process and introduce QFlip, an adaptive strategy for FlipIt based on temporal difference reinforcement learning. We prove theoretical results on the convergence of our new strategy against an opponent playing with a Periodic strategy. We confirm our analysis experimentally by extensive evaluation of QFlip against specific opponents. QFlip converges to the optimal adaptive strategy for Periodic and Exponential opponents using associated state spaces. Finally, we introduce a generalized QFlip strategy with composite state space that outperforms a Greedy strategy for several distributions including Periodic and Uniform, without prior knowledge of the opponent's strategy. We also release an OpenAI Gym environment for FlipIt to facilitate future research.

Keywords: Security games · FlipIt · Reinforcement learning · Adaptive strategies · Markov Decision Processes · Online learning.

1 Introduction

Motivated by sophisticated cyber-attacks such as Advanced Persistent Threats (APT), the FlipIt game was introduced by van Dijk et al. as a model of cyber-interactions in APT-like scenarios [3]. FlipIt is a two-player cybersecurity game in which the attacker and defender contend for control of a sensitive resource (for instance a password, cryptographic key, computer system, or network). Compared to other game-theoretical models, FlipIt has the unique characteristic of *stealthiness*, meaning that players are not notified about the exact state of the resource during the game. Thus, players need to schedule moves during the game with minimal information about the opponent's strategy. The challenge of determining the optimal strategy is in finding the best move times to take back resource control, while at the same time minimizing the overall number of moves (as players pay a cost upon moving). FlipIt is a repeated, continuous game, in which players can move at any time and benefits are calculated according to the asymptotic control of the resource minus the move cost.

The original FlipIt paper performed a detailed analysis of non-adaptive strategies in which players move according to a renewal process selected at the beginning of the game. Non-adaptive strategies are randomized, but do not benefit from feedback received during the game. In the real world, players naturally get information about the game and the opponent's strategy as play progresses. For instance, if detailed logging and monitoring is performed in an organization, an attacker might determine the time of the last key rotation or machine refresh upon system takeover. van Dijk et al. defined *adaptive strategies* that consider various amounts of information received during gameplay, such as the time since the last opponent move. However, analysis and experimentation in the adaptive case has remained largely unexplored. In a theoretical inspection, van Dijk et al. prove that the optimal Last Move adaptive strategy against Periodic and Exponential opponents is a Periodic strategy. They also introduce an adaptive Greedy strategy that selects moves to maximize local benefit. However, the Greedy strategy requires extensive prior knowledge about the opponent (the exact probability distribution of the renewal process), and does not always result in the optimal strategy [3]. Other extensions of FlipIt analyzed modified versions of the game [13,5,19,10,6], but mostly considered non-adaptive strategies.

In this paper, we tackle the challenge of analyzing the two-player FlipIt game with one adaptive player and one non-adaptive renewal player. We limit the adaptive player's knowledge to the opponent's last move time and show how this version of the game can be modeled as an agent interacting with a Markov Decision Process (MDP). We then propose for the first time the use of temporal difference reinforcement learning for designing adaptive strategies in the FlipIt game. We introduce QFlip, a Q-Learning based adaptive strategy that plays the game by leveraging information about the opponent's last move times. We explore in this context the instantiation of various reward and state options to maximize QFlip's benefit against a variety of opponents.

We start our analysis by considering an opponent playing with the Periodic with random phase strategy, also studied by [3]. We demonstrate for this case that QFlip with states based on the time since opponent's last move converges to the optimal adaptive strategy (playing immediately after the opponent with the same period). We provide a theoretical analysis of the convergence of QFlip against this Periodic opponent. Additionally, we perform detailed experiments in the OpenAI Gym framework, demonstrating fast convergence for a range of parameters determining the exploration strategy and learning decay. Next, we perform an analysis of QFlip against an Exponential opponent, for which van Dijk et al. determined the optimal strategy [3]. We show experimentally that QFlip with states based on the player's own move converges to the optimal strategy and the time to convergence depends largely on the adaptive player's move cost and the Exponential player's distribution parameters. Finally, we propose a generalized, composite QFlip instantiation that uses as state the time since last moves for both players. We show that composite QFlip converges to the optimal strategy for Periodic and Exponential. Remarkably, QFlip has no prior information about the opponent strategy at the beginning of the game, and most of the time outperforms the Greedy algorithm (which leverages information about the opponent strategy). For instance, QFlip achieves average benefit between 5% and 50% better than Greedy against Periodic and 15% better than Greedy against a Uniform player.

The implications of our findings are that reinforcement learning is a promising avenue for designing optimal learning-based strategies in cybersecurity games. Practically, our results also reveal that protecting systems against adaptive adversaries is a difficult task and defenders need to become adaptive and agile in face of advanced attackers. To summarize, our contributions in the paper are:

- We model the FlipIt game with an adaptive player competing against a renewal opponent as an MDP.
- We propose QFlip, a versatile generalized Q-Learning based adaptive strategy for FlipIt that does not require prior information about the opponent strategy.
- We prove QFlip converges to the optimal strategy against a Periodic opponent.
- We demonstrate experimentally that QFlip converges to the optimal strategy and outperforms the Greedy strategy for a range of opponent strategies.
- We release an OpenAI Gym environment for FlipIt to aid future researchers.

Paper organization. We start with surveying the related work in Section 2. Then we introduce the FlipIt game in Section 3 and describe our MDP modeling of FlipIt and the QFlip strategy in Section 4. We analyze QFlip against a Periodic opponent theoretically in Section 5. We perform experimental evaluation of Periodic and Exponential strategies in Section 6. We evaluate generalized composite QFlip against four distributions in Section 7, and conclude in Section 8.

2 Related Work

FlipIt, introduced by van Dijk et al. [3], is a non-zero-sum cybersecurity game where two players compete for control over a shared resource. The game distinguishes itself by its stealthy nature, as moves are not immediately revealed to players during the game. Finding an optimal (or dominant) strategy in FlipIt implies that a player can schedule its defensive (or attack) actions most effectively against stealthy opponents. van Dijk et al. proposed multiple non-adaptive strategies and proved results about their strongly dominant opponents and Nash Equilibria [3]. They also introduce the Greedy adaptive strategy and show that it results in a dominant strategy against Periodic and Exponential players, but it is not always optimal. The original paper left many open questions about designing general adaptive strategies for FlipIt. van Dijk et al. [1] analyzed the applications of the game in real-world scenarios such as password and key management.

Additionally, several FlipIt extensions have been proposed and analyzed. These extensions focus on modifying the game itself, adding additional players [10,6], resources [13], and move types [19]. FlipLeakage considers a version of

FlipIt in which information leakage is gradual and ownership of resource is obtained incrementally [5]. Zhang et al. consider limited resources with an upper bound on the frequency of moves and analyze Nash Equilibria in this setting [25]. Several games study human defenders players against automated attackers using Periodic strategies [20,8,18]. All of this work uses exclusively non-adaptive players, often limiting analysis to solely opponents playing periodically. The only previous work that considers adaptive strategies is by Laszka et al. [15,14], but in a modification of the original game with non-stealthy defenders. QFlip can generalize to play adaptively in these extensions, which we leave to future work.

Reinforcement learning (RL) is an area of machine learning in which an agent takes action on an environment, receiving feedback in the form of a numerical reward, and adapting its action policy over time in order to maximize its cumulative reward. Traditional methods are based primarily on Monte Carlo and temporal difference Q-Learning [22]. Recently, approximate methods based on deep neural networks have proved effective at complex games such as Backgammon, Atari and AlphaGo [23,17,21].

RL has emerged in security games in recent years. Han et al. use RL for adaptive cyber-defense in a Software-Defined Networking setting and consider adversarial poisoning attacks against the RL training process [9]. Hu et al. proposes the idea of using Q-Learning as a defensive strategy in a cybersecurity game for detecting APT attacks in IoT systems [11]. Motivated by HeartBleed, Zhu et al. consider an attacker-defender model in which both parties synchronously adjust their actions, with limited information on their opponent [26]. Other RL applications include network security [4], spatial security games [12], security monitoring [2], and crowdsensing [24]. Markov modeling for moving target defense has also been proposed [7,16].

To the best of our knowledge, our work presents the first application of RL to stealthy security games, resulting in the most effective adaptive FlipIt strategy.

3 Background on the FlipIt Game

FlipIt is a two-player game introduced by van Dijk et al. to model APT-like scenarios [3]. In FlipIt, players move at any time to take control of a resource. In practice, the resource might correspond to a password, cryptographic key, or computer system that both attacker and defender wish to control. Upon moving, players pay a move cost (different for each player). Success in the game is measured by player benefit, defined as the asymptotic amount of resource control (gain) minus the total move cost as described in Figure 2. The game is infinite, and we consider a discrete version of the game in which players can move at discrete time ticks. Figure 1 shows an example of the FlipIt game, and Figure 2 provides relevant notation we will use in the paper.

An interesting aspect of FlipIt is that the players do not automatically learn the opponent's moves in the game. In other words, moves are stealthy, and players need to move without knowing the state of the resource. There are two main classes of strategies defined for FlipIt:

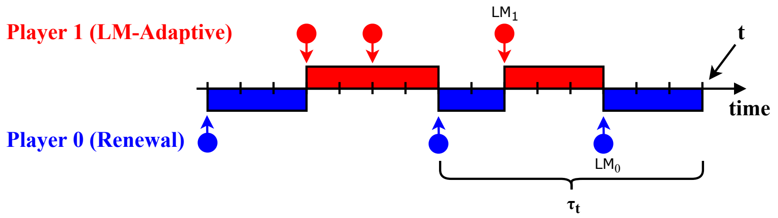


Fig. 1: Example of FlipIt game between Last Move adaptive Player 1 and Player 0 using a Renewal strategy. Rounded arrows indicate player moves. The first move of Player 1 is *flipping*, and the second move is *consecutive*. τ_t is the time since Player 0's last *known* move at time t and $\mathsf{LM_i}$ is Player i's *actual* last move at time t. Due to the stealthy nature of the game, $\tau_t \geq t - \mathsf{LM_0}$.

Symbol	Description
t	Time step (tick)
k_i	Player i's move cost
Γ_i	Player i's total gain (time in control)
n_i	Player i's total moves
β_i	Player i's total benefit. $\beta_i = \Gamma_i - k_i \cdot n_i$
τ_t	Time since opponent's last known move at time t
$\mathsf{LM_i}$	Player i's actual last move time at time t
ρ	Player 0's average move time

Symbol	Description
s_t	Observation
a_t	Action
r_t	Reward
γ	Future discount
α	Count of a_t in s_t
ϵ	Exploration parameter
d	Exploration discount
p	New move probability

Fig. 2: FlipIt notation (left) and QFlip notation (right)

Non-adaptive Strategies. Here, players do not receive any feedback upon moving. Non-adaptive strategies are determined at the beginning of the game, but they might employ randomization to select the exact move times. *Renewal strategies* are non-adaptive strategies that generate the intervals between consecutive moves according to a renewal process. The inter-arrival times between moves are independent and identically distributed random variables chosen from a probability density function (PDF). Examples of renewal strategies include:

- Periodic with random phase ($\mathcal{P}_\delta$): The player first moves uniformly at random with phase $R_\delta \in (0, \delta)$, with each subsequent move occurring periodically, i.e., exactly at δ time units after the previous move.
- Exponential: The inter-arrival time is distributed according to an exponential (memoryless) distribution $\mathcal{E}_\lambda$ with rate λ. The probability density function for $\mathcal{E}_\lambda$ is $f_{\mathcal{E}_\lambda}(x) = \lambda e^{-\lambda x}$, for $x > 0$, and 0 otherwise.
- Uniform: The inter-arrival time is distributed according to an uniform distribution $\mathcal{U}_{\delta,u}$ with parameters δ and u. The probability density function for $\mathcal{U}_{\delta,u}$ is $f_{\mathcal{U}_{\delta,u}}(x) = 1/u$, for $x \in [\delta - u/2, \delta + u/2]$, and 0 otherwise.
- Normal: The inter-arrival time is distributed according to a normal distribution $\mathcal{N}_{\mu,\sigma}$ with mean μ and standard deviation σ. The probability density function for $\mathcal{N}_{\mu,\sigma}$ is $f_{\mathcal{N}_{\mu,\sigma}}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2}$, for $x \in R$.

Adaptive Strategies. In these strategies, players receive feedback during the game and can adaptively change their subsequent moves. In *Last Move (LM)* strategies, players receive information about the opponent's last move upon moving in the game. This is the most restrictive and therefore most challenging subset of adaptive players, so we only focus on LM adaptive strategies here.

Theoretical analysis of the optimal LM strategy against specific Renewal strategies has been shown [3]. For the Periodic strategy, the optimal LM strategy is to move right after the Periodic player (whose moves can be determined from the LM feedback received during the game). The memoryless property of the exponential distribution implies that the probability of moving at any time is independent of the time elapsed since the last player's move. Thus, an LM player that knows the Exponential opponent's last move time has no advantage over a non-adaptive player. Accordingly, the dominant LM strategy against an Exponential player is still a Periodic strategy, with the period depending on the player's move cost and the rate of the Exponential opponent.

Greedy Strategy. To the best of our knowledge, the only existing adaptive strategy against general Renewal players is the "Greedy" strategy [3]. Greedy calculates the "local benefit", $L(z)$ of a given move time, z, as:

$$L(z) = \frac{1}{z}\Big[\int_{x=0}^{z} x\hat{f}_0(x)dx + z\int_{z}^{\infty}\hat{f}_0(x)dx - k_1\Big], \tag{1}$$

where $\hat{f}_0(x) = f_0(\tau+x)/(1-F_0(\tau))$, f_0 is the probability density function (PDF) of the opponent's strategy, F_0 is the corresponding cumulative density function (CDF), and τ is the interval since the opponent's last move. Greedy finds the move time, $\hat{z}$, which maximizes this local benefit, and schedules a move at $\hat{z}$ if the maximum local benefit is positive. In contrast, if the local benefit is negative, Greedy chooses not to move, dropping out of the game.

Although the Greedy strategy is able to compete with any Renewal strategy, it is dependent on prior knowledge of the opponent's strategy. van Dijk et al. showed that Greedy can play optimally against Periodic and Exponential players [3]. However, they showed a strategy for which Greedy is not optimal. This motivates us to look into other general adaptive strategies that *achieve higher benefit than Greedy* and *require less knowledge about the opponent's strategy.*

4 New Adaptive Strategy for FlipIt

Our main insight is to apply traditional reinforcement learning (RL) strategies to the FlipIt security game to create a Last Move adaptive strategy that outperforms existing adaptive strategies. We find that modeling FlipIt as a *Markov Decision Process* (*MDP*) and defining an LM Q-Learning strategy is non-trivial, as the stealthy nature of the game resists learning. We consider the most challenging setting, in which the adaptive player has no prior knowledge on the opponent's strategy. In this section, we present QFlip, a strategy which is able to overcome those challenges and elegantly compete against any Renewal opponent.

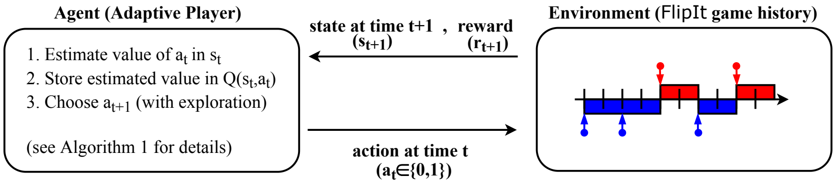

Agent (Adaptive Player)
1. Estimate value of a_t in s_t
2. Store estimated value in Q(s_t,a_t)
3. Choose a_{t+1} (with exploration)
(see Algorithm 1 for details)
state at time t+1 , reward
(s_{t+1}) (r_{t+1})
action at time t
(a_t∈{0,1})
Environment (FlipIt game history)

Fig. 3: Modeling FlipIt as an MDP.

Table 1: Observation Schemes

Scheme	s_0	s_t for $t > 0$
oppLM	-1	τ_t if $\mathsf{LM}_1 >$ player 0 first move, otherwise -1
ownLM	0	$t - \mathsf{LM}_1$
composite	$(s_0^{\mathsf{ownLM}}, s_0^{\mathsf{oppLM}})$	$(s_t^{\mathsf{ownLM}}, s_t^{\mathsf{oppLM}})$

4.1 Modeling FlipIt as an MDP

Correctly modeling the game of two-player FlipIt as an *MDP* is as important to our strategy's success as the RL algorithm itself. In our model, player 1 is an *agent* interacting with an *environment* defined by the control history of the FlipIt resource as depicted in Figure 3.

We consider the infinite but discrete version of FlipIt and say that at every time step (tick), $t \in \{1, 2, \dots\}$, the game is in some state $s_t \in \mathcal{S}$ where $\mathcal{S}$ is a set of observed state values dependent on the history of the environment. At each time t, the agent chooses an action $a_t \in \mathcal{A} = \{0, 1\}$ where 0 indicates waiting, and 1 indicates moving. The environment updates accordingly and sends the agent state s_{t+1} and reward r_{t+1} defined in Table 1 and Equation (3), respectively.

Defining optimal state values and reward functions is essential to generating an effective RL algorithm. In a stealthy game with an unknown opponent, this is a non-trivial task that we will investigate in the following paragraphs.

Modeling State. At each time step t, the LM player knows two main pieces of information: its own last move time (LM_1), and the time since the opponent's last *known* move (τ_t). The observed state can therefore depend on one or both of these values. We define three observation schemes in Table 1. We compare these observation schemes against various opponents in the following sections.

Modeling Reward. Temporal difference learning algorithms leverage incremental updates from rewards at each time tick, and therefore require the environment to transmit meaningful reward values. A good reward should be flexible, independent of prior knowledge of opponent strategy, and most importantly promote the ultimate goal of achieving high benefit. We divide actions into three resulting categories: *flipping*, *consecutive*, and *no-play* based on the type of action and the state of the environment as depicted in Table 2.

The most straightforward reward after each action is a *resulting benefit*

$$\beta_1^{(a_t,s_t)} = \Gamma_1^{(a_t,s_t)} - k_1 \tag{2}$$

Table 2: Player 1 Action Categories

Move type	a_t	Env State	Cost	Outcome	Explanation
flipping	1	$\mathsf{LM}_0 > \mathsf{LM}_1$	$-k_1$	$\tau_{t+1} = t - \mathsf{LM}_0 + 1$	Player 1 takes control
consecutive	1	$\mathsf{LM}_0 \leq \mathsf{LM}_1$	$-k_1$	$\tau_{t+1} = \tau_t + 1$	Player 1 moves while in control
no-play	0	any	0	$\tau_{t+1} = \tau_t + 1$	Player 1 takes no action

where $\mathit{\Gamma}_1^{(a_t,s_t)}$ is the *resulting gain*, or Player 1's additional time in control between time t and the opponent's next move as a result of taking action a_t in state s_t. These rewards would sum to equal Player 1's total benefit over the course of the game, therefore exactly matching the goal of maximized benefit.

For *consecutive* moves, $\beta_1^{(a_t,s_t)} = -k_1$, as Player 1 is already in control, and therefore attains no additional gain, resulting in a wasted move. For *no-plays*, $\beta_1^{(a_t,s_t)} = 0$, as not moving guarantees no additional gain and incurs no cost.

The main challenge here comes from determining reward for *flipping* moves. Consider the case where the opponent plays more than once between two of the agent's moves. Here, it is impossible to calculate an accurate $\beta_1^{(a_t,s_t)}$, as the agent cannot determine the exact time they lost control. Moreover, there is no way to calculate future gain from any move against a randomized opponent, as the opponent's next move time is unknown.

We acknowledge a few ineffective responses to these challenges. The first rewards Player 1 for playing soon after the opponent (higher reward for lower resulting τ_{t+1} values). This works against a Periodic opponent, but not work against an Exponential opponent as it does not reward optimal play. Another approach is a reward based on prior gain, rather than resulting gain. This is difficult to calculate and rewards previous moves, rather than the current action.

We determined experimentally that the best reward for a_t against an unknown opponent is a fixed constant related to the opponent's move frequency as follows

$$r_{t+1} = \begin{cases} 0 & \text{if } \textit{no-play} \text{ at time } t \\ -k_1 & \text{if } \textit{consecutive} \text{ move at time } t \\ \frac{\rho - k_1}{c} & \text{if } \textit{flipping} \text{ move at time } t \end{cases} \tag{3}$$

where ρ is an estimate of Player 0's average move frequency and c is a constant determined before gameplay (for normalization). Playing often toward the beginning of the game and keeping track of the observed move times can provide a rough estimate of ρ. This reward proves highly effective, while maintaining the flexibility to play against any opponent without any details of their strategy.

4.2 The QFlip Strategy

In this section we present a new, highly effective LM adaptive strategy, QFlip, based on existing temporal difference reinforcement learning techniques. QFlip plays within our FlipIt model from Section 4.1. Though optimized to play against Renewal opponents, a QFlip player can compete against any player, including other adaptive opponents. To the best of our knowledge, QFlip is the first adaptive strategy which can play FlipIt against both Renewal and non-Renewal opponents without any prior knowledge about their strategy.

Value Estimation. QFlip uses feedback attained from the environment after each move and the information gathered during gameplay to estimate the value of action a_t in state s_t. Player 1 has no prior knowledge of the opponent's strategy, therefore must learn an optimal strategy in real-time. We adopt an online temporal difference model of value estimation where $Q(s_t, a_t)$ is the expected value of taking action a_t in state s_t as in [22].

We start by defining the actual value of an action a_t in state s_t as a combination of the immediate reward and potential future value as

$$V_{s_t,a_t} = r_{t+1} + \gamma \cdot \max_{a' \in \mathcal{A}} Q(s_{t+1}, a'). \tag{4}$$

where $0 \leq \gamma \leq 1$ is a constant discount to the estimated future value, and r_{t+1} is the environment-provided reward from Equation (3).

After each tick, we update our value estimate by a discounted difference between estimated move value and actual move value as follows:

$$Q_{\alpha+1}(s_t, a_t) = Q_\alpha(s_t, a_t) + \frac{1}{\alpha+1}(V_{s_t,a_t} - Q_\alpha(s_t, a_t)) \tag{5}$$

where α is the number of times action a has been performed in state s, and $1/\alpha$ is the *step-size* parameter, which discounts the change in estimate proportionally to the number of times this estimate has been modified.

This update policy uses the estimate error to step toward the optimal value estimate at each state. Note that, if $\gamma = 0$, we play with no consideration of the future, and $Q_\alpha(s, a)$ is just an average of the environment-provided rewards over all times action a was performed in state s.

Action Choice (Exploration). A key element of any reinforcement learning algorithm is balancing exploitation of learned value estimation with exploration of new states and actions to avoid getting stuck in local maxima. We employ a modified *decaying-ϵ-greedy* exploration strategy from [22] as

$$a_t = \begin{cases} \text{choose uniformly at random from } \mathcal{A} \text{ with probability } \epsilon' \\ \text{argmax}_{a \in \mathcal{A}} Q(s_t, a) \text{ with probability } 1 - \epsilon' \end{cases} \tag{6}$$

where $\epsilon' = \epsilon \cdot e^{-d \cdot v}$ for constant exploration and decay parameters $0 \leq \epsilon, d < 1$ and v equal to the number of times QFlip has visited state s_t. If $Q(s_t, 0) = Q(s_t, 1)$, we choose $a_t = 0$ with probability p, and $a_t = 1$ with probability $1 - p$.

Algorithm Definition. We then define the agent's policy in Algorithm 1. This is a temporal difference Q-Learning based algorithm. The algorithm first estimates the opponent move rate, ρ, by playing several times. This step is important to determine if it should continue playing or drop out (when $k_1 \geq \rho$), and to fix the environment reward. If the agent decides to play, it proceeds to initialize the

Q table of estimated rewards in each state and action to 0. The agent's initial state is set according to Table 1. The action choice is based on exploration, as previously discussed. Once an action is selected, the agent receives the reward and new state from the environment and updates Q according to Equation (5).

Algorithm 1

```
Estimate rate of play $\rho$ of opponent and drop out if $k_1 \geq \rho$
Initialize 2D table Q with all zeros
Initialize $s_0$ according to observation type (see section 4.1)
for $t \in \{1, 2, \dots\}$ do
    if Q($s_t$,0)=Q($s_t$,1) then
        $a_t \leftarrow 0$ with probability $p$, else $a_t \leftarrow 1$
    else
        Choose action $a_t$ according to equation (6)
    Simulate action $a_t$ on environment, and observe $s_{t+1}$, $r_{t+1}$
    Update Q($s_t$,$a_t$) according to equation (5)
```

5 Theoretical Analysis for Periodic Opponent

We consider first an opponent playing periodically with random phase. Previous work has focused primarily on analyzing $\mathcal{P}_\delta$ strategies in a non-adaptive context [3,15,14,6]. We first show theoretically that QFlip eventually learns to play optimally against a Periodic opponent when the future discount γ is set at 0. We employ this restriction because, when $\gamma > 0$, the actual value of a_t in s_t (V_{s_t,a_t}) depends on the maximum estimated value in s_{t+1}, which changes concurrently with $Q(s_t, a_t)$. Additionally, Section 6.2 shows experimentally that changing γ does not have much effect on benefit.

In the discrete version of FlipIt a player using the Periodic strategy $\mathcal{P}_\delta$ plays first at some uniformly random $R_\delta \in \{0, \dots, \delta\}$, then plays every δ ticks for all subsequent moves. In this case, the optimal LM strategy is to play immediately after the opponent. Our main result is the following theorem, showing that QFlip converges to the optimal strategy with high probability.

Theorem 1. *Playing against a $\mathcal{P}_\delta$ opponent with $\gamma = 0$, $k_1 < \delta$,* QFlip *using the* ownLM *observation scheme converges to the optimal LM strategy as $t \to \infty$.*

We will prove this theorem by first showing that QFlip visits state $\delta + 1$ infinitely often, then claiming that QFlip will eventually play once in state $\delta + 1$. We conclude by proving QFlip eventually learns to play in state $\delta + 1$ and no other state. This is exactly the optimal strategy of playing right after the opponent. Because the $\mathcal{P}_\delta$ strategy is deterministic after the random phase, we can model Player 1's known state and transitions according to the actual state of the game as in Figure 4. We prove several lemmas and finally the main theorem below.

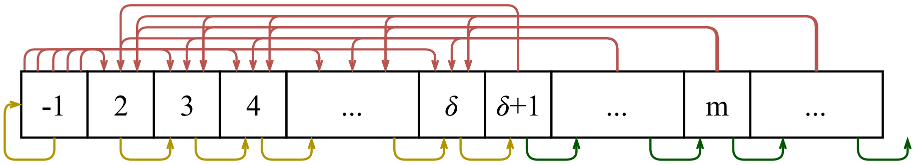


Fig. 4: QFlip using oppLM states against $\mathcal{P}_\delta$. Arrows indicate state transitions. Red is $a_t = 1$, green is $a_t = 0$, and yellow is either $a_t = 0$ or $a_t = 1$.

Lemma 1. *If $t > R_\delta$ and $a_t = 1$,* QFlip *will visit state $\delta + 1$ in at most δ additional time steps.*

Proof. We want to show that, for any s_t, with $t > R_\delta$, choosing $a_t = 1$ means that QFlip will visit state $\delta + 1$ again in at most δ additional time steps.

Case 1: Assume $1 < s_t < \delta + 1$. We see from Figure 4 that $s_{t+1} = s_t + 1$, for all a_t when $-1 < s_t < \delta + 1$. Therefore, QFlip will reach state $\delta + 1$ in $\delta + 1 - s_t < \delta$ additional time steps.

Case 2: Assume $s_t \geq \delta + 1$. The opponent is $\mathcal{P}_\delta$, so $\mathsf{LM}_1 < \mathsf{LM}_0$ at time t. Given $a_t = 1$, Table 2 gives $s_{t+1} = t - \mathsf{LM}_0 + 1 < \delta + 1$, returning to Case 1.

Case 3: Assume $s_t = -1$ and QFlip chooses $a_t = 1$. From Tables 1 and 2, we have that $s_{t+1} = t - \mathsf{LM}_0 < \delta + 1$, returning again to Case 1. □

Lemma 2. *Playing against a $\mathcal{P}_\delta$ opponent with $\gamma = 0$, $0 \leq p < 1$ and $k_1 < \delta$,* QFlip *visits state $\delta + 1$ infinitely often.*

Proof. We prove by induction on the number of visits, n, to state $\delta + 1$ that QFlip visits state $\delta + 1$ infinitely often.

Base of Induction: We show that, starting from $s_0 = -1$, QFlip will reach $s_t = \delta + 1$ with probability converging to 1. If QFlip chooses $a_t = 1$ when $R_\delta < t \leq \delta$, this is a *flipping* move, putting QFlip in state $s_{t+1} < \delta + 1$. If not, $t > \delta \geq R_\alpha$. When $s_t = -1$, QFlip chooses $a_t = 0$ with probability p and $a_t = 1$ with probability $1 - p$. Therefore

$$P[\text{QFlip not flipping after } v \text{ visits to } s = -1] = p^v. \tag{7}$$

This implies that QFlip will flip with probability $1 - p^v \to 1$ as $t = v \to \infty$. By Lemma 1, QFlip will reach state $\delta + 1$ in finite steps with probability 1.

Inductive Step: Assume QFlip visits state $\delta + 1$ n times. Because we are considering an infinite game, at any time t there are infinitely many states that have not been visited. Therefore $\exists m \geq \delta + 1$ such that $\forall s > m, Q(s) = (0, 0)$.

If QFlip flips at state $s_t \in \{\delta + 1, ..., m\}$, it will reach $\delta + 1$ again in a finite number of additional steps by Lemma 1.

If QFlip does not flip at state $s_t \in \{\delta + 1, ..., m\}$, we have

$$P[\text{QFlip does not move after } z \text{ steps}] = p^z \tag{8}$$

since probability of moving when $Q(s) = (0,0)$ is p and not moving implies $s_{t+1} = s_t{+}1 > m$. Therefore, as $t = z \to \infty$, QFlip will flip again with probability $1 - p^z \to 1$.

By mathematical induction, we have that state $\delta{+}1$ is visited infinitely often with probability converging to 1. □

Lemma 3. *If $\gamma = 0$ and $k_1 < \delta$, QFlip will eventually choose to move in state $\delta + 1$ with probability 1.*

Proof. If QFlip flips in any visit to state $\delta + 1$, the conclusion follows.

Assume QFlip does not flip in state $s = \delta + 1$. Since $\gamma = 0$, from Equation (4), $V_{s_t,a_t} = 0$ for $a_t = 0$. Therefore $Q(s) = (0,0)$ and we have

$$P[\text{QFlip does not flip after } v \text{ visits to state } \delta + 1] = p^v. \tag{9}$$

By Lemma 2, we know that QFlip visits state $\delta + 1$ infinitely often. Therefore, probability that QFlip moves in state $\delta + 1$ is $1 - p^v \to 1$ as $v \to \infty$. □

Proof of Theorem 1 We will now prove the original theorem, using these lemmas. To prove that QFlip plays optimally against $\mathcal{P}_\delta$, we must show that it will eventually (1) play at $s = \delta + 1$ at each visit and (2) not play at $s \neq \delta + 1$ at any visit. Assuming $\gamma = 0$, we have from section 2.5 of [22] that

$$Q_{\alpha+1}(s,a) = Q_\alpha(s,a) + \frac{1}{\alpha+1} \cdot (r_{\alpha+1} - Q_\alpha(s,a)) = \frac{1}{\alpha+1} \sum_{\alpha+1}^{i=1} r_i. \tag{10}$$

Here we denote by r_i the reward obtained the i-th time state s was visited and action a was taken. Additionally, $\mathcal{P}_\delta$ plays every δ time steps after the random phase. Therefore we derive from Equation (3):

$$r_i = \begin{cases} 0 & \text{if } a_i = 0 \\ -k_1 & \text{if } a_i = 1 \text{ and } 1 \le s_i \le \delta \\ \frac{\delta - k_1}{c} & \text{if } a_i = 1 \text{ and } s_i \ge \delta + 1 \end{cases} \tag{11}$$

By Equations (10) and (11), we have for all states s,

$$Q_\alpha(s,0) = \frac{1}{\alpha} \sum_{i=1}^{\alpha} 0 = 0. \tag{12}$$

First we show that QFlip will eventually choose $a_t = 0$ in all states $1 < s_t < \delta + 1$. Consider some s_t such that $1 < s < \delta + 1$. If QFlip never chooses $a_t = 1$ in this state, we are done. Assume QFlip plays at least once in this state, $\alpha > 0$, then

$$Q_\alpha(s,1) = \frac{1}{\alpha} \sum_{i=1}^{\alpha} -k_1 = -k_1 < 0 = Q_\alpha(s,0) \tag{13}$$

since $k_1 > 0$. Therefore $\text{argmax}_{a \in \mathcal{A}} Q(s,a) = 0$. Because $\epsilon' = \epsilon \cdot e^{-d \cdot v}$, and $0 \leq \epsilon, d < 1$, as $v \to \infty$ we have that $\epsilon' \to 0$. Therefore $P[$QFlip does not play at $s] \to 1$ for $1 \leq s \leq \delta$ as desired.

Next we show that QFlip will eventually play at state $\delta + 1$ at each visit. From Lemma 3, we know that QFlip will play once at $s = \delta + 1$ with probability 1, meaning $\alpha > 0$ with probability 1. By Equations (10) and (11) we have for $\alpha > 0$ and $s = \delta + 1$.

$$Q_\alpha(s,1) = \frac{1}{\alpha} \sum_{i=1}^{\alpha} \frac{\delta - k_A}{c} = \frac{\delta - k_A}{c} > 0 = Q_\alpha(s,0). \tag{14}$$

Now $\text{argmax}_{a \in \mathcal{A}} Q(s,a) = 1$, so as $\epsilon' \to 0$, $P[$QFlip plays at $s = \delta + 1] \to 1$.

If QFlip plays at state $\delta + 1$, it will not reach states $s > \delta + 1$, and thus cannot play in those states. Therefore, as $t \to \infty$, $P[$QFlip plays optimally$] \to 1$.

□

6 QFlip Against $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$ Opponents

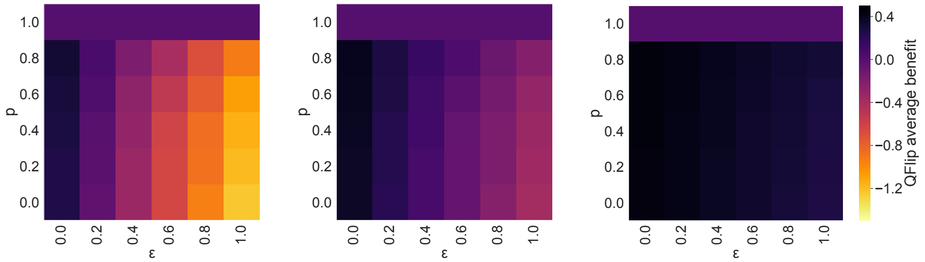
In this section we show experimentally that QFlip learns an optimal strategy against $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$ opponents. Prior to starting play, we allow QFlip to choose its observation scheme based on the opponent's strategy. QFlip chooses the oppLM observation against $\mathcal{P}_\delta$ and the ownLM observation against $\mathcal{E}_\lambda$, but sets other parameters of QFlip identically. This reflects the theoretical analysis from [3] which states that an optimal adaptive strategy against a $\mathcal{P}_\delta$ opponent depends on τ, while τ is irrelevant in optimal strategies against an $\mathcal{E}_\lambda$ opponent. Next section, we generalize QFlip to play with no knowledge of opponent strategy.

6.1 Implementation

All simulations (https://github.com/lisaoakley/flipit-simulation) are written in Python 3.5 with a custom OpenAI Gym environment for FlipIt (https://github.com/lisaoakley/gym-flipit). We ran each experiment over a range of costs and Player 0 parameters within the constraint that $k_1 < \rho$, as other values have an optimal drop out strategy. For consistency, we calculated ρ from the distribution parameters before running simulations. We report averages across multiple runs, choosing number of runs and run duration to ensure convergence. Integrals are calculated using the *scipy.integrate.quad* function. For Greedy's maximization step we used *scipy.optimize.minimize* with the default "BFGS" algorithm. QFlip can run against a variety of opponents with minimal configuration, thus we set all QFlip hyper-parameters identically across experiments, namely $\gamma = 0.8$, $\epsilon = 0.5$, $d = 0.05$, $c = 5$, and $p = 0.7$ unless otherwise noted.

6.2 QFlip vs. Periodic

In Section 5 we proved that QFlip will eventually play optimally against $\mathcal{P}_\delta$ when future discount γ is 0 and $k_1 < \delta$. In this section, we show experimentally that



(a) $8,000$ ticks (b) $16,000$ ticks (c) $64,000$ ticks

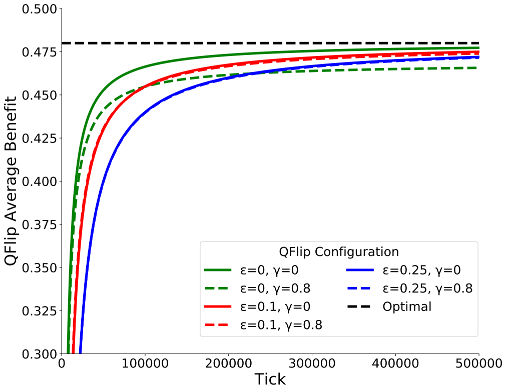
Fig. 5: Player's 1 average benefit for different ϵ and p parameters at three time ticks. Here QFlip plays with oppLM, $\gamma = 0$ and $k_1 = 25$ against a $\mathcal{P}_\delta$ opponent with $\delta = 50$, averaged over 10 runs. Darker purples mean higher average benefit. QFlip converges to optimal (.48), improving more quickly with low exploration.

most configurations of QFlip using oppLM quickly lead to an optimal strategy. Additionally, we show there is little difference in benefit when $\gamma > 0$.

Learning Speed. When $\gamma = 0$, Equation (4) reduces to $V_{s_t,a_t} = r_{t+1}$. In this case we verify that QFlip learns an optimal strategy for all exploration parameters, but that lower exploration rates cause QFlip to reach optimal benefit more quickly. Against a Periodic opponent, $Q_\alpha(s,a)$ is constant after QFlip takes action a in state s at least once ($\alpha > 0$). Thus, exploring leads to erroneous moves. When the probability of moving for the first time in state s is low (p is high), QFlip makes fewer costly incorrect moves in states $s < \delta + 1$ leading to higher benefit. When $p = 1$, QFlip never plays and $\beta_1 = 0$. Figure 5 displays Player 1's average benefit for different values of p and ϵ. QFlip achieves close to optimal benefit after $64,000$ ticks with high exploration rates ϵ and high probability of playing in new states $(1 - p)$, and in as little as $8,000$ ticks with no exploration ($\epsilon = 0$) and low $1 - p$.

Varied Configurations. When $\gamma > 0$, V_{s_t,a_t} factors in the estimated value of state s_{t+1} allowing QFlip to attain positive reward for choosing not to move in state $\delta + 1$. The resulting values in the Q table can negatively impact learning. Figure 6 shows the average benefit over time (left) and the number of non-optimal runs (right) for different ϵ and γ values. We observe that QFlip performs non-optimally on 38% of runs with $\gamma = 0.8$ and $\epsilon = 0$ but has low benefit variation between runs. However, increasing ϵ even to 0.1 compensates for this and allows QFlip to play comparably on average with future estimated value ($\gamma > 0$) as with no future estimated value ($\gamma = 0$). This result allows us flexibility in configuring QFlip which we will leverage to maintain hyper-parameter consistency against all opponents. In the rest of the paper we set $\gamma = 0.8$, $\epsilon = 0.5$, and $p = 0.7$.

Comparison to Greedy. Assuming the Greedy strategy against $\mathcal{P}_\delta$ plays first at time δ, it will play optimally with probability $1 - k_1/\delta$. However, with probability k_1/δ, Greedy will drop out after its first adaptive move [3]. We compare



(a) Learning over time averaged over 50 runs per configuration. Optimal benefit vs. $\mathcal{P}_\delta$ with $\delta = 50$ and $k_1 = 25$ in a discrete game is 0.48

γ	ϵ	# non-optimal runs	min benefit	max benefit
0	0	0	0.477	0.478
	0.1	0	0.474	0.476
	0.25	0	0.471	0.473
	0.5	0	0.465	0.468
0.8	0	19	0.417	0.478
	0.1	3	0.455	0.476
	0.25	1	0.452	0.473
	0.5	0	0.465	0.468

(b) Statistics over 50 runs. "Non-optimal" runs have average benefit $> .02$ less than optimal (.48) after 500,000 ticks.

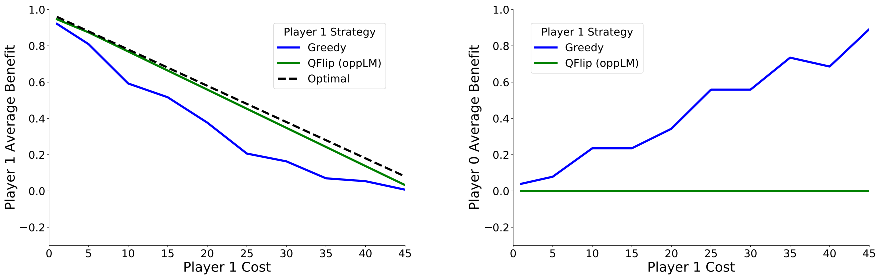
Fig. 6: QFlip using oppLM with $k_1 = 25$ playing against $\mathcal{P}_\delta$ with $\delta = 50$.

QFlip and Greedy against $\mathcal{P}_\delta$ for $\delta = 50$ across many costs in Figure 7. QFlip consistently achieves better average benefit across runs, playing close to optimally on average. Additionally, Player 0 with $k_0 = 1$ attains more benefit on average against a Greedy opponent as a result of these erroneous drop-outs. For $k_1 < 45$, QFlip attains benefit between 5% and 50% better than Greedy on average.

6.3 QFlip vs. Exponential

The optimal LM strategy against an $\mathcal{E}_\lambda$ opponent is proven in [3] to be $\mathcal{P}_\delta$ with δ dependent on k_0 and λ. The exponential distribution is memoryless, so optimal δ is independent of time since the *opponent's* last move. Optimal QFlip ignores τ and moves δ steps after its *own* last move. QFlip therefore prefers the ownLM observation space from Table 1, rather than oppLM used against Periodic.

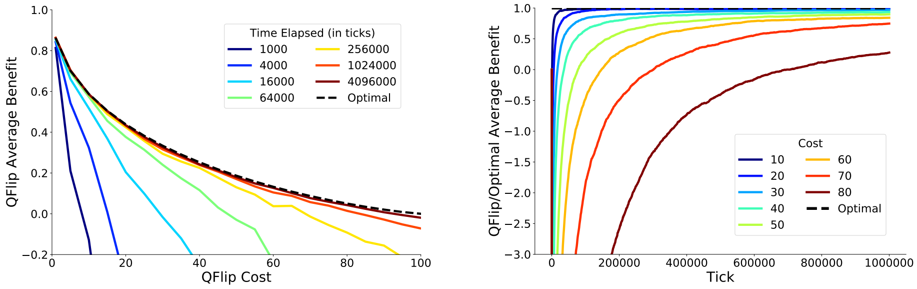
For QFlip to learn any $\mathcal{P}_\delta$ strategy, it must visit states $s < \delta$ many times. When playing against an Exponential opponent, the optimal δ grows quickly as k_1 increases. For instance, against an $\mathcal{E}_\lambda$ opponent with $\lambda = 1/100$ the optimal $\mathcal{P}_\delta$ strategy is $\delta = 53$ for $k_1 = 10$ and $\delta = 389$ for $k_1 = 90$ [3]. As a result, the optimal Periodic strategy takes longer to learn as k_1 grows. Figure 8 (a) shows the average benefit versus cost after running the algorithm for up to 4.096 million ticks for rate of the Exponential distribution $\lambda = 1/100$. For small costs, QFlip learns to play optimally within a very short time $(16,000$ ticks). As the move cost increases, QFlip naturally takes longer to converge. We verified this for other values of λ as well. Figure 8 (b) shows how the benefit varies by time for various move costs. Given enough time, QFlip converges to a near-optimal Periodic strategy for all costs (even as high as $k_1 = 100$, which results in drop out for $\lambda = 1/100$).



(a) Player 1 average benefit by cost

(b) Player 0 average benefit by cost

Fig. 7: Player 1 and Player 0's average benefit for QFlip and Greedy across Player 1 costs. QFlip with oppLM playing against $\mathcal{P}_\delta$ with fixed $k_0 = 1$ and $\delta = 50$ for 250,000 ticks, averaged over 100 runs.



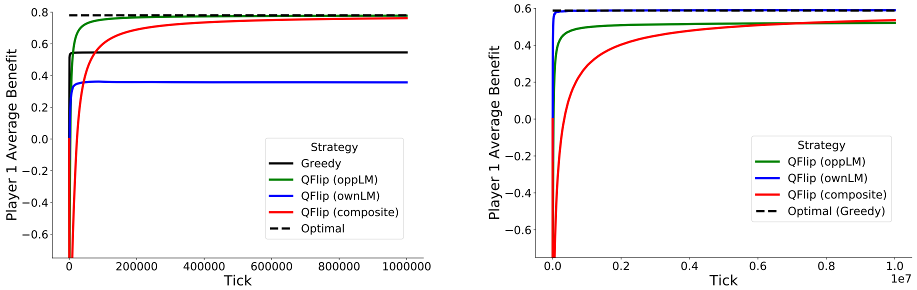
(a) QFlip average benefit versus cost. QFlip becomes near-optimal for all costs when duration is high.

(b) Ratio of QFlip and Optimal average benefit versus time. QFlip achieves optimal benefit quickly when costs are low.

Fig. 8: QFlip with ownLM playing against $\mathcal{E}_\lambda$ with $\lambda = 1/100$.

7 Generalized QFlip Strategy

Previous sections show that QFlip converges to optimal using the oppLM and ownLM observation schemes for the $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$ opponents respectively. In this section we show that QFlip using a composite observation scheme can play optimally against $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$, and perform well against other Renewal strategies without any knowledge of the opponent's strategy. The composite strategy uses as states both Player 1's own last move time (LM_1), and the time since the opponent's last *known* move (τ_t), as described in Table 1. Composite QFlip is the first general adaptive FlipIt strategy that has no prior information on the opponent.



(a) QFlip against $\mathcal{P}_\delta$with $\delta = 50$.

(b) QFlip against $\mathcal{E}_\lambda$ with $\lambda = 1/100$.

Fig. 9: QFlip with $k_1 = 10$ averaged over 10 runs. QFlip converges to optimal against $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$ using oppLM and ownLM observation schemes respectively, and plays close to optimally against both with composite observation scheme.

7.1 Composite QFlip Against $\mathcal{P}_\delta$ and $\mathcal{E}_\lambda$ Opponents

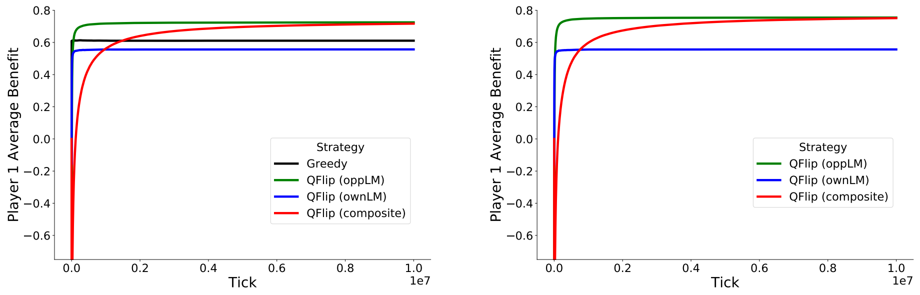
Figure 9 shows that QFlip's average benefit eventually converges to optimal against both $\mathcal{E}_\lambda$ and $\mathcal{P}_\delta$ when using a composite observation scheme. We note that it takes significantly longer to converge to optimal when using the composite scheme. This is natural, as QFlip has an enlarged state space (quadratic compared to oppLM and ownLM observation schemes) and now visits each state less frequently. We leave approximation methods to expedite learning to future work.

7.2 Composite QFlip Against Other Renewal Opponents

The composite strategy results in flexibility against multiple opponents. We evaluate QFlip using composite observations against Uniform and Normal Renewal opponents in Figure 10. QFlip attains 15% better average benefit than Greedy against $\mathcal{U}_{\delta,u}$. Figure 10 also shows that QFlip attains a high average benefit of 0.76 against a $\mathcal{N}_{\mu,\sigma}$ opponent. We do not compare $\mathcal{N}_{\mu,\sigma}$ to Greedy as the numerical packages we used were unable to find the maximum local benefit from Equation (1). QFlip using composite attains average benefit within 0.01 of QFlip using oppLM (best performing observation scheme) against both opponents after 10 million ticks.

8 Conclusions

We considered the problem of playing adaptively in the FlipIt security game by designing QFlip, a novel strategy based on temporal difference Q-Learning instantiated with three different observations schemes. We showed theoretically that QFlip plays optimally against a Periodic Renewal opponent using the oppLM observation. We also confirmed experimentally that QFlip converges against Periodic and Exponential opponents, using the ownLM observation scheme in the



(a) QFlip vs. $\mathcal{U}_{\delta,u}$ with $\delta = 100$, $u = 50$ (b) QFlip vs. $\mathcal{N}_{\mu,\sigma}$ with $\mu = 100$, $\sigma = 10$

Fig. 10: QFlip's average benefit by time for Uniform (left) and Normal (right) distributions with $k_1 = 10$. QFlip with oppLM and composite observations outperforms Greedy against $\mathcal{U}_{\delta,u}$ averaged over 10 runs. Against both opponents, composite converges to oppLM as time increases.

Exponential case. Finally, we showed general QFlip with a composite observation scheme performs well against Periodic, Exponential, Uniform, and Normal Renewal opponents. Generalized QFlip is the first adaptive strategy which can play against any opponent with no prior knowledge.

We performed detailed experimental evaluation of our three observation schemes for a range of distributions parameters and move costs. Interestingly, we showed that certain hyper-parameter configurations for the amount of exploration (ϵ and d), future reward discount (γ), and probability of moving in new states ($1 - p$) are applicable against a range of Renewal strategies. Thus, QFlip has the advantage of requiring minimal configuration. Additionally, we released an OpenAI Gym environment for FlipIt to aid future researchers.

In future work, we plan to consider extensions of the FlipIt game, such as multiple resources and different types of moves. We are interested in analyzing other non-adaptive strategies besides the class of Renewal strategies. Finally, approximation methods from reinforcement learning have the potential to make our composite strategy faster to converge and we plan to explore them in depth.

Acknowledgements

We would like to thank Ronald Rivest, Marten van Dijk, Ari Juels, and Sang Chin for discussions about reinforcement learning in FlipIt. We thank Matthew Jagielski, Tina Eliassi-Rad, and Lucianna Kiffer for discussing the theoretical analysis. This project was funded by NSF under grant CNS-1717634. This research was also sponsored by the U.S. Army Combat Capabilities Development Command Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Combat Capabilities Development Command Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

1. Bowers, K.D., van Dijk, M., Griffin, R., Juels, A., Oprea, A., Rivest, R.L., Triandopoulos, N.: Defending against the unknown enemy: Applying FlipIt to system security. In: Proceedings of the Conference on Decision and Game Theory for Security. GameSec (2012)
2. Chung, K., Kamhoua, C.A., Kwiat, K.A., Kalbarczyk, Z.T., Iyer, R.K.: Game theory with learning for cyber security monitoring. In: 2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE). pp. 1–8 (Jan 2016). https://doi.org/10.1109/HASE.2016.48
3. van Dijk, M., Juels, A., Oprea, A., Rivest, R.L.: FlipIt: The game of stealthy takeover. Journal of Cryptology **26**, 655–713 (2013)
4. Elderman, R., Pater, L.J.J., Thie, A.S., Drugan, M.M., Wiering, M.: Adversarial reinforcement learning in a cyber security simulation. In: ICAART (2017)
5. Farhang, S., Grossklags, J.: FlipLeakage: A game-theoretic approach to protect against stealthy attackers in the presence of information leakage. In: Zhu, Q., Alpcan, T., Panaousis, E., Tambe, M., Casey, W. (eds.) Decision and Game Theory for Security. pp. 195–214. Springer International Publishing, Cham (2016)
6. Feng, X., Zheng, Z., Hu, P., Cansever, D., Mohapatra, P.: Stealthy attacks meets insider threats: A three-player game model. In: MILCOM 2015 - 2015 IEEE Military Communications Conference. pp. 25–30 (Oct 2015). https://doi.org/10.1109/MILCOM.2015.7357413
7. Feng, X., Zheng, Z., Mohapatra, P., Cansever, D.: A Stackelberg game and Markov modeling of moving target defense. In: Rass, S., An, B., Kiekintveld, C., Fang, F., Schauer, S. (eds.) Decision and Game Theory for Security. pp. 315–335. Springer International Publishing, Cham (2017)
8. Grossklags, J., Reitter, D.: How task familiarity and cognitive predispositions impact behavior in a security game of timing. In: 2014 IEEE 27th Computer Security Foundations Symposium. pp. 111–122 (July 2014). https://doi.org/10.1109/CSF.2014.16
9. Han, Y., Rubinstein, B.I.P., Abraham, T., Alpcan, T., De Vel, O., Erfani, S., Hubczenko, D., Leckie, C., Montague, P.: Reinforcement learning for autonomous defence in Software-Defined Networking. In: Bushnell, L., Poovendran, R., Başar, T. (eds.) Decision and Game Theory for Security. pp. 145–165. Springer International Publishing, Cham (2018)
10. Hu, P., Li, H., Fu, H., Cansever, D., Mohapatra, P.: Dynamic defense strategy against advanced persistent threat with insiders. In: 2015 IEEE Conference on Computer Communications (INFOCOM). pp. 747–755 (April 2015). https://doi.org/10.1109/INFOCOM.2015.7218444
11. Hu, Q., Lv, S., Shi, Z., Sun, L., Xiao, L.: Defense against advanced persistent threats with expert system for Internet of Things. In: Ma, L., Khreishah, A., Zhang, Y., Yan, M. (eds.) Wireless Algorithms, Systems, and Applications. pp. 326–337. Springer International Publishing, Cham (2017)

12. Klíma, R., Tuyls, K., Oliehoek, F.A.: Markov security games: Learning in spatial security problems (2016)
13. Laszka, A., Horvath, G., amd Levente Buttyán, M.F.: FlipThem: Modeling targeted attacks with FlipIt for multiple resources. In: Proceedings of the Conference on Decision and Game Theory for Security. GameSec (2014)
14. Laszka, A., Johnson, B., Grossklags, J.: Mitigating covert compromises: A game-theoretic model of targeted and non-targeted covert attacks. In: 9th International Conference on Web and Internet Economics (WINE) (2013)
15. Laszka, A., Johnson, B., Grossklags, J.: Mitigation of targeted and non-targeted covert attacks as a timing game. In: 4th International Conference on Decision and Game Theory for Security - Volume 8252. pp. 175–191. GameSec 2013, Springer-Verlag New York, Inc., New York, NY, USA (2013)
16. Maleki, H., Valizadeh, S., Koch, W., Bestavros, A., van Dijk, M.: Markov modeling of moving target defense games. In: Proceedings of the 2016 ACM Workshop on Moving Target Defense. pp. 81–92. MTD '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2995272.2995273
17. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.A.: Playing Atari with deep reinforcement learning. CoRR **abs/1312.5602** (2013)
18. Nochenson, A., Grossklags, J.: A behavioral investigation of the FlipIt game. In: 12th Workshop on the Economics of Information Security (WEIS) (2013)
19. Pham, V., Cid, C.: Are we compromised? Modelling security assessment games. In: Grossklags, J., Walrand, J. (eds.) Decision and Game Theory for Security. pp. 234–247. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
20. Reitter, D., Grossklags, J., Nochenson, A.: Risk-seeking in a continuous game of timing. In: 13th International Conference on Cognitive Modeling (ICMM) (2013)
21. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of Go with deep neural networks and tree search. Nature **529**, 484–503 (2016)
22. Sutton, R.S., Barto, A.G.: Introduction to Reinforcement Learning. MIT Press, Cambridge, MA, USA, 1st edn. (1998)
23. Tesauro, G.: Temporal difference learning and TD-Gammon. Commun. ACM **38**(3), 58–68 (Mar 1995). https://doi.org/10.1145/203330.203343, http://doi.acm.org/10.1145/203330.203343
24. Xiao, L., Li, Y., Han, G., Dai, H., Poor, H.V.: A secure mobile crowdsensing game with deep reinforcement learning. IEEE Transactions on Information Forensics and Security **13**(1), 35–47 (Jan 2018). https://doi.org/10.1109/TIFS.2017.2737968
25. Zhang, M., Zheng, Z., Shroff, N.B.: Stealthy attacks and observable defenses: A game theoretic model under strict resource constraints. In: 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP). pp. 813–817 (Dec 2014). https://doi.org/10.1109/GlobalSIP.2014.7032232
26. Zhu, M., Hu, Z., Liu, P.: Reinforcement learning algorithms for adaptive cyber defense against Heartbleed. In: Proceedings of the First ACM Workshop on Moving Target Defense. pp. 51–58. MTD '14, ACM, New York, NY, USA (2014). https://doi.org/10.1145/2663474.2663481