Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

SPECIAL ISSUE ARTICLE

Adaptiveness and Consistency of A Class of Online Ensemble Learning Algorithms

Carol Young[1] | Ningshi Yao[2] | Fumin Zhang*[2]

[1]High Consequence Automation and Robotics, Sandia National Laboratories, New Mexico, USA

[2]School of Electrical and Computer Engineering, Georgia Institute of Technology, GA, USA

Correspondence
Fumin Zhang, School of Electrical and Computer Engineering, Georgia Institute of Technology, USA. Email: fumin@atech.edu

Present Address
Room 406, Technology Square Research Bldg, 85 Fifth Street NW, Atlanta, Georgia, USA, 30308.

Summary

Expert based ensemble learning algorithms often serve as online learning algorithms for an unknown, possibly time-varying, probability distribution. Their simplicity allows flexibility in design choices, leading to variations that balance adaptiveness and consistency. This paper provides an analytical framework to quantify the adaptiveness and consistency of expert based ensemble learning algorithms. With properly selected states, the algorithms are modeled as a Markov chains. Then quantitative metrics of adaptiveness and consistency can be calculated through mathematical formulas, other than relying on numerical simulations. Results are derived for several popular ensemble learning algorithms. Success of the method has also been demonstrated in both simulation and experimental results.

KEYWORDS:
Adaptiveness and consistency analysis, Ensemble learning, Expert-based learning algorithm

1 | INTRODUCTION

Online ensemble learning algorithms have been developed to address the need to efficiently learn an unknown probability distribution (a concept) in real-time, while data is being collected. An ensemble contains a set of sub-algorithms called the "experts", each making different predictions for the concept[1]. Each expert is assigned a weight that determines the confidence that the algorithm has in its prediction. The output with the greatest confidence is then chosen by the algorithm as the overall prediction[2].

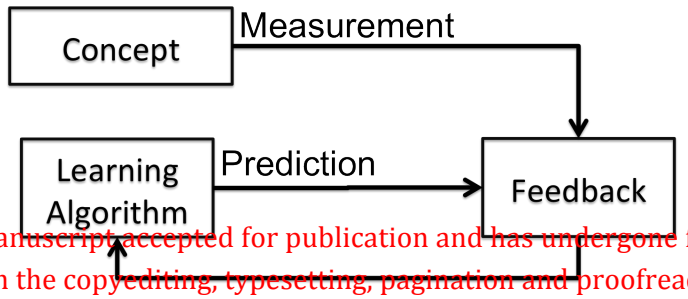


FIGURE 1 Structure of online learning

[0]The email address of Carol Young is cyoung44@gatech.edu. The emails of Ningshi Yao and Fumin Zhang are {nyao6,fumin}@gatech.edu.

The learning progresses iteratively over time. The general setup of each iteration of a learning algorithm is illustrated in Figure 1. At each time step, the algorithm makes a prediction, and then receives feedback from the concept being learned about the quality of that prediction. This feedback is then utilized by the algorithm to adjust the confidence weights of the experts, aiming to improve its prediction in the next time step. The algorithm thus can adapt to a time-varying concept. As an online learning algorithm, ensemble learning does not store historical data. Online ensembles learn each incoming training example separately, rather than in batches, and then discard it. By doing so, these approaches are able to learn the data stream iteratively, potentially being faster and requiring less memory than batch-based approaches[3]. These approaches also avoid the need for selecting an appropriate data batch size. This may reduce the problems associated with poor base models resulting from small batch sizes.

The most historically significant examples of online ensemble learning algorithms are the Weighted Majority Algorithm (WMA) and the Winnow Algorithm. Error bounds have been introduced[4,5] to evaluate the performance of these algorithms. Later algorithms, such as the Committee algorithm, have been subjected to error bound analysis and experimental verification[6,7]. In so far, error bounds computed through numerical experiments are the most commonly used metric for analyzing performance of these learning algorithms[8].

Error metrics alone are not sufficient for determining the number of steps needed for learning to converge. Previous research has shown that long learning times and rapidly changing predictions are preventing online learning methods from being effectively utilized in applications such as human robot interactions[9]. It is important to evaluate the number of steps required for the algorithm to reach a steady prediction about the concept. If the number of steps is too large, then the benefit to having learning implemented is negligible. And, if the number of steps is too small, then the algorithm may be overly sensitive to disturbances or mistakes because of rapidly changing predictions. Consider a robot running an online ensemble learning algorithm while interacting with a human. If the learning time is too long, then the robot will not be able to adapt timely to what the human wants. And if the learning time is too short, then the human might perceive the rapidly changing of outputs as erratic behavior. Progress has been made on this problem by employing an online ensemble learning algorithm called Gabe-S++ which separates its experts into groups based on the general strategy each expert employs[10]. The general strategy is then revealed to the human, which improved the human's response to the learning algorithm[11]. However this is limited to specific types of implementations and does not address the ability to determine the number of steps for learning to converge.

Main Contributions Our previous work introduced two performance metrics, called consistency and adaptiveness, for online ensemble learning algorithms with only two experts[12]. The goal of these metrics is to measure the number of steps for the algorithm to reach a steady prediction for the concept. We define consistency as a measure of how often the prediction of the algorithm changes when there is uncertainty in the concept. A large value for consistency means that the algorithm will not rapidly change predictions. We define adaptiveness as a measure of how fast an algorithm changes its prediction when the concept drifts, i.e. changes over time. A small value for adaptiveness means that the algorithm learns more quickly.

This paper generalizes our work from two-expert learning algorithms to N-expert learning algorithms. Our main contributions are the introduction of quantative metrics of adaptiveness and consistency for N-expert ensemble learning algorithms. The metrics are computed from the mean hitting steps based on Markov chain models of the ensemble learning algorithms, which have not appeared in the literature other than the two-expert case in our conference paper[12].

Related Literature Online ensemble learning algorithm have been utilized in a broad range of applications[11,13,14,15]. Recently, several comprehensive surveys on ensemble learning have been published in the literature[16,17,18].

One of the main features to distinguish between different online ensemble learning approaches for non-stationary data with concept drift is the use of concept drift detection methods. They are divided into active or passive categories. The active online ensemble approaches[19,20,21] are not so common as passive ones because of the requirement of developing explicit drift detection methods. And if concept drift detectors fail to detect drifts or present false alarms, these approaches will be unable to react to drifts. Passive approaches[22,23,3,24,25,26] are ones which do not use explicit concept drift detection methods, which are more commonly seen in literature. In this work, we will focus on the passive ensemble learning methods.

Different passive online ensembles have different strategies to assign weights to classifiers, as well as to decide when to add or remove experts from the ensemble in order to react to potential concept drifts. Most of these approaches present mechanisms to continuously adapt to concept drifts that may occur in the stream. How fast adaptation is achieved and how sensitive this adaptation is to noise usually depends on parameters. One example is the Dynamic Weighted Majority (DWM) algorithm that can add and delete experts[22]. In DWM, each expert has a weight that is reduced by a multiplicative constant when it makes a wrong prediction, similar to Littlestone and Warmuth's Weighted Majority Algorithm[27]. New experts are added when DMW

provides wrong output for a given training example. The newly added experts can learn potentially new concepts from scratch, avoiding the need for existing experts to forget their old knowledge when there is concept drift. However, DWM may not perform so well as WMA under stationary conditions which have no concept noise. Another work by Minku et al examined how different diversity levels of experts affect their generalization for new concepts[28]. This work also shows that diversity may reduce the initial increase in error caused by a drift. Their latter work[20] proposed a new ensemble learning algorithm that can adapt to concept drifts. ADWIN is another approach in which ensemble members can be reset when their accuracy degrades significantly because of the concept drifts[29]. Other approaches include regret minimization[30], tracking of the averages[31], and using an age discount factor[32] can all adjust the algorithm to adapt to the concept changes to some degrees. However, these methods were evaluated based on error rate using numerical experimentation after a drift occurs. Their adaptiveness and consistency have not been evaluated through theoretical analysis.

Relevant to consistency, one consideration is to achieve robustness to noise. Some effort has been reported in the literature. Zhu et al. propose a dynamic attribute-oriented classifier selection (AO-DCS) mechanism to integrate base experts for effective mining from data streams with a significant amount of noise[33]. In cases where experts consider multiple parameters, the confidence of each expert over the observed parameters is noted and only experts that exceed a confidence threshold are allowed to vote[34]. In cases where experts are generated in real time from a data stream, an aggregate ensemble method is used to combine multiple frameworks for generating experts[35]. These algorithms have a reduced reaction to noise and less prediction changes. However, the concept of consistency was not explicitly considered, and while there was consideration of error bound, there was no theoretical analysis of robustness or consistency.

In summary, previous results on evaluating adaptiveness and consistency are very limited. The performance of an ensemble learning algorithm in non-stationary environments with concept drifts and noise typically is usually evaluated by introducing data streams with artificially generated concept drifts. This makes it difficult to provide an in-depth understanding of the behaviour of an ensemble learning algorithm[8]. Our work proposes a theoretical analysis method to evaluate adaptiveness and consistency and make comparison among different ensemble learning algorithms. And the Markov chain models provide a more systematic way to study ensemble learning behaviour.

The paper is organized as follows. Section II reviews three ensemble learning algorithms and defines the problem of adaptiveness and consistency analysis. Section III introduces the Markov chain models of the three ensemble learning algorithms. Section IV introduces the metrics for consistency and adaptiveness and presents the formula of these metrics for the three algorithms. Section V shows simulation and experimental results to support our analysis. Section VI provides conclusions.

2 | ENSEMBLE LEARNING ALGORITHMS

To study the performance of expert based learning algorithms, we first make the following assumption regarding the “concept" being learned by these algorithms.

Assumption 1. We assume that the concept is a discrete random variable that assumes integer values $1, 2, ..., N$. It is then described by a probability distribution p_i where $i = 1, 2, ..., N$. We also assume that the expert indexed by i always select the value of i as its prediction.

We consider three learning algorithms (see Figure 1) to learn the probability distribution p_i. The weighted majority algorithm (WMA) and the Winnow algorithm[4,5] are two classic expert-based learning algorithms. The reason why we study these two algorithms is because they are the origin of many passive ensemble leaning algorithms. The latter DWM[22], addictive expert ensembles (or AddExp)[36] and horse racing ensembles (or HRE)[3] algorithms are all modified versions of WMA and Winnow. Another algorithm we will analyze in this paper is the Multiple Expert Algorithm (MEA), which is an extended version of the Dual Expert Algorithm (DEA) published in our previous work[12]. We have already analyzed the adaptiveness and consistency of DEA and compared it with two experts WMA and Winnow.

The common properties for these algorithms are as follows. N experts are employed, and the ith expert will always make the prediction to select the value i for the concept. Hence the probability for the ith expert to make a correct prediction (e.g. being success) is p_i, and the probability for the ith expert to make an error is $\tilde{p}_i = 1 - p_i$. If a wrong prediction is made, then the weight associated with the expert will decrease. The algorithm will select the next prediction by comparing the weights of all experts, and the expert associated with the largest weight will always be selected to make the next prediction. The algorithm

generally starts by assigning equal weights to all experts. The three algorithms differ by how they handle weights when a correct prediction is made.

2.1 | Three Algorithms

Algorithm 1 The Weighted Majority Algorithm (WMA)

1: Set $W_1 = W_2 = ... = W_N = 0.5$
2: Choose the winner λ as min $\left(\text{argmax}\left(W_1, W_2, ..., W_N\right)\right)$
3: **if** Error **then**
4: $\quad W_\lambda = \frac{W_\lambda}{2}$
5: **else if** Success **then**
6: $\quad W_\lambda = W_\lambda$
7: **end if**

Algorithm 1 presents the pseudo code for the WMA with N experts. Line 2 indicates that WMA selects the index of the expert with the highest weight. If there are multiple experts having highest weights at an iteration, then WMA selects the one with the smallest index. After the selection is made, WMA updates the weight of the selected expert. If an error occurs, then the weight of the selected expert is divided by 2. If a success occurs, then the weight *remains the same* as the previous iteration.

Remark 1. Because all weights start with 2^{-1} and can only be decreased by a factor of 2, all weights in the WMA will be of the form 2^{-i} where $i \geq 1$ is an integer.

Algorithm 2 The Winnow Algorithm

1: Set $W_1 = W_2 = ... = W_N = 0.5$
2: Choose winner λ as min $\left(\text{argmax}\left(W_1, W_2, ..., W_N\right)\right)$
3: **if** Error **then**
4: $\quad W_\lambda = \frac{W_\lambda}{2}$
5: **else if** Success **then**
6: $\quad W_\lambda = 2W_\lambda$
7: **end if**

Algorithm 2 presents the pseudo code for the Winnow algorithm with N experts. Similar as WMA, the Winnow algorithm selects the expert with the highest weight. If there are multiple experts having the same highest weights, then Winnow selects the one with the smallest index. If an error occurs, then the weight of the selected expert is divided by 2. The difference between WMA and Winnow algorithm is shown by line 6. If a success occurs, i.e. the prediction by an expert is correct, then its weight is *multiplied by* 2.

Algorithm 3 presents the pseudo code for the MEA with N experts. The MEA has the same expert selection rule and tie breaker rule as the WMA and winnow algorithms, which is presented by line 2. Also identical to the WMA and the Winnow, if an error occurs, then the weight of the selected expert is divided by 2. The difference is that, as shown by lines 6 and 7, MEA algorithm *puts an upper bound on the weights*. If a success occurs and the weight of the selected expert is less than 0.5, then the weight is multiplied by 2. Otherwise, the weight of the selected expert remains the same. Based on these weight updating rules and the initial value of the weights, the weights of MEA are also in the form of 2^i where i is an integer.

2.2 | Problem Formulation

Our goal in this paper is to provide quantitative evaluation of the adaptiveness and consistency of the three ensemble learning algorithms. We now introduce necessary mechanisms to formulate the problems.

Algorithm 3 The Multiple Expert Algorithm (MEA)

Set $W_1 = W_2 = ... = W_N = 0.5$
Choose output λ from min $\left(\text{argmax}\left(W_1, W_2, ..., W_N\right)\right)$
if Error **then**
    $W_\lambda = \frac{W_\lambda}{2}$
else if Success **then**
    if $W_\lambda < 0.5$ **then**
        $W_\lambda = 2W_\lambda$
    else
        $W_\lambda = W_\lambda$
    end if
end if

Let α be the index where p_α is the maximum value among all indices $i = 1, 2, ..., N$. The value α is called a preference of the concept. We consider this preference as a strong preference, which is described by the following assumption:

Assumption 2. We assume that $p_\alpha > p_i, \forall i \neq \alpha$ and $p_\alpha > 0.5$.

The assumption indicates that α is unique. For convenience of notations, we rename the indices of the $(N-1)$ experts that are not α as $\beta_1, ..., \beta_{N-1}$ with $\beta_1 = \alpha + 1$ and $\beta_{N-1} = (\alpha + N - 1) \mod N$. The probabilities of all the experts satisfy that $p_\alpha + \sum_{i=1}^{N-1} p_{\beta_i} = 1$. We can rewrite this equality as $1 - p_\alpha = \sum_{i=1}^{N-1} p_{\beta_i}$. Since $p_{\beta_i} \geq 0$ for any $i = 1, ..., N-1$, we have $1 - p_\alpha = \sum_{i=1}^{N-1} p_{\beta_i} \geq p_{\beta_i} \geq 0$.

Remark 2. The assumption requires that a concept has a strong preference for value α. This assumption is usually required for ensemble learning to perform well. Under this assumption, the learning algorithms will eventually select expert α as the winning expert more often than other experts.

The problem for evaluating adaptiveness for an ensemble learning algorithm can then be formulated as follows:

Problem 1. (*evaluating adaptiveness*) Suppose the winning expert is currently not α, calculate the averaged number of iterations before an algorithm chooses α as the winning expert.

Remark 3. Under this problem formulation, adaptiveness is measured by how many iterations for the algorithm to learn the strong preference for the concept. If the number of iterations is small, then the algorithm can quickly learn the preference. This is especially preferred when the value of α suddenly changes. The algorithm can adapt to this change quickly.

On the other hand, the problem for evaluating consistency for an ensemble learning algorithm can be formulated as follows:

Problem 2. (*evaluating consistency*) Suppose the winning expert is currently α, calculate the averaged number of iterations before an algorithm chooses another expert, different from α, as the winning expert.

Remark 4. Under this problem formulation, consistency is measured by how many iterations can the algorithm tolerate before it chooses a winning expert that is not the strong preference of the concept. If the number of iterations is large, then the algorithm can tolerate large but temporary deviations from the preference.

The two problems can be solved by numerical simulations. However, numerical simulations can only provide answer on a case by case basis, and often fail to offer insights of the solutions. In this paper, we propose analytic solutions to the two problems based on Markov chain models for the three ensemble learning algorithms. These Markov chain models are also novel contributions, which will be introduced in the next section.

3 | MARKOV CHAIN MODELS

For the three ensemble learning algorithms, the states of the Markov chain models can be described using up to three state variables, λ, R and n. We define λ as the integer index of the winning expert whose prediction for the concept is also λ. The

formula to compute λ is

$$\lambda = \min\left(\text{argmax}\left(W_1, W_2, ..., W_N\right)\right). \tag{1}$$

The state variable R is defined as the ratio between the maximal weight and the minimal weight among all experts. The formula to compute R is

$$R = \begin{cases} \log_2\left(\frac{2\max(W_1,...,W_N)}{\min(W_1,...,W_N)}\right), & \text{if } \lambda = 1 \\ \log_2\left(\frac{\max(W_1,...,W_N)}{\min(W_1,...,W_N)}\right), & \text{otherwise} \end{cases} \tag{2}$$

When the prediction $\lambda = 1$, we multiply the maximal weight with 2 to normalize the ratio R. Because all the weights are in the form of 2^{-i}, R is an integer. Since the ratio $\frac{\max(W_1,...,W_N)}{\min(W_1,...,W_N)}$ is greater than or equal to 1, then $R \geq 0$.

To model the MEA, we also need to introduce another state variable n as

$$n = \begin{cases} \log_2\left(\frac{1}{\min(W_1,...,W_N)}\right), & \text{if } \lambda = 1 \\ \log_2\left(\frac{0.5}{\min(W_1,...,W_N)}\right), & \text{otherwise} \end{cases} \tag{3}$$

Here n represents the maximal achievable value of R. Because the weights in MEA have an upper bound 0.5, replacing the term $\max(W_1, ..., W_N)$ in equation (2) with 0.5 results in equation (3). Based on this definition, $R \leq n$ always holds.

3.1 | N-expert Weighted Majority Algorithm

In the following proposition, we will show that for WMA, the value of R can only be 1.

Proposition 1. The variable R equals 1 for all $\lambda = 1, ..., N$ for WMA.

Proof. We will prove this proposition by contradiction. First, we assume that R can be 0. The only case where $R = 0$ is when $\lambda \neq 1$ and $\frac{\max(W_1,...,W_N)}{\min(W_1,...,W_N)} = 1$, which means that $W_1 = W_2 = \cdots = W_N$. However, since all the weights are the same, based on line 2 of Algorithm 1, the index of the winning expert λ should be 1, which contradicts to the fact that $\lambda \neq 1$. Therefore, we have shown that $R > 0$.

Secondly, we assume that R can be 2. Due to equation (2), we need to discuss two cases. **Case 1**: $\lambda = 1$. In this case, the maximal weight is W_1. If $R = 2$, then we have $W_1 = 2\min(W_2, ..., W_N)$. Denote the smallest index of the minimal weights among $(W_2, ..., W_N)$ to be j. We have $W_1 = 2W_j$ and $j > 1$. Since the initial weights are all identical and the weight of an expert can only be decreased if this expert is selected by WMA at one iteration, $W_1 = 2W_j$ can only occur if the expert j is selected as the winning expert at one previous iteration and then be divided by 2. This means that the weights of experts 1 and j are the same but the algorithm selects expert j with $j > 1$, which contradicts to the tie breaker of WMA, i.e., line 2 in Algorithm 1. **Case 2**: $\lambda \neq 1$. In this case, we denote the smallest index of the maximal weights among $(W_2, ..., W_N)$ to be j_1 and the smallest index of the minimal weights among $(W_1, ..., W_N)$ to be j_2. Since we assume $R = 2$, we have $W_{j_1} = 4W_{j_2}$. Since the initial weights are all identical and the weight of an expert can only be decreased if this expert is selected by WMA at one iteration, $W_{j_1} = 4W_{j_2}$ can only occur if the expert j_2 is the winning expert at one previous iteration and then be divided by 2. This means that WMA selects expert j_2 with $W_{j_1} = 2W_{j_2}$ at one previous iteration, which violates line 2 in Algorithm 1. Therefore, by contradiction, we have shown that $R \neq 2$. Using similar argument, we can also show that R cannot be greater than 2.

In conclusion, R is an integer that must satisfy $R > 0$ and $R < 2$, which means that R can only be equal to 1. □

Based on Proposition 1, we can derive two corollaries which present the relations among the weights of all the experts. The corollaries will help to determine the structure of the Markov chain of WMA.

Corollary 1. For WMA, if the index of the winning expert λ is 1, then we have $W_1 = W_2 = \cdots = W_N$.

Proof. Based on equation (2), we have

$$R = \log_2\left(\frac{2\max(W_1, ..., W_N)}{\min(W_1, ..., W_N)}\right) = 1,$$

which means that $\frac{\max(W_1,...,W_N)}{\min(W_1,...,W_N)} = 1$, i.e., the maximal weights equal the minimal weights. This shows that all the weights are the same. □

Corollary 2. For WMA, if the index of the winning expert λ is greater than 1, then we have $W_1 = \cdots = W_{\lambda-1} = \frac{1}{2}W_\lambda = \cdots = \frac{1}{2}W_N$.

Proof. Since the index of the winning expert is λ, it means that for any $j \in \{1, ..., \lambda-1\}$, $W_j < W_\lambda$ must be satisfied. Otherwise, if there exists an index $j < \lambda$ and $W_j \geq W_\lambda$, then the winning expert should be j. Based on equation (2), we have

$$R = \log_2 \left(\frac{W_\lambda}{\min(W_1, ..., W_N)} \right) = 1,$$

which means that $\frac{1}{2}W_\lambda = \min(W_1, ..., W_N)$. Based on Remark 1, if a weight W_j is strictly smaller than W_λ, then $W_j \leq \frac{1}{2}W_\lambda$. Therefore, we have $\frac{1}{2}W_\lambda = \min(W_1, ..., W_N) \leq W_j \leq \frac{1}{2}W_\lambda$ for $j \in \{1, ..., \lambda-1\}$, which can only hold if $W_1 = \cdots = W_{\lambda-1} = \frac{1}{2}W_\lambda$. If $\lambda = N$, then the proof is complete.

If $1 < \lambda < N$, then we need to show that $W_\lambda = \cdots = W_N$. We also show this by contradiction. Because λ is the index of the maximal weight, we have $W_\lambda \geq W_j$ for $j \in \{\lambda+1, ..., N\}$. If there exists an index $j \in \{\lambda+1, ..., N\}$ such that $W_j < W_\lambda$, then $W_j = \frac{1}{2}W_\lambda$ since $R = 1$. Because all the initial weights are 0.5 and the weight of an expert can only be decreased if this expert is the winning expert at one iteration, $W_j = \frac{1}{2}W_\lambda$ can only occur if the expert j is selected previously and then is divided by 2. This means that the weights of experts λ and j are the same but the algorithm selects expert j as he winning expert with $j > \lambda$, which contradicts to the line 2 in Algorithm 1. Therefore, for any $j \in \{\lambda+1, ..., N\}$, $W_j = W_\lambda$. □

Utilize the two variables defined by equations (1) and (2), we can construct the Markov chain of WMA. Since R is always 1, the total number of the states of the WMA Markov chain is N. The states are $(\lambda, 1)$ where $\lambda = 1, ..., N$. We will first show how states transit when a success occurs.

Lemma 1. For all states $(\lambda, 1)$ where $\lambda \in \{1, ..., N\}$, if a success occurs, then the state remains in $(\lambda, 1)$.

Proof. Line 5 in Algorithm 1 shows that the weight of the winning expert does not change if a success occurs. Therefore, if a success occurs, the index of the winning expert remains the same at the next selection. □

Next we will show how the states of the Markov chain transit when an error occurs.

Lemma 2. For all states $(\lambda, 1)$ where $\lambda \in \{1, ..., N-1\}$, if an error occurs, then the state transits to state $(\lambda+1, 1)$.

Proof. Because of equation (2), we need to discuss two cases. **Case 1**: $\lambda = 1$. Based on Corollary 1, we have $W_1 = W_2 = \cdots = W_N$. From line 3, if an error occurs, then W_1 is divided by 2. At the next selection, the decreased weight W_1 satisfies $W_1 = \frac{1}{2}W_2 = \cdots = \frac{1}{2}W_N$. Then based on line 2, the index of the winning expert will be 2, which makes the state transit from $(1, 1)$ to $(2, 1)$.
Case 2: $\lambda \neq 1$. Based on Corollary 2, we have $W_1 = \cdots = W_{\lambda-1} = \frac{1}{2}W_\lambda = \cdots = \frac{1}{2}W_N$. If an error occurs, then W_λ is divided by 2. At the next selection, the decreased weight W_λ satisfies $W_1 = \cdots = W_\lambda = \frac{1}{2}W_{\lambda+1} = \cdots = \frac{1}{2}W_N$. Then based on line 2, the index of the winning expert will be $\lambda+1$, which makes the state transit from $(\lambda, 1)$ to $(\lambda+1, 1)$. □

Lemma 3. For the state $(N, 1)$, if an error occurs, then the state transits to $(1, 1)$.

Proof. Based on Corollary 2, we have $W_1 = \cdots = W_{N-1} = \frac{1}{2}W_N$. If an error occurs, then W_N is divided by 2. At the next selection, the decreased weight W_N satisfies $W_1 = \cdots = W_N$. Then based on line 2, the index of the winning expert will be 1, which makes the state transit from $(N, 1)$ to $(1, 1)$. □

To summarize, Lemma 1 presents the transition probabilities for each success and error. Lemma 1 presents how the states transit when a success occurs. Lemma 2 and Lemma 3 present how the states transit when an error occurs. Based on these Lemmas, the Markov chain for WMA can be constructed as shown in Figure 2.

The full transition matrix $\mathbb{P}_{WMA}$ of WMA is

$$\mathbb{P}_{WMA} = \begin{bmatrix} p_1 & \widetilde{p}_1 & 0 & \dots & 0 \\ 0 & p_2 & \widetilde{p}_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \widetilde{p}_N & 0 & 0 & \dots & p_N \end{bmatrix}. \tag{4}$$

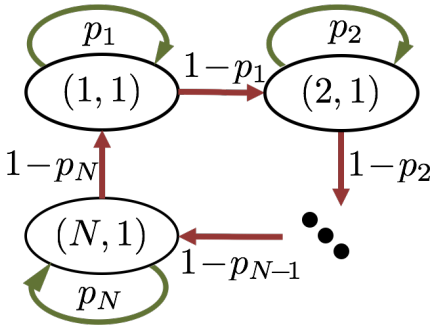


FIGURE 2 Markov chain for N-Expert WMA.

3.2 | N-expert Winnow Algorithm

We use the same variables λ and R in equations (1) and (2) to describe the states of the Markov chain of the Winnow algorithm. And same as WMA, the probability for a success to occur is p_λ and the probability for an error to occur is $\widetilde{p}_\lambda$. All weights in Winnow algorithm are of the form 2^{-i} where i is an integer. Different from WMA, i can be negative, which means that the weight of an expert can have an unbounded increase. Therefore, the variable $R \geq 1$ for Winnow is not always equal to 1 but can be infinitely large.

The following proposition reveal the relations among the weights in Winnow algorithm.

Proposition 2. For Winnow, if the index of the winning expert λ is 1, then we have $W_2 = \cdots = W_N$. If the index of the winning expert λ is greater than 1, then we have $W_1 = \cdots = W_{\lambda-1} = \frac{1}{2}W_{\lambda+1} = \cdots = \frac{1}{2}W_N$.

Proof. Since the initial weights are all equal, the expert 1 will be the winning expert at the first selection step and the weight W_1 will change after the first selection. If a success occurs, then W_1 will be multiplied by 2. If an error occurs, then W_1 will be divided by 2. Since $W_1 \geq W_2 = \cdots = W_N$, expert 1 is always the winning expert and the weight W_1 is the only weight whose value can be changed from the staring selection step until the selection step when $W_1 = \frac{1}{2}W_2 = \cdots = \frac{1}{2}W_N$. All the other weights are equal to the initial weight, i.e., the weights satisfy $W_2 = \cdots = W_N$.

Then at the selection step when $W_1 = \frac{1}{2}W_2 = \cdots = \frac{1}{2}W_N$, because $W_2 = 2W_1 > W_1$ and $W_2 = \cdots = W_N$, the expert 2 will be the winning expert. Then starting from the selection step when $W_1 = \frac{1}{2}W_2 = \cdots = \frac{1}{2}W_N$, expert 2 is the winning expert and the weight W_2 is the only weight whose value can be changed until the selection step when $W_2 = \frac{1}{2}W_3$. The weights satisfy $W_1 = W_2 = \frac{1}{2}W_3 = \cdots = \frac{1}{2}W_N$ and then starting from the selection step when $W_2 = \frac{1}{2}W_3$, expert 3 will be the winning expert.

Similar, if at a selection step, expert λ is the winning expert, then the weight W_λ is the weight whose value can be changed until the selection step when the weights satisfy $W_\lambda = \frac{1}{2}W_{\lambda+1} = \cdots = \frac{1}{2}W_N$. Then starting from the selection step when $W_\lambda = \frac{1}{2}W_{\lambda+1}$, expert $\lambda+1$ will be the winning expert.

If $\lambda = N$, all the weights are equal at the selection step when W_N is changed to be equal to W_1. Then the weights updating will repeat the above process. □

Similar to WMA, we will first show how the states of Markov chain of Winnow transit when a success occurs.

Lemma 4. For all states $[\lambda, R]$, if a success occurs, then the state transits to $[\lambda, R+1]$.

Proof. Since the current state is $[\lambda, R]$, the weight W_λ is greater than or equal to all the other weight W_j where $j \neq \lambda$ with $W_\lambda = 2^R \min_{j\in\{1,\ldots,N\}\setminus\lambda} W_j$. If a success occurs, then by line 5 of algorithm 2, we know updated weight of expert λ, denoted as W'_λ, equals $2W_\lambda$. Therefore, W'_λ is greater than all the other weight W_j where $j \neq \lambda$. Then at the next iteration, the index of the winning expert should still be λ. And $W'_\lambda = 2^{R+1} \min_{j\in\{1,\ldots,N\}\setminus\lambda} W_j$, which makes the state transit to $[\lambda, R+1]$. □

Based on Proposition 2, we will show how the states of the Markov chain of Winnow transit when an error occurs.

Lemma 5. For all states (λ, R) with $R \geq 2$, if an error occurs, then the state transits to state $(\lambda, R-1)$.

Proof. Based on Proposition 2, if $\lambda = 1$, we have $W_1 = 2^{R-1}W_j$ where $j \in \{2, ..., N\}$. If an error occurs, then by line 3 of algorithm 2, W_1 is decreased by a factor of 2. Then the newly updated weight $W_1 = 2^{R-2}W_j$ for $j \in \{2, ..., N\}$. Since $R \geq 2$,

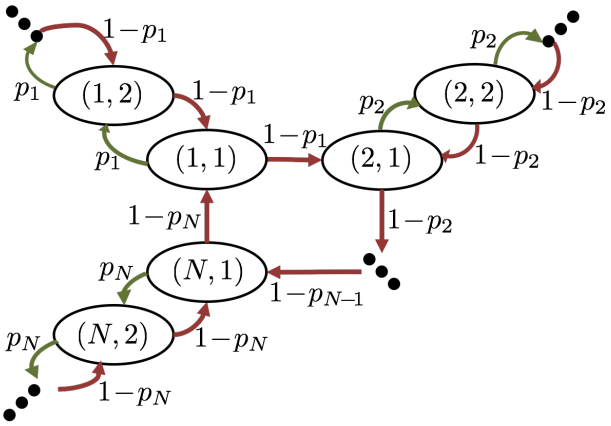


FIGURE 3 Markov chain for N-Expert Winnow Algorithm.

the coefficient $2^{R-2} \geq 1$, which means that $W_1 \geq W_j$ for $j \in \{2, ..., N\}$. At the next selection, expert 1 will be selected with R reduced by 1, which makes the new state $(1, R-1)$.

If $\lambda = N$, then we have $W_N = 2^R \min(W_1, ..., W_{N-1})$. Based on Proposition 2, we have $W_N = 2^R W_j$ where $j \in \{1, ..., N-1\}$. If an error occurs, W_N is decreased by a factor of 2. Then the newly updated weight $W_N = 2^{R-1} W_j$ for all $j < N$. Since $R \geq 2$, $W_N \geq 2W_j$ for all $j < N$. At the next selection, expert N will be selected with R reduced by 1, which makes the new state $(N, R-1)$.

If $\lambda \neq 1$ and $\lambda < N$, we have $W_1 = 2^R \min(W_1, ..., W_N)$. Based on Proposition 2, we have $W_1 = 2^R W_j$ for $j < \lambda$ and $W_1 = 2^{R-1} W_j$ for $j > \lambda$. If an error occurs, W_λ is decreased by a factor of 2. Then the newly updated weight $W_\lambda = 2^{R-2} W_j$ for $j > \lambda$. Since $R \geq 2$, $W_\lambda \geq W_j$ for all $j \neq \lambda$. At the next selection, expert λ will be selected with R reduced by λ, which makes the new state $(\lambda, R-1)$. □

Lemma 6. For all states (λ, R) with $\lambda \in \{1...N-1\}$ and $R = 1$, if an error occurs, then the state transits to $(\lambda + 1, 1)$.

Proof. This is the same case as Lemma 2. □

Lemma 7. For the state $(N, 1)$, if an error occurs, then the state transits to $(1, 1)$.

Proof. This is the same case as Lemma 3. □

In summary, Lemma 1 shows the transition probabilities for each success and error. Lemma 4 presents how states transit when a success occurs. Lemma5, 6, and 7 present how states transit when an error occurs. Based on these Lemmas, the Markov chain for Winnow can be constructed as shown in Figure 3.

3.3 | Multi-Expert Algorithm

We use variables λ, R and n to model the states of Markov chain for MEA. We first present how states transit if a success occurs.

Lemma 8. For any state (λ, R, n) with $R < n$, if a success occurs, then the state transits to state $(\lambda, R+1, n)$.

Proof. If $R < n$, then we have $W_\lambda < 0.5$. And since λ is the winning expert, the weight W_λ is greater than or equal to all the other weights. If a success occurs, then by line 7 of algorithm 3, then W_λ will be doubled and the updated weight W_λ will be greater than all the other weights. At the next selection, λ will be the winning expert. And since the only changed weight is W_λ, the value of $\min(W_1, ..., W_N)$ is unchanged. Therefore, n does not change and R is increased by 1, which makes the state transit to state $(\lambda, R+1, n)$. □

Lemma 9. For any (λ, R, n) with $R = n$, if a success occurs, then the state remains at (λ, R, n).

Proof. If $R = n$, then we have $W_\lambda = 0.5$. If a success occurs, then by line 9 of algorithm 2, then W_λ will remain the same. With all the weights unchanged at the next selection, the state also remains the same. □

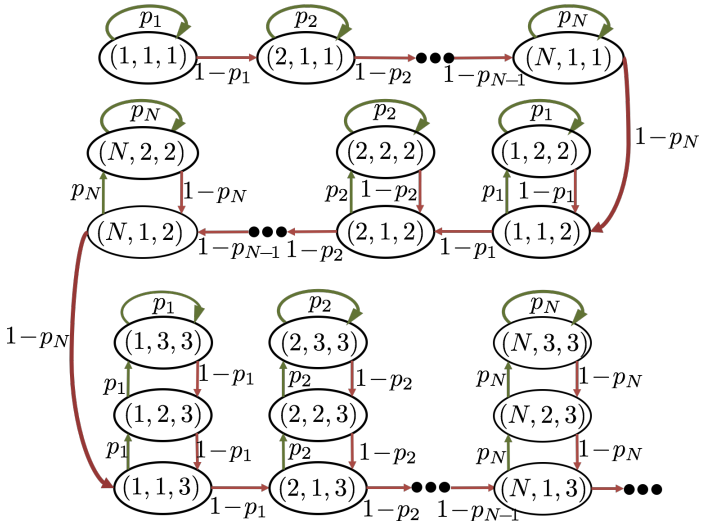


FIGURE 4 Markov chain for N-Expert MEA.

Lemma 10. For all states (λ, R, n) with $R \geq 2$, if an error occurs, then the state transits to $(\lambda, R-1, n)$

Proof. The variables λ and R follow the same proof as Lemma 5. For the variable n, since the only changed weight is W_λ and the updated W_λ is still greater than or equal to the other weights, the value of $\min(W_1, ..., W_N)$ is unchanged. Therefore, the value of n also does not change. □

Lemma 11. For all states (λ, R, n) with $\lambda \in \{1...N-1\}$ and $R = 1$, if an error occurs, then the state transits to $(\lambda+1, 1, n)$.

Proof. The variables λ and R follow the same proof as Lemma 6. For the variable n, if $\lambda = 1$, then the updated W_1' equals $\frac{1}{2}\min(W_1, ..., W_N)$. Then after the next selection, with $\lambda = 2$ and $\min(W_1', ..., W_N) = \frac{1}{2}\min(W_1, ..., W_N)$, the value of n does not change because the numerator and denominator of equation (3) are both divided to its half. If $\lambda > 1$, then the updated W_λ equals $\min(W_1, ..., W_N)$. Because of $\lambda > 1$ and $\lambda + 1 > 1$, the numerator in equation (3) remains 0.5. Therefore, the value of n remains the same. □

Lemma 12. For states $(N, 1, n)$, if an error occurs, then the state transits to $(1, 1, n+1)$.

Proof. If $\lambda = N$, then $W_N > W_j$ for $j \in \{1, ..., N-1\}$. In addition, since $R = \log_2\left(\frac{\max(W_1...W_N)}{\min(W_1...W_N)}\right) = 1$ which implies $W_N = 2\min(W_1, ..., W_N)$. Combining these two results, we have $W_N = 2W_1 = ... = 2W_{N-1}$. Then if an error occurs, W_N is divided by 2, resulting in the updated weight W_N'. The updated weight W_N' satisfies $W_N' = W_1 = \cdots = W_{N-1}$. And imputing this into equations (1) and (2), we have $\lambda = 1$ and $R = 1$. For variable n, the value of $\min(W_1, ..., W_N')$ equals $\min(W_1, ..., W_N)$. However, since the winning expert changes to be $\lambda = 1$, the numerator in equation (3) changes to be 1 instead of 0.5. Therefore, the value n becomes $n+1$, resulting in the transition from state $(N, 1, n)$ to state $(1, 1, n+1)$. □

The Markov chain for MEA can be constructed as shown by Figure 4. As we can see, the structure of MEA is more complicated than WMA and Winnow because MEA has relatively most complicated updating rules for weights.

4 | ADAPTIVENESS AND CONSISTENCY ANALYSIS

We are now ready to provide the analytic solutions for the problems of evaluating adaptiveness and consistency for the three learning algorithms. The strong preference α represents the states in the Markov chain modes where $\lambda = \alpha$. Then the number of iterations to transit to or from these states can be calculated as the mean hitting steps between certain subsets of states on the Markov chains.

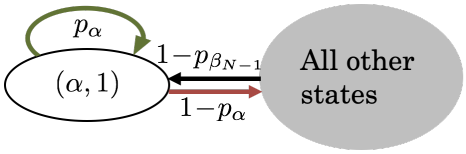


FIGURE 5 The partial Markov chain for N-Expert WMA. The blue circle represents the set A and the grey circle represents the set B.

4.1 | Mean Hitting Steps

Let the function $h_A(k)$ be defined as the mean hitting steps[37] for the state of a Markov chain starting from state k and eventually move to the subset A that contains some states. Obviously $h_A(k) = 0$, if the state k belongs to A. Let $\mathbb{P}$ be the transition matrix associated with the Markov chain. The number of rows and columns in $\mathbb{P}$ corresponding to the number of states in the Markov chain. After all the rows and columns corresponding to the states in A have been removed from the transition matrix $\mathbb{P}$ of the Markov chain, we obtain a sub-matrix $P_{/A}$, called the partial transition matrix. Let $\mathbf{h}_A$ be the column vector formed by all the values of $h_A(k)$ for all the state k that are not in A. Then the mean hitting steps can be calculated by

$$\mathbf{h}_A = (I - P_{/A})^{-1}\mathbf{1} \tag{5}$$

where I is the identity matrix, and $\mathbf{1}$ is a column vector with each element equals to 1. The dimension of $\mathbf{h}_A$ and $\mathbf{1}$ equals to the number of states that are not in the subset A.

For our purpose, we define the subset A as the set which contains all the states where $\lambda = \alpha$. We also define the subset B as the set which contains all the states where $\lambda \neq \alpha$. Then the set $A \bigcup B$ contains all the states in a Markov chain that models one of the three algorithms. We will first construct the partial transition matrices $P_{/A}$ and $P_{/B}$, and then compute the mean hitting times $\mathbf{h}_A$ and $\mathbf{h}_B$ using equation (5), which will be used as the metric for adpativeness and consistency analysis.

For WMA, since shown by Figure 5, A contains only one state $(\alpha, 1)$. After A is removed, the partial transitions matrix is

$$P_{/A} = \begin{bmatrix} p_{\beta_1} & \widetilde{p}_{\beta_1} & 0 & \dots & 0 \\ 0 & p_{\beta_2} & \widetilde{p}_{\beta_2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & p_{\beta_{N-1}} \end{bmatrix}. \tag{6}$$

Based on equations (5) and (6), we can compute the mean hitting steps from states in the set B to the set A as

$$\begin{aligned} \mathbf{h}_A &= \begin{bmatrix} 1-p_{\beta_1} & -\widetilde{p}_{\beta_1} & 0 & \dots & 0 \\ 0 & 1-p_{\beta_2} & -\widetilde{p}_{\beta_2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1-p_{\beta_{N-1}} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} \widetilde{p}_{\beta_1} & -\widetilde{p}_{\beta_1} & 0 & \dots & 0 \\ 0 & \widetilde{p}_{\beta_2} & -\widetilde{p}_{\beta_2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \widetilde{p}_{\beta_{N-1}} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{\widetilde{p}_{\beta_1}} & \frac{1}{\widetilde{p}_{\beta_2}} & \frac{1}{\widetilde{p}_{\beta_3}} & \dots & \frac{1}{\widetilde{p}_{\beta_{N-1}}} \\ 0 & \frac{1}{\widetilde{p}_{\beta_2}} & \frac{1}{\widetilde{p}_{\beta_3}} & \dots & \frac{1}{\widetilde{p}_{\beta_{N-1}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{\widetilde{p}_{\beta_{N-1}}} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \\ \sum_{i=2}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \\ \vdots \\ \sum_{i=N-1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \end{bmatrix} \end{aligned} \tag{7}$$

In the mean hitting steps, we will take the first element in $\mathbf{h}_A$ as the metric to quantify adaptiveness, which is

$$\mathbf{h}_A(1) = \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}}. \tag{8}$$

For WMA, after B is removed, the only state left is $(\alpha, 1)$. Then $P_{/B}$ is an 1×1 matrix as

$$P_{/B} = \left[p_\alpha\right]. \tag{9}$$

Based on equations (5) and (9), we can compute the mean hitting steps from the state $(\alpha, 1)$ to the set B as

$$\mathbf{h}_B = \frac{1}{1 - p_\alpha} = \frac{1}{\widetilde{p}_\alpha}. \tag{10}$$

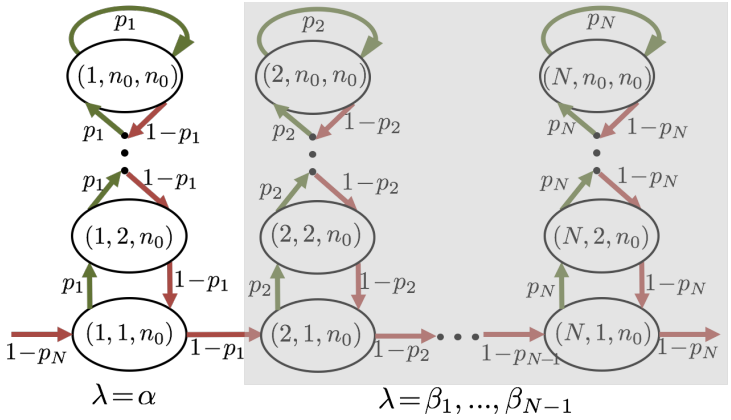


FIGURE 6 Partial Markov chains of MEA with $n = n_0$. The circles not in the grey area represent the states in set A_{n_0} with preferred output and the circles in the grey area represent the states in set B_{n_0}.

As we can see from the Markov chain models, Winnow is a sub-chain of MEA when n goes to infinity. Therefore, we will first presents formula to the meaning hitting steps for MEA in the following part and then present the results for Winnow by letting n go to infinity.

For each n, we define A_n to be the set that contains all the states (α, R, n) and B_n to be the set that contains all the states (λ, R, n) where $\lambda \neq \alpha$. The sets A_n and B_n with $n = n_0$ are illustrated by Figure 6. For an n, we will first introduce the partial transition matrix P_i^n for the partial Markov chain consisting of states (i, R, n) for fixed i and n.

$$P_i^n = \begin{bmatrix} 0 & p_i & 0 & \dots & 0 & 0 & 0 \\ 1-p_i & 0 & p_i & \dots & 0 & 0 & 0 \\ 0 & 1-p_i & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & p_i & 0 \\ 0 & 0 & 0 & \dots & 1-p_i & 0 & p_i \\ 0 & 0 & 0 & \dots & 0 & 1-p_i & p_i \end{bmatrix} \tag{11}$$

where P_i^n is an $n \times n$ matrix. We also define a matrix C_i^n to be

$$C_i^n = \begin{bmatrix} 1-p_i & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} \tag{12}$$

which is also an $n \times n$ matrix. Then the partial transition matrix $\mathbb{P}^n$ for the partial Markov chain consisting of states (i, R, n) for $i = 1, ..., N$ under a fixed n is

$$\mathbb{P}^n = \begin{bmatrix} P_1^n & C_1^n & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & P_2^n & C_2^n & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & P_3^n & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & P_N^n \end{bmatrix} \tag{13}$$

where $\mathbf{0}$ is an $n \times n$ matrix with all elements equal to 0.

After A_n is removed, the partial transitions matrix is

$$P_{/A_n}^n = \begin{bmatrix} P_{\beta_1}^n & C_{\beta_1}^n & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & P_{\beta_2}^n & C_{\beta_2}^n & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & P_{\beta_3}^n & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & P_{\beta_{N-1}}^n \end{bmatrix} \tag{14}$$

Based on equation (5), to compute the mean hitting time from states in the set B_n to the set A_n, we need to compute $\left(I - P^n_{/A_n}\right)^{-1}$.

Lemma 13. The inverse of matrix $I - P^n_{/A_n}$ is

$$\begin{bmatrix} \left(I - P^n_{\beta_1}\right)^{-1} & Q^n_{\beta_2} & Q^n_{\beta_3} & \cdots & Q^n_{\beta_{N-1}} \\ \mathbf{0} & \left(I - P^n_{\beta_2}\right)^{-1} & Q^n_{\beta_3} & \cdots & Q^n_{\beta_{N-1}} \\ \mathbf{0} & \mathbf{0} & \left(I - P^n_{\beta_3}\right)^{-1} & \cdots & Q^n_{\beta_{N-1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \left(I - P^n_{\beta_{N-1}}\right)^{-1} \end{bmatrix}$$

where $Q^n_{\beta_i}$ is an $n \times n$ matrix as

$$Q^n_{\beta_i} = \begin{bmatrix} \frac{1}{\widetilde{p}_{\beta_i}} & \cdots & \frac{p^{j-1}_{\beta_i}}{\widetilde{p}^j_{\beta_i}} & \cdots & \frac{p^{n-1}_{\beta_i}}{\widetilde{p}^n_{\beta_i}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{1}{\widetilde{p}_{\beta_i}} & \cdots & \frac{p^{j-1}_{\beta_i}}{\widetilde{p}^j_{\beta_i}} & \cdots & \frac{p^{n-1}_{\beta_i}}{\widetilde{p}^n_{\beta_i}} \end{bmatrix}. \tag{15}$$

Proof. The proof is in Appendix A.1. □

Same as WMA, we will take the first element in $\mathbf{h}^n_A$ as the metric to measure the adaptiveness, i.e.

$$\mathbf{h}^n_A(1) = \sum_{i=1}^{N-1} \sum_{j=1}^{n} \frac{p^{j-1}_{\beta_i}}{\widetilde{p}^j_{\beta_i}} = \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \frac{1 - \left(\frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}\right)^n}{1 - \frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}}. \tag{16}$$

For the MEA, after B_n is removed, the states left are (α, R, n). Then $P^n_{/B}$ is an n by n matrix as

$$P^n_{/B_n} = \left[P^n_\alpha\right]. \tag{17}$$

For consistency, we take the first element in $\mathbf{h}_B$ as the metric. The formula is

$$\mathbf{h}^n_B(1) = \sum_{j=1}^{n} \frac{p^{j-1}_\alpha}{\widetilde{p}^j_\alpha}. \tag{18}$$

For Winnow, the sets A and B are illustrated by Figure 7. The set A contains states (α, R) with $R = 1, 2, ...$ The transition matrix is the same as $\mathbb{P}^n$ of MEA when n goes to infinity. Therefore, after A is removed, the partial transitions matrix is the limit of $P^n_{/A_n}$ as $n \to \infty$. Since $\widetilde{p}_{\beta_i} = 1 - p_{\beta_i} > 0.5 > p_{\beta_i}$, we can compute the first element in the mean hitting times as

$$\mathbf{h}^\infty_A(1) = \lim_{n\to\infty} \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \frac{1 - \left(\frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}\right)^n}{1 - \frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}} = \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i}} \frac{1}{1 - \frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}} = \sum_{i=1}^{N-1} \frac{1}{\widetilde{p}_{\beta_i} - p_{\beta_i}}. \tag{19}$$

For consistency, since $\widetilde{p}_\alpha = 1 - p_\alpha < 0.5 < p_\alpha$, the first element in $\mathbf{h}_B$ is

$$\mathbf{h}^\infty_B(1) = \lim_{n\to\infty} \sum_{j=1}^{n} \frac{p^{j-1}_\alpha}{\widetilde{p}^j_\alpha} = +\infty. \tag{20}$$

4.2 | Adaptiveness Analysis for Three Learning Algorithms

Adaptiveness is a measure of how fast an algorithm changes its prediction when the concept drifts, i.e. changes over time. We denote the measure of adaptiveness as t_A and we use the first element in $\mathbf{h}_A$ to quantify adaptiveness of the three learning algorithms. A smaller mean hitting steps from the initial state to the hitting set indicates better adaptiveness.

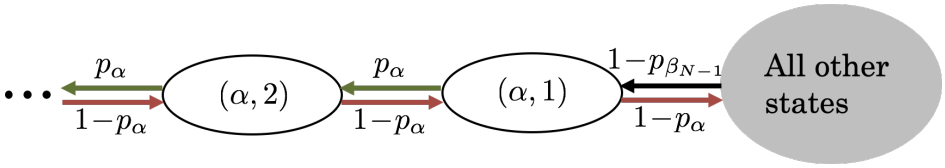


FIGURE 7 Partial Markov chain for N-Expert Winnow Algorithm. The white circles represent the states in set A and the grey circle represents the states in set B.

Furthermore, to make the metric for adaptiveness insensitive to the number of expert and non-prefered experts, we will compute the maximum values of these mean hitting steps under the possible values of the probabilities p_{β_i}. As a result, the metric for adaptiveness will only depend on the probability for the strong preference p_α.

WMA: For the WMA, its adaptiveness is measured by the maximum mean hitting steps from the initial state $(\beta_1, 1)$ to the hitting set A, $\mathbf{h}_A(1)$ in equation (8). The maximum is computed over all possible values for p_{β_i} for all i.

Proposition 3. Given p_α, the maximal value of $\mathbf{h}_A(1)$ for WMA is

$$t_{A,\text{WMA}} = \frac{1}{p_\alpha} + N - 2. \tag{21}$$

Proof. The proof is in Appendix A.2. □

Winnow: For the Winnow, its adaptiveness is measured by the maximum mean hitting steps from state $(\beta_1, 1)$ to hitting set A, $\mathbf{h}_A^\infty(1)$ in equation (19). The maximum is computed over all possible values for p_{β_i} for all i.

Proposition 4. Given p_α, the maximum value of $\mathbf{h}_A^\infty(1)$ for Winnow is

$$t_{A,\text{WIN}} = \frac{1}{p_\alpha - \widetilde{p}_\alpha} + N - 2. \tag{22}$$

Proof. The proof is in Appendix A.3. □

MEA: For the MEA, its adaptiveness is measured by the mean hitting steps from the state $(\beta_1, 1, n)$ to hitting set A_n, i.e. $\mathbf{h}_A^n(1)$, based on equation (16).

Proposition 5. Given p_α, the maximum value of $\mathbf{h}_A^n(1)$ is

$$t_{A,\text{MEA}}^n = \sum_{i=1}^{n} \left[\frac{(\widetilde{p}_\alpha)^{i-1}}{p_\alpha^i} \right] + N - 2. \tag{23}$$

Proof. The proof is in Appendix A.4. □

4.3 | Consistency Analysis for Three Learning Algorithms

Consistency is a measure of how often a learning algorithm changes its prediction in case that there is a temporary change of the concept preference. We denote the measure of consistency as t_C and we use the first element in $\mathbf{h}_B$ to quantify consistency of the three learning algorithms. A larger mean hitting time indicates a better consistency.

WMA: For the WMA, its consistency is measured by the mean hitting time from the initial state $(\alpha, 1)$ to the hitting set B. Based on equation (10), the formula to compute consistency for WMA is

$$t_{C,\text{WMA}} = \frac{1}{\widetilde{p}_\alpha}. \tag{24}$$

Winnow: For the Winnow, its consistency is measured by the mean hitting time from the initial state $(\alpha, 1)$ to the hitting set B. Based on equation (20), the formula to compute consistency for Winnow is

$$t_{C,\text{WIN}} = +\infty. \tag{25}$$

MEA: For MEA, its consistency is measured by the mean hitting time from the initial state $(\alpha, 1, n)$ to the hitting set B_n. Based on equation (18), the formula to compute consistency for MEA is

$$t^n_{C,\text{MEA}} = \sum_{i=1}^{n} \frac{p_\alpha^{i-1}}{\widetilde{p}_\alpha^{i}} = \frac{1}{\widetilde{p}_\alpha} \frac{1 - \left(\frac{p_\alpha}{\widetilde{p}_\alpha}\right)^n}{1 - \frac{p_\alpha}{\widetilde{p}_\alpha}}. \tag{26}$$

4.4 | Comparisons among the Three Learning Algorithms

When a strong preference exists and $n = 1$,

$$t^1_{C,\text{MEA}} = \frac{1}{\widetilde{p}_\alpha} \text{ and } t^1_{A,\text{MEA}} = \frac{1}{p_\alpha} + N - 2. \tag{27}$$

These equations are identical to equations (24) and (21) that give the consistency and adaptiveness of WMA.

When $n \to \infty$, since $p_\alpha > \widetilde{p}_\alpha$, the formula for an infinite sum of a geometric sequence gives

$$t^\infty_{C,\text{MEA}} = \infty \text{ and } t^\infty_{A,\text{MEA}} = \frac{1}{p_\alpha - \widetilde{p}_\alpha} + N - 2. \tag{28}$$

These equations are identical to equations (25) and (22) that give the consistency and adaptiveness of Winnow. Hence, the WMA and the Winnow algorithms can be seen as extreme cases of the MEA.

For all the finite nonzero values of n, $t^n_{A,\text{MEA}}$ and $t^n_{C,\text{MEA}}$ take values between the two extreme values. MEA has increased consistency over WMA but not Winnow and the lower bound is that n should be greater than or equal to 2. If $n = 1$, then WMA and MEA would have the same consistency because $t_{C,\text{WMA}} = t^n_{C,\text{MEA}} = \frac{1}{\widetilde{p}_\alpha}$. And if $n > 1$, the consistency of WMA is $t_{C,\text{WMA}} = \frac{1}{\widetilde{p}_\alpha}$ while the consistency of MEA is $t^n_{C,\text{MEA}} = \sum_{i=1}^{n} \frac{p_\alpha^{i-1}}{\widetilde{p}_\alpha^{i}}$ which is definitely greater than $\frac{1}{\widetilde{p}_\alpha}$ because it is a summation of strictly positive terms with the first term being $\frac{1}{\widetilde{p}_\alpha}$. This has confirmed our observations from prior simulations and experiments that the MEA is more adaptive than the Winnow, and more consistent than the WMA algorithm[38]. As the number of times that MEA changed output increases, corresponding to an increase in n, MEA becomes less adaptive and more consistent.

5 | SIMULATION AND EXPERIMENTAL RESULTS

In this section, we collected human data in a hallway passing experimental setup and utilize the Markov chain model to estimate the concept preference of each human participant, which has not been seen in the previous literature. We also utilized the collected human data to analyze the consistency of WMA, Winnow, and MEA. We also simulate WMA, Winnow, and MEA to learn a concept with manually inserted drift to further verification the adaptiveness and consistency results of the three learning algorithms. The simulation results support the analysis we presented in the previous Section.

5.1 | Learning Human Preference

The human preference data collection is set in a simple hallway with a stationary 'robot' in the center. We recruited 10 human participants. Each participant was asked to walk past the stationary robot at least 10 times. For each time, the side (left or right) when the human participant chose to pass the robot was recorded. A data set (more than 10 data points) was collected for each person. Since we collected each human participant's data in a short time window (less than one hour), we assume the human's concept does not contain any drift which means each human participant has a preferred side to pass the robot and the preference does not change during the time we conducted the experiment. However, the data may contain some noise which mean that, at some trials, a human may not pass the robot on his/her preferred side.

Among each data set of each participant, we counted how many times a human participant passing the robot on the right side, denoted as $N_{i,\text{right}}$ where $i = 1, ..., 10$ is the index of the human participant and $p_{1,i} = N_{i,\text{right}}/N_i$, which serves as the ground truth in our experiment.

For an online ensemble learning, it is not possible to store all past data and estimate the preference $p_{1,i}$ as above. However, with the Markov Chain model proposed in Section 3, we just need to leverage the initial state value and the current state value to

TABLE 1 Table of estimated $\hat{p}_{1,i}$ on experimental data

		WMA		Winnow		MEA	
Human participant	Truth $p_{1,i}$	$\hat{p}_{1,i}$	error	$\hat{p}_{1,i}$	error	$\hat{p}_{1,i}$	error
1	0.8462	1.0	0.1538	0.85	0.0038	0.85	0.0038
2	0.4167	0.0	0.4167	0.40	0.0167	0.0	0.4167
3	0.8333	1.0	0.1667	0.85	0.0167	0.85	0.0167
4	0.3333	1.0	0.6667	0.35	0.0167	0.35	0.0167
5	0.5000	0.0	0.5000	0.50	0.000	0.045	0.0500
6	0.5385	0.0	0.5385	0.55	0.0115	0.45	0.0715
7	0.7143	1.0	0.2857	0.70	0.0143	0.65	0.0643
8	0.3333	0.0	0.3333	0.35	0.0167	0.35	0.0167
9	0.5000	0.0	0.5000	0.50	0.000	0.45	0.0500
10	0.5000	0.0	0.5000	0.50	0.000	0.55	0.0500
Average Error		0.5562		0.0096		0.0756	

estimate the only unknown parameter $\hat{p}_{1,i}$ in the Markov transition matrix. To calculate $\hat{p}_{1,i}$, we use a max likelihood estimation

$$\hat{p}_{1,i} = \text{argmax}_{p_1} \mathbf{u}_0^T \mathbb{P}^k(p_{1,i}) \mathbf{u}_k \tag{29}$$

where $\mathbf{u}_0$ is the observed likelihood distribution from the initial state and $\mathbf{u}_k$ is the observed likelihood distribution at the current state at the k-th iteration. $\mathbb{P}$ is the transition matrix with parameter $p_{1,i}$ of a specific Markov chain that models one learning algorithm from WMA, Winnow or MEA and $\mathbb{P}^k$ is the k-th power of matrix $\mathbb{P}$.

Table 1 shows the estimation results based on 10 data sets, the ground truth value, and the error between the ground truth and estimated value. It can be seen that if there is no concept drift, then the estimated results based on Winnow Markov Chain model have the smallest error compared to the ground truth. Also, the estimated $\hat{p}_{1,i}$ based on MEA has a relatively accurate results, although it is not as good as Winnow. WMA is the least accurate.

We also run WMA, Winnow, and MEA to learn the human concept based on the experimental data. From the human experimental results, WMA had an average 8.2 switches per human participant during the learning process, Winnow had an average 3.4 switches and MEA had an average 6.8 switches. If more switches occur in the output from a learning algorithm, then this learning algorithm is less consistent. This result also confirms our analysis where WMA is the least consistent, followed by MEA, and Winnow is the most consistent learning algorithm among the three studied learning algorithms.

As for adaptiveness verification, since it is very hard to detect if a human has a concept drift and when the drift occur, analyzing the adaptiveness of three algorithms based on human data is very challenging. Therefore, in the next sub-session, we simulated human concepts with manually inserted concept drift to study the adaptiveness of WMA, Winnow and MEA.

5.2 | Consistency and Adaptiveness Analysis through Simulation

We assume the preferred action of the concept corresponds to the prediction of expert 1, which means $\alpha = 1$ in the simulation setup. Based on the initial weights of these learning algorithms, the initially selected expert is always expert 1 which matches with the concept the algorithms want to learn. Here we note that the variable n introduced for the MEA has the physical meaning as the number of times that a learning algorithm continuously predicted the preferred action 1. We will use this definition of n for the comparison among WMA, Winnow and MEA.

5.2.1 | Consistency

We simulated WMA and MEA, each learning algorithm with 2, 3, and 7 experts respectively. We set the probability for the preferred action, p_α, to be 0.7. That is to say, the concept has a deviation probability of 0.3. Under this deviation, we want to test whether each of these learning algorithm can consistently select expert 1. For each algorithm, We ran 400 trials for each algorithm. The simulation of a trial terminates when the value of n is greater than 6. The total time steps of all trials we ran for WMA and MEA are no more than 10000. We define the number of the time steps from the initial time to the last time before

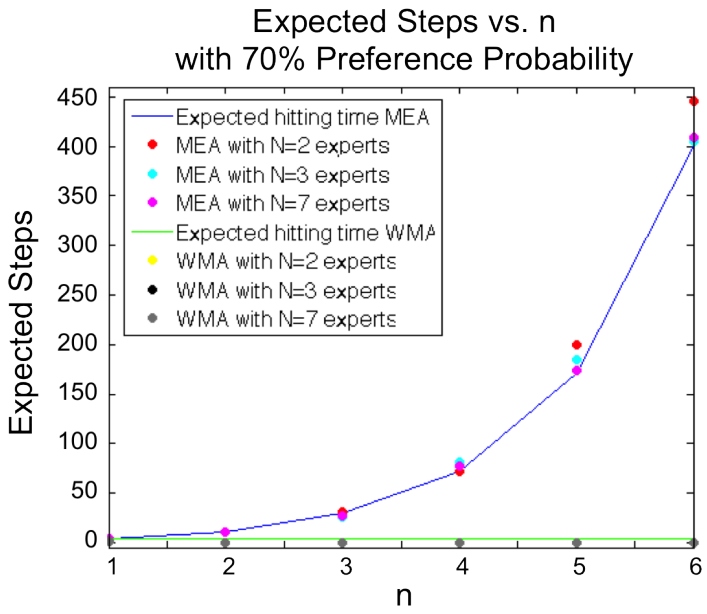


FIGURE 8 Expected and averaged steps until switching for a given n with 70% preference probability

the nth prediction switching from the action 1 to other action as t_n^i, where i represents the number of trials and $n = 1, ..., 6$. For example, if the prediction output of a trial i is $[1, 1, 1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 1, 1, ...]$, then the measured time steps t_n^i are

$$[\underbrace{1, 1, 1, 1}_{t_1^i=4}, 2, 2, \underbrace{1, 1}_{t_2^i=2}, 2, 2, \underbrace{1, 1, 1, 1, 1}_{t_3^i=5}, ...].$$

We then compute the averaged time steps t_n among 400 trials for each n, where $t_n = \frac{1}{400}\sum_{i=1}^{400} t_n^i$. The larger averaged number t_n means better consistency. Figure 8 presents the simulation results. As we can see from this figure, WMA has a smaller averaged number, meaning that WMA is less consistent. The average numbers of MEA are larger than WMA, which means that MEA has better consistency than WMA. We can also observe that an exponential increase occurs in the number of steps as the value of n increases. This also justifies our conclusion in Section 4.4 that MEA becomes more consistent as n increases. Notice that Figure 8 does not include the Winnow Algorithm. This is because the averaged time steps of Winnow is greater than 10000, which is significantly larger than the averaged time steps for WMA and MEA. Therefore, the results for Winnow are not included in this figure but are verified to be greater than the averaged time steps of MEA in our simulation. This verifies that Winnow is more consistent than MEA, which has already been discussed in our analysis.

5.2.2 | Adaptiveness

We simulated the case where the preferred action of the concept changed, i.e. $\alpha \neq 1$, after several time steps. The number of time steps before drift takes place, denoted by k, was ranged from 1 to 100 and 100 trials were simulated for each k. The deviation probability was set to be 0.3 both before, and after drift. For each trial, we counted how many time steps are needed for a learning algorithm to change its prediction from 1 to the new preferred action. Then we take the average of these time steps among the 100 trials for each k.

Figure 9 shows the average time steps of WMA, Winnow and MEA, each with two experts. The reason why only two experts are considered in this simulation is because this analysis focuses on measuring how long each type of the learning algorithm needs to leave a non-preferred state. Therefore, the case with 2 experts is enough to support the analysis.

As we can see from Figure 9, for WMA and MEA, the averaged time steps required to adapt to the new preference remains almost a constant over all k. The Winnow has a linear increase in the average time steps required to adapt as k increases. For WMA, the growth is slow because it has a small $t_{A,\mathrm{WMA}}$ and the initial distance to the switch state is always 1. For MEA, the growth is also slow, but faster than WMA. This is because $t_{A,\mathrm{MEA}}$ is limited by n which, as shown in figure 8, becomes less likely to change after each increase in n. The value of n also limits the initial distance to the switch state. For the Winnow

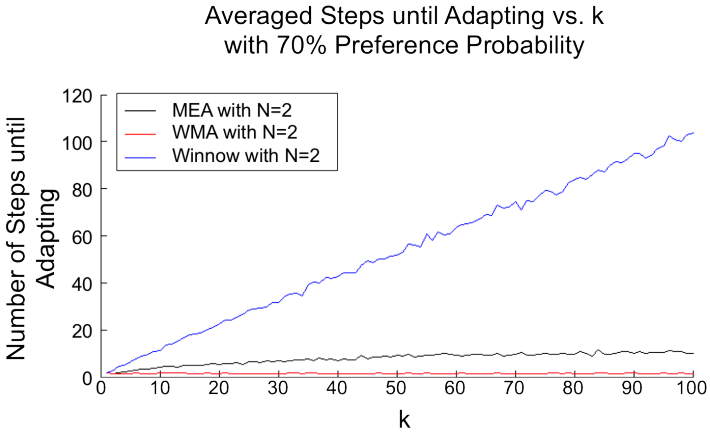


FIGURE 9 Averaged steps until adapting for a given number of steps before drift with 70% preference probability of two expert algorithms.

algorithm, the growth is much faster because it not only has a larger $t_{A,\mathrm{WIN}}$, but its initial distance to the switch state increases when the number of trials increase before the drift occurs. This shows that as the run-time of a learning algorithm increases, the Winnow algorithm becomes less adaptive and therefore it is not ideal for long-term learning. MEA, on the other hand, has better consistency than WMA, and better adaptiveness than Winnow.

6 | CONCLUSION AND FUTURE WORK

A novel approach is proposed to model simple N-Expert online ensemble learning algorithms. This model is then used to analyze consistency and adaptiveness for three learning algorithms: the WMA, the Winnow algorithm, and the MEA, when any number of experts are used, as long as the number of experts is equal to the number of possible outputs. We model the behaviors of the learning algorithms using Markov chains, and discover the connection between mean hitting times, adaptiveness and consistency. Our work is motivated by the human robot interaction experiments in our previous work[38]. Both adaptiveness and consistency are important characteristics for online learning algorithms when applied to human robot interaction. Our work has shown that Markov chain analysis can be used to quantify both characteristics. Future work consists of expanding the Markov chain analysis to the cases where the number of possible outputs is less than the number of experts used.

ACKNOWLEDGMENTS

The research work is supported by ONR grants N00014-19-1-2556 and N00014-19-1-2266; AFOSR grant FA9550-19-1-0283; NSF grants CNS-1828678, SAS-1849228 and GCR-1934836; NRL grants N00173-17-1-G001 and N00173-19-P-1412 ; and NOAA grant NA16NOS0120028.

APPENDIX

A PROOFS OF LEMMA AND PROPOSITIONS

A.1 Proof for Lemma 13

Based on the definition of the inverse of a matrix, we need to show

$$(I - P^n_{\beta_i})^{-1}(I - P^n_{\beta_i}) = I,\ i = 1, ..., N-1, \tag{A1}$$

$$-(I - P^n_{\beta_{i-1}})^{-1}C^n_{\beta_{i-1}} + Q^n_{\beta_i}(I - P^n_{\beta_i}) = \mathbf{0},\ i = 2, ..., N-1, \tag{A2}$$

$$-Q^n_{\beta_{i-1}}C^n_{\beta_{i-1}} + Q^n_{\beta_i}(I - P^n_{\beta_i}) = \mathbf{0},\ i = 3, ..., N-1. \tag{A3}$$

Equation (A1) is true based on the definition of an inverse matrix. To show equations (A2) and (A3) are true, we first need to show following lemmas.

Lemma 14. Every element in the first column of $\left(I - P^n_{\beta_i}\right)^{-1}$ is $\frac{1}{1-p_{\beta_i}}$.

Proof. If $n = 1$, then $I - P^n_\beta = 1 - p_\beta$. Thus the only element in $\left(I - P^n_{\beta_i}\right)^{-1}$ is $\frac{1}{1-p_{\beta_i}}$.

If $n = 2$, then we have

$$\left(I - P^n_{\beta_i}\right)^{-1} = \begin{bmatrix} 1 & -p_{\beta_i} \\ p_{\beta_i} - 1 & 1 - p_{\beta_i} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{1-p_{\beta_i}} & \frac{p_{\beta_i}}{(1-p_{\beta_i})^2} \\ \frac{1}{1-p_{\beta_i}} & \frac{1}{(1-p_{\beta_i})^2} \end{bmatrix}.$$

For $n > 2$, the inverse of the matrix $\left(I - P^n_{\beta_i}\right)$ is not simple to compute. Let us define

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,n} \end{bmatrix} = \left(I - P^n_{\beta_i}\right)^{-1} \tag{A4}$$

Based on the definition of an inverse, we have $(I - P^n_\beta)X = I$. Then we can solve the following n equations with variables $x_{1,1}, ..., x_{n,1}$

$$x_{1,1} - p_{\beta_i}x_{2,1} = 1, \tag{A5}$$

$$-(1 - p_{\beta_i})x_{i-1,1} + x_{i,1} - p_{\beta_i}x_{i+1,1} = 0,\ i = 2...n-1, \tag{A6}$$

$$-(1 - p_{\beta_i})x_{n-1,1} + (1 - p_{\beta_i})x_{n,1} = 0 \tag{A7}$$

From equation (A7), we can derive that $x_{n,1} = x_{n-1,1}$. And plugging this result in to the $(n-1)$-th equation, i.e. where $i = n-1$, we get $-(1 - p_{\beta_i})x_{n-2,1} + (1 - p_\beta)x_{n,1} = 0$ which means $x_{n-2,1} = x_{n,1}$. By induction, we can show that $x_{i,1} = x_{n,1}, \forall i = 1...n$. Then we can plug this in to the equation (A5) to obtain $x_{1,1} - p_{\beta_i}x_{1,1} = 1$ and thus $x_{i,1} = \frac{1}{1-p_{\beta_i}}, \forall i = 1...n$. □

Lemma 15. The first row of matrix $\left(I - P^n_{\beta_i}\right)^{-1}$ is $\left[\frac{1}{\widetilde{p}_{\beta_i}} ... \frac{p^{j-1}_{\beta_i}}{\widetilde{p}^j_{\beta_i}} ... \frac{p^{n-1}_{\beta_i}}{\widetilde{p}^n_{\beta_i}}\right]$.

Proof. If $n = 1$ or 2, the inverse matrix $\left(I - P^n_{\beta_i}\right)^{-1}$ has already been shown in the proof above. For $n > 2$, we still have $X(I - P^n_\lambda) = I$. Then we can solve n equations with variables $x_{1,1}, ..., x_{1,n}$.

$$x_{1,1} - (1 - p_{\beta_i})x_{1,2} = 1, \tag{A8}$$

$$-p_{\beta_i}x_{1,i-1} + x_{1,i} - (1 - p_{\beta_i})x_{1,i+1} = 0,\ i = 2...n-1, \tag{A9}$$

$$-p_{\beta_i}x_{1,n-1} + (1 - p_{\beta_i})x_{1,n} = 0. \tag{A10}$$

From equation (A10), we can derive that $x_{1,n} = \frac{p_{\beta_i}}{1-p_{\beta_i}}x_{1,n-1}$. And plugging this result in to the $(n-1)$-th equation, i.e. where $i = n-1$, we get

$$-p_{\beta_i}x_{1,n-2} + x_{1,n-1} - (1 - p_{\beta_i})\frac{p_{\beta_i}}{(1 - p_{\beta_i})}x_{1,n-1} = 0.$$

Rearranging this equation, we can have $-p_{\beta_i}x_{1,n-2} + (1-p_{\beta_i})x_{1,n-1} = 0$ resulting in $x_{1,n-1} = \frac{p_{\beta_i}}{1-p_{\beta_i}}x_{1,n-2}$. By induction, we can show that $x_{1,i} = \frac{p_{\beta_i}}{1-p_{\beta_i}}x_{1,i-1}, \forall i = 2...n$. Then we can plug this in to the equation (A8) to obtain $x_{1,1} - (1-p_{\beta_i})\frac{p_{\beta_i}}{1-p_{\beta_i}}x_{1,1} = x_{1,1} - p_{\beta_i}x_{1,1} = 1$, which determines $x_{1,1} = \frac{p_{\beta_i}}{1-p_{\beta_i}} = \frac{p_{\beta_i}}{\widetilde{p}_{\beta_i}}$ and $x_{1,i} = \frac{p_{\beta_i}^{i-1}}{\widetilde{p}_{\beta_i}^{i}}\forall i = 1...n$. □

Based on Lemma 14 and the definition of $C^n_{\beta_{i-1}}$ in equation (12), we can compute

$$\left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} = \begin{bmatrix} (I-P^n_{\beta_{i-1}})^{-1}_{1,1} & \dots & (I-P^n_{\beta_{i-1}})^{-1}_{1,n} \\ \vdots & \ddots & \vdots \\ (I-P^n_{\beta_{i-1}})^{-1}_{n,1} & \dots & (I-P^n_{\beta_{i-1}})^{-1}_{n,n} \end{bmatrix} \begin{bmatrix} 1-p_{\beta_{i-1}} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{1-p_{\beta_{i-1}}}{1-p_{\beta_{i-1}}} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1-p_{\beta_{i-1}}}{1-p_{\beta_{i-1}}} & 0 & \dots & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix}$$

Then we can easily show

$$\left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} \left(I - P^n_{\beta_i}\right)^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} (I-P^n_{\beta_i})^{-1}_{1,1} & \dots & (I-P^n_{\beta_i})^{-1}_{1,n} \\ \vdots & \ddots & \vdots \\ (I-P^n_{\beta_i})^{-1}_{n,1} & \dots & (I-P^n_{\beta_i})^{-1}_{n,n} \end{bmatrix}$$
$$= \begin{bmatrix} (I-P^n_{\beta_i})^{-1}_{1,1} & \dots & (I-P^n_{\beta_i})^{-1}_{1,n} \\ \vdots & \ddots & \vdots \\ (I-P^n_{\beta_i})^{-1}_{1,1} & \dots & (I-P^n_{\beta_i})^{-1}_{1,n} \end{bmatrix}$$

where every row equals the first row of matrix $(I - P^n_{\beta_i})^{-1}$. Then based on Lemma 15 and the definition of $Q^n_{\beta_i}$ in equation (15), we have

$$\left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} \left(I - P^n_{\beta_i}\right)^{-1} = Q^n_{\beta_i}.$$

Then multiplying matrix $\left(I - P^n_{\beta_i}\right)$ to both sides of the above equation, we have

$$\left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} = Q^n_{\beta_i}\left(I - P^n_{\beta_i}\right) \Rightarrow -\left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} + Q^n_{\beta_i}\left(I - P^n_{\beta_i}\right) = \mathbf{0}.$$

which shows that equation (A2) is true for $i = 2, ..., N-1$. Similarly, for equation (A3), we have

$$Q^n_{\beta_{i-1}} C^n_{\beta_{i-1}} = \begin{bmatrix} \frac{1}{\widetilde{p}_{\beta_{i-1}}} & \dots & \frac{p^{n-1}_{\beta_{i-1}}}{\widetilde{p}^n_{\beta_{i-1}}} \\ \vdots & \ddots & \vdots \\ \frac{1}{\widetilde{p}_{\beta_{i-1}}} & \dots & \frac{p^{n-1}_{\beta_{i-1}}}{\widetilde{p}^n_{\beta_{i-1}}} \end{bmatrix} \begin{bmatrix} 1-p_{\beta_{i-1}} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix} = \left(I - P^n_{\beta_{i-1}}\right)^{-1} C^n_{\beta_{i-1}} = Q^n_{\beta_i}\left(I - P^n_{\beta_i}\right)$$
$$\Rightarrow -Q^n_{\beta_{i-1}} C^n_{\beta_{i-1}} + Q^n_{\beta_i}\left(I - P^n_{\beta_i}\right) = \mathbf{0}.$$

which shows that equation (A3) is true.

A.2 Proof for Proposition 3

Define a vector $\mathbf{p}_\beta = [p_{\beta_1}, ..., p_{\beta_{N-1}}]^T$. Then $t_{A,\text{WMA}}$ can be viewed as a function of $\mathbf{p}_\beta$. Also, the probabilities in $\mathbf{p}_\beta$ must satisfy the equality constraint $\sum_{i=1}^{N-1} p_{\beta_i} = 1 - p_\alpha$. We define the constraint function $g(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} p_{\beta_i} - (1-p_\alpha)$. To find the maximal

value of $t_{A,\mathrm{WMA}}$, we need to solve an optimization problem

$$\max_{\mathbf{p}_\beta} \quad t_{A,\mathrm{WMA}}(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} \frac{1}{1-p_{\beta_i}} \tag{A11}$$
$$\text{s.t} \quad g(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} p_{\beta_i} - (1-p_\alpha) = 0.$$

Since the partial derivatives of $g(\mathbf{p}_\beta)$ are non-zero, we can utilize the Lagrange multiplier to find the extrema of (A11) using the following equations

$$\frac{\partial t_{A,\mathrm{WMA}}(\mathbf{p}_\beta)}{\partial p_{\beta_i}} - c\frac{\partial g(\mathbf{p}_\beta)}{\partial p_{\beta_i}} = \left(1-p_{\beta_i}\right)^{-2} - c = 0 \tag{A12}$$

where c is the Lagrange multiplier and $i = 1, ..., N-1$. The equation (A12) gives two solutions, $1-\sqrt{\frac{1}{c}}$ and $1+\sqrt{\frac{1}{c}}$. The solution $1+\sqrt{\frac{1}{c}}$ does not satisfy $0 \le p_{\beta_i} < 0.5$. Therefore, $p_{\beta_i} = 1-\sqrt{\frac{1}{c}}$ for $i = 1, ..., N-1$. Then we have $p_{\beta_1} = \cdots = p_{\beta_{N-1}}$. Using the equality constraints, we can compute $p_{\beta_1} = \cdots = p_{\beta_{N-1}} = \frac{1-p_\alpha}{N-1}$. Based on this result, the only extrema is

$$t_{A,\mathrm{WMA}} = (N-1)\frac{1}{1-\frac{1-p_\alpha}{N-1}} = \frac{(N-1)^2}{N-2+p_\alpha}. \tag{A13}$$

Next we consider the boundary conditions $p_{\beta_i} = 1-p_\alpha$ for some $i \in \{1, ..., N-1\}$ and $p_{\beta_j} = 0, j \neq i$. Solving $t_{A,\mathrm{WMA}}$ for these boundary conditions gives

$$t_{A,\mathrm{WMA}} = \frac{1}{1-(1-p_\alpha)} + (N-2)\frac{1}{1-0} = \frac{1}{p_\alpha} + N - 2. \tag{A14}$$

Then we need to compare the values in equation (A13) and (A14) by taking the difference of them. Let

$$\begin{aligned} e_{\mathrm{WMA}} &= \frac{1}{p_\alpha} + N - 2 - \frac{(N-1)^2}{N-2+p_\alpha} = \frac{p_\alpha + N - 2 + p_\alpha(p_\alpha+N-2)(N-2) - (N-1)^2 p_\alpha}{p_\alpha(p_\alpha+N-2)} \\ &= \frac{p_\alpha + N - 2 + p_\alpha^2(N-2) + [(N-2)^2 p_\alpha - (N-1)^2 p_\alpha]}{p_\alpha(p_\alpha+N-2)} = \frac{p_\alpha + N - 2 + p_\alpha^2(N-2) - (2N-3)p_\alpha}{p_\alpha(p_\alpha+N-2)} \\ &= \frac{(N-2)p_\alpha^2 - (2N-4)p_\alpha + N - 2}{p_\alpha(p_\alpha+N-2)} = \frac{(N-2)(p_\alpha^2 - 2p_\alpha + 1)}{p_\alpha(p_\alpha+N-2)} = \frac{(N-2)(p_\alpha-1)^2}{p_\alpha(p_\alpha+N-2)}. \end{aligned} \tag{A15}$$

Since $N \ge 2$ and $0 < p_\alpha \le 1$, we have that $(N-2)(p_\alpha-1)^2 \ge 0$ and $p_\alpha(p_\alpha+N-2) > 0$. Therefore, $e_{\mathrm{WMA}} \ge 0$. This means that $t_{A,\mathrm{WMA}}$ at the boundary is always greater than or equal to $t_{A,\mathrm{WMA}}$ at the extrema. Thus $\frac{1}{p_\alpha} + N - 2$ is the maximal value.

A.3 Proof for Proposition 4

Similarly as the proof for Proposition 3, to find the maximal value of $t_{A,\mathrm{WIN}}$, we need to solve an optimization problem

$$\max_{\mathbf{p}_\beta} \quad t_{A,\mathrm{WIN}}(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} \frac{1}{1-2p_{\beta_i}} \tag{A16}$$
$$\text{s.t} \quad g(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} p_{\beta_i} - (1-p_\alpha) = 0.$$

we can utilize the Lagrange multiplier to find the extrema of (A16) using

$$\frac{\partial t_{A,\mathrm{WIN}}(\mathbf{p}_\beta)}{\partial p_{\beta_i}} - c\frac{\partial g(\mathbf{p}_\beta)}{\partial p_{\beta_i}} = 2\left(1-p_{\beta_i}\right)^{-2} - c = 0 \tag{A17}$$

where c is the Lagrange multiplier and $i = 1, ..., N-1$. The equation (A17) gives two solutions, $1-\sqrt{\frac{2}{c}}$ and $1+\sqrt{\frac{2}{c}}$. The solution $1+\sqrt{\frac{2}{c}}$ does not satisfy $0 \le p_{\beta_i} < 0.5$. Therefore, $p_{\beta_i} = 1-\sqrt{\frac{2}{c}}$ for $i = 1, ..., N-1$. Then we have $p_{\beta_1} = \cdots = p_{\beta_{N-1}}$.

Using the equality constraints, we can compute $p_{\beta_1} = \cdots = p_{\beta_{N-1}} = \frac{1-p_\alpha}{N-1}$. Based on this result, the only extrema is

$$t_{A,\text{WIN}} = (N-1)\frac{1}{1-2\frac{1-p_\alpha}{N-1}} = \frac{(N-1)^2}{N-3+2p_\alpha}. \tag{A18}$$

For the boundary conditions $p_{\beta_i} = 1-p_\alpha$ and $p_{\beta_j} = 0, j \neq i$, we have

$$t_{A,\text{WIN}} = \frac{1}{1-2(1-p_\alpha)} + N - 2 = \frac{1}{2p_\alpha - 1} + N - 2. \tag{A19}$$

Taking the difference of (A18) and (A19), we have

$$\begin{aligned} e_{\text{WIN}} &= \frac{1}{2p_\alpha-1} + N - 2 - \frac{(N-1)^2}{N-3+2p_\alpha} = \frac{2p_\alpha+N-3+(2p_\alpha-1)[(2p_\alpha+N-3)(N-2)-(N-1)^2]}{(2p_\alpha-1)(2p_\alpha+N-3)} \\ &= \frac{2p_\alpha+N-3+(2p_\alpha-1)[(2p_\alpha(N-2)-(3N-5)]}{(2p_\alpha-1)(2p_\alpha+N-3)} = \frac{4(N-2)p_\alpha^2-2(4N-8)p_\alpha+4N-8}{(2p_\alpha-1)(2p_\alpha+N-3)} \\ &= \frac{4(N-2)(p_\alpha^2-2p_\alpha+1)}{(2p_\alpha-1)(2p_\alpha+N-3)} = \frac{4(N-2)(p_\alpha-1)^2}{(2p_\alpha-1)(2p_\alpha+N-3)} \end{aligned} \tag{A20}$$

Since $N \geq 2$ and $0.5 < p_\alpha \leq 1$, we have $(p_\alpha-1)^2 \geq 0$, $p_\alpha - 0.5 > 0$, $2p_\alpha - 1 > 0$ and $2p_\alpha + N - 3 > 0$. Therefore, $e_{\text{WIN}} \geq 0$. This means that $t_{A,\text{WIN}}$ at the boundary is always greater than or equal to $t_{A,\text{WIN}}$ at the extrema. Thus $\frac{1}{2p_\alpha-1} + N - 2$ is the maximal value for $t_{A,\text{WIN}}$.

A.4 Proof for Proposition 5

To find the maximal value of $t^n_{A,\text{MEA}}$, we need to solve an optimization problem

$$\begin{aligned} \max_{\mathbf{p}_\beta} \quad & t^n_{A,\text{MEA}}(\mathbf{p}_\beta) = \sum_{i=1}^{N-1}\sum_{j=1}^{n} \frac{p_{\beta_i}^{j-1}}{\widetilde{p}_{\beta_i}^{j}} \\ \text{s.t} \quad & g(\mathbf{p}_\beta) = \sum_{i=1}^{N-1} p_{\beta_i} - (1-p_\alpha) = 0. \end{aligned} \tag{A21}$$

We utilize the Lagrange multiplier to find the extrema of (A21) using

$$\frac{\partial t^n_{A,\text{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_i}} - c\frac{\partial g(\mathbf{p}_\beta)}{\partial p_{\beta_i}} = 0$$

Compute the first partial derivative and we can get

$$\frac{\partial t^n_{A,\text{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_i}} = \sum_{j=1}^{n} \frac{\partial\left(\frac{p_{\beta_i}^{j-1}}{\widetilde{p}_{\beta_i}^{j}}\right)}{\partial p_{\beta_i}} = \sum_{j=1}^{n} \frac{\left(p_{\beta_i}+j-1\right)p_{\beta_i}^{j-2}}{\left(1-p_{\beta_i}\right)^{j+1}} = c. \tag{A22}$$

Since $0 \leq p_{\beta_i} < 0.5$ for $i = 1...N-1$, we can conclude that $p_{\beta_1} = ... = p_{\beta_{N-1}}$ if equation (A22) holds for all i. We can prove it by contradiction. If we assume that there exist a pair of indices, k and q, such that $0 \leq p_{\beta_q} < p_{\beta_k} < 0.5$ and $\frac{\partial t^n_{A,\text{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_q}} = \frac{\partial t^n_{A,\text{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_k}} = c$, then we have $p_{\beta_q} + j - 1 < p_{\beta_k} + j - 1$, $p^j_{\beta_q} \leq p^j_{\beta_k}$ and $\left(1-p_{\beta_q}\right)^{j+1} > \left(1-p_{\beta_k}\right)^{j+1}$. Therefore, the j-th terms in equation (A22) of indices q and k satisfy

$$\frac{\left(p_{\beta_q}+j-1\right)p_{\beta_q}^{j-2}}{\left(1-p_{\beta_q}\right)^{j+1}} < \frac{\left(p_{\beta_k}+j-1\right)p_{\beta_k}^{j-2}}{\left(1-p_{\beta_k}\right)^{j+1}} \ \forall j = 1, ..., n.$$

The sum of every term results in

$$\sum_{j=1}^{n} \frac{\left(p_{\beta_q}+j-1\right)p_{\beta_q}^{j-2}}{\left(1-p_{\beta_q}\right)^{j+1}} < \sum_{j=1}^{n} \frac{\left(p_{\beta_k}+j-1\right)p_{\beta_k}^{j-2}}{\left(1-p_{\beta_k}\right)^{j+1}}.$$

This means that $\frac{\partial t^n_{A,\mathrm{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_q}} < \frac{\partial t^n_{A,\mathrm{MEA}}(\mathbf{p}_\beta)}{\partial p_{\beta_k}}$, which contradicts to our assumption. Then we have shown that $p_{\beta_1} = ... = p_{\beta_{N-1}}$. Using the equality constraints, we can compute $p_{\beta_1} = \cdots = p_{\beta_{N-1}} = \frac{1-p_\alpha}{N-1}$. Based on this result, the only extrema is

$$t^n_{A,\mathrm{MEA}} = (N-1)\sum_{j=1}^{n} \frac{\frac{1-p_\alpha}{N-1}^{j-1}}{\left(1-\frac{1-p_\alpha}{N-1}\right)^j} = \sum_{j=1}^{n} \frac{(1-p_\alpha)^{j-1}(N-1)^2}{\left(p_\alpha+N-2\right)^j}. \tag{A23}$$

For the boundary conditions $p_{\beta_i} = 1-p_\alpha$ and $p_{\beta_j} = 0, j \neq i$, we have

$$t^n_{A,\mathrm{MEA}} = \sum_{j=1}^{n} \frac{(1-p_\alpha)^{j-1}}{(1-1+p_\alpha)^j} + \sum_{k=1,k\neq i}^{N-1}\sum_{j=1}^{n} \frac{0^{j-1}}{(1-0)^j} = \sum_{j=1}^{n}\left[\frac{(1-p_\alpha)^{j-1}}{p_\alpha^j}\right] + N - 2. \tag{A24}$$

Taking the difference of (A23) and (A24), we have

$$e_{\mathrm{MEA}} = \sum_{j=1}^{n}\left[\frac{(1-p_\alpha)^{j-1}}{p_\alpha^j}\right] + N - 2 - \sum_{j=1}^{n} \frac{(1-p_\alpha)^{j-1}(N-1)^2}{\left(p_\alpha+N-2\right)^j}.$$

Separating the first terms of the two sums, we can rewrite the above equation as

$$e_{\mathrm{MEA}} = \frac{1}{p_\alpha} + \sum_{j=2}^{n}\left[\frac{(1-p_\alpha)^{j-1}}{p_\alpha^j}\right] + N - 2 - \frac{(N-1)^2}{p_\alpha+N-2} - \sum_{j=2}^{n} \frac{(1-p_\alpha)^{j-1}(N-1)^2}{\left(p_\alpha+N-2\right)^j}.$$

Then we can rearrange the equation as

$$e_{\mathrm{MEA}} = \left[\frac{1}{p_\alpha} + N - 2 - \frac{(N-1)^2}{p_\alpha+N-2}\right] + \sum_{j=1}^{n}\left[\frac{(1-p_\alpha)^j}{p_\alpha^{j+1}} - \frac{(1-p_\alpha)^j(N-1)^2}{\left(p_\alpha+N-2\right)^{j+1}}\right]. \tag{A25}$$

The first term $\frac{1}{p_\alpha}+N-2-\frac{(N-1)^2}{p_\alpha+N-2} \geq 0$ has already be shown in Proposition 3. Then we need to show the second term is also greater than or equal to 0. For the j-th term of the sum, we have

$$\frac{(1-p_\alpha)^j}{p_\alpha^{j+1}} - \frac{(1-p_\alpha)^j(N-1)^2}{\left(p_\alpha+N-2\right)^{j+1}} = \frac{1-p_\alpha}{p_\alpha^2}\left(\frac{1-p_\alpha}{p_\alpha}\right)^{j-1} - \frac{(1-p_\alpha)(N-1)^2}{\left(p_\alpha+N-2\right)^2}\left(\frac{1-p_\alpha}{p_\alpha+N-2}\right)^{j-1}.$$

Since $N \geq 2$ and $1 \geq p_\alpha > 0.5$, we can derive $p_\alpha + N - 2 \geq p_\alpha$, which leads to

$$\frac{1}{p_\alpha} \geq \frac{1}{p_\alpha+N-2} \Rightarrow \left(\frac{1-p_\alpha}{p_\alpha}\right)^{j-1} \geq \left(\frac{1-p_\alpha}{p_\alpha+N-2}\right)^{j-1}. \tag{A26}$$

To show $e_{\mathrm{MEA}} \geq 0$, we just need to show the coefficient $\frac{1-p_\alpha}{p_\alpha^2} \geq \frac{(1-p_\alpha)(N-1)^2}{\left(p_\alpha+N-2\right)^2}$ by computing

$$\begin{aligned}\frac{1-p_\alpha}{p_\alpha^2} - \frac{(1-p_\alpha)(N-1)^2}{\left(p_\alpha+N-2\right)^2} &= (1-p_\alpha)\frac{\left(p_\alpha+N-2\right)^2 - (N-1)^2p_\alpha^2}{p_\alpha^2\left(p_\alpha+N-2\right)^2} = (1-p_\alpha)\frac{\left(p_\alpha+N-2\right)^2 - [(N-1)p_\alpha]^2}{p_\alpha^2\left(p_\alpha+N-2\right)^2} \\ &= \frac{\left(Np_\alpha+N-2\right)(N-2)(1-p_\alpha)^2}{p_\alpha^2\left(p_\alpha+N-2\right)^2}.\end{aligned}$$

Since $(1-p_\alpha)^2 \geq 0$, $N-2 \geq 0$, and $Np_\alpha + N - 2 \geq 0$, we have

$$\frac{1-p_\alpha}{p_\alpha^2} \geq \frac{(1-p_\alpha)(N-1)^2}{\left(p_\alpha+N-2\right)^2} \geq 0. \tag{A27}$$

Combining equations (A25), (A26) and (A27), we can show that $e_{\mathrm{MEA}} \geq 0$. This means that $t^n_{A,\mathrm{MEA}}$ at the boundary is greater than or equal to $t^n_{A,\mathrm{MEA}}$ at the extrema. Thus $t^n_{A,\mathrm{MEA}} = \sum_{i=1}^{n}\left(\frac{\tilde{p}_\alpha^{j-1}}{p_\alpha^j}\right) + N - 2$ is the maximal value.

References

1. Wiering MA, Van Hasselt H. Ensemble algorithms in reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 2008; 38(4): 930–936.
2. Polikar R. Ensemble Learning. In: Zhang C, Ma Y., eds. *Ensemble Machine Learning*Springer US. 2012 (pp. 1-34)
3. Kuncheva L. Classifier Ensembles for Changing Environments. In: Roli F, Kittler J, Windeatt T., eds. *Multiple Classifier Systems*. 3077 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 2004 (pp. 1-15)
4. Littlestone N, Warmuth MK. The weighted majority algorithm. In: IEEE. ; 1989: 256–261.
5. Littlestone N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning* 1988; 2(4): 285–318.
6. Mesterharm C. A multi-class linear learning algorithm related to winnow. In: ; 2000: 519–525.
7. Mesterharm C, Hsu DF. Combinatorial fusion with on-line learning algorithms. In: IEEE. ; 2008: 1–8.
8. Krawczyk B, Minku LL, Gama J, Stefanowski J, Woźniak M. Ensemble learning for data stream analysis: A survey. *Information Fusion* 2017; 37: 132–156.
9. Babushkin V, Oudah M, Chenlinangjia T, Alshaer A, Crandall JW. Online Learning in Repeated Human-Robot Interactions. In: ; 2014.
10. Crandall JW. Non-myopic learning in repeated stochastic games. *arXiv preprint arXiv:1409.8498* 2014.
11. Oudah M, Babushkin V, Chenlinangjia T, Crandall JW. Learning to interact with a human partner. In: ACM. ; 2015: 311–318.
12. Young C, Khan A, Zhang F. Adaptiveness and consistency of expert based learning algorithms selecting reactions to human movements. In: IEEE. ; 2017: 1530–1535.
13. Zambelli M, Demiris Y. Online multimodal ensemble learning using self-learned sensorimotor representations. *IEEE Transactions on Cognitive and Developmental Systems* 2017; 9(2): 113–126.
14. Parikh D, Polikar R. An ensemble-based incremental learning approach to data fusion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 2007; 37(2): 437–450.
15. Xu Y, Cao X, Qiao H. An efficient tree classifier ensemble-based approach for pedestrian detection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 2011; 41(1): 107–117.
16. Sagi O, Rokach L. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2018; 8(4): e1249.
17. Gomes HM, Barddal JP, Enembreck F, Bifet A. A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)* 2017; 50(2): 1–36.
18. Woźniak M, Graña M, Corchado E. A survey of multiple classifier systems as hybrid systems. *Information Fusion* 2014; 16: 3–17.
19. Nishida K, Yamauchi K. Adaptive classifiers-ensemble system for tracking concept drift. In: . 6. IEEE. ; 2007: 3607–3612.
20. Minku LL, Yao X. DDD: A new ensemble approach for dealing with concept drift. *IEEE transactions on knowledge and data engineering* 2011; 24(4): 619–633.
21. Bifet A, Holmes G, Pfahringer B, Kirkby R, Gavaldà R. New ensemble methods for evolving data streams. In: ; 2009: 139–148.

22. Kolter JZ, Maloof MA. Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research* 2007; 8: 2755–2790.

23. Kolter JZ, Maloof MA. Using additive expert ensembles to cope with concept drift. In: ; 2005: 449–456.

24. Stanley KO. Learning concept drift with a committee of decision trees. *Informe técnico: UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA* 2003.

25. Brzezinski D, Stefanowski J. Combining block-based and online methods in learning ensembles from concept drifting data streams. *Information Sciences* 2014; 265: 50–67.

26. Yoshida Si, Hatano K, Takimoto E, Takeda M. Adaptive online prediction using weighted windows. *IEICE TRANSACTIONS on Information and Systems* 2011; 94(10): 1917–1923.

27. Littlestone N, Warmuth MK. The weighted majority algorithm. *Information and computation* 1994; 108(2): 212–261.

28. Minku LL, White AP, Yao X. The impact of diversity on online ensemble learning in the presence of concept drift. *IEEE Transactions on knowledge and Data Engineering* 2009; 22(5): 730–742.

29. Bifet A, Frank E, Holmes G, Pfahringer B. Accurate ensembles for data streams: Combining restricted hoeffding trees using stacking. In: ; 2010: 225–240.

30. Crammer K, Mansour Y, Even-Dar E, Vaughan JW. Regret Minimization With Concept Drift.. In: ; 2010: 168–180.

31. Ross GJ, Adams NM, Tasoulis DK, Hand DJ. Exponentially weighted moving average charts for detecting concept drift. *Pattern Recognition Letters* 2012; 33(2): 191–198.

32. Soares SG, Araújo R. An on-line weighted ensemble of regressor models to handle concept drifts. *Engineering Applications of Artificial Intelligence* 2015; 37: 392–406.

33. Zhu X, Wu X, Yang Y. Dynamic classifier selection for effective mining from noisy data streams. In: IEEE. ; 2004: 305–312.

34. Krawczyk B, Cano A. Online ensemble learning with abstaining classifiers for drifting and noisy data streams. *Applied Soft Computing* 2018; 68: 677–692.

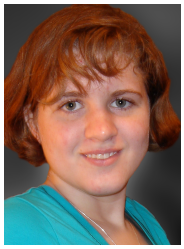
35. Zhang P, Zhu X, Shi Y, Guo L, Wu X. Robust ensemble learning for mining noisy data streams. *Decision Support Systems* 2011; 50(2): 469–479.

36. Kolter JZ, Maloof MA. Using Additive Expert Ensembles to Cope with Concept Drift. In: ICML '05. ACM; 2005; New York, NY, USA: 449–456

37. Privault N. First Step Analysis. In: Springer. 2013 (pp. 95–116).

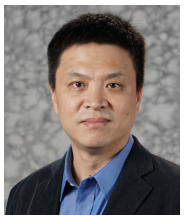
38. Young C, Zhang F. A learning algorithm to select consistent reactions to human movements. In: American Control Conference (ACC). ; 2016: 6152–6157.

AUTHOR BIOGRAPHY

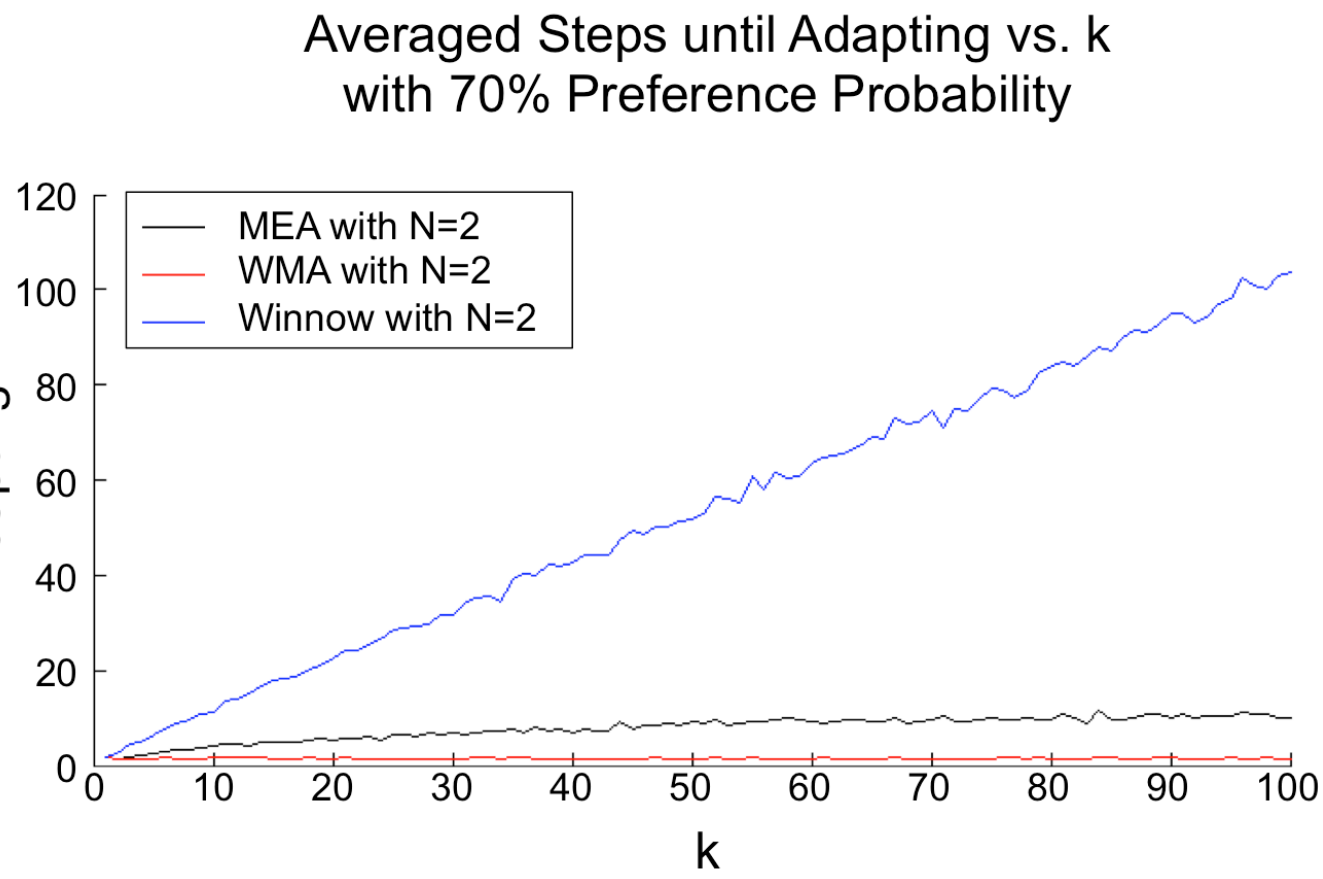


Carol Young is a postdoc at Sandia National Laboratories working in Robotics R&D. She graduated from the Georgia Institute of Technology in 2019 with a PhD in Robotics. Her research interests include machine learning, cyber-physical systems, and algorithm design and analysis.

Ningshi Yao received her Ph.D. degree from the School of Electrical and Computer Engineering at Georgia Institute of Technology, Atlanta, USA, in 2020. She received the B.S. degree from Zhejiang University, Hangzhou, China, in 2010. Her research interests include scheduling and control co-design, cyber physical system and human robot interaction.

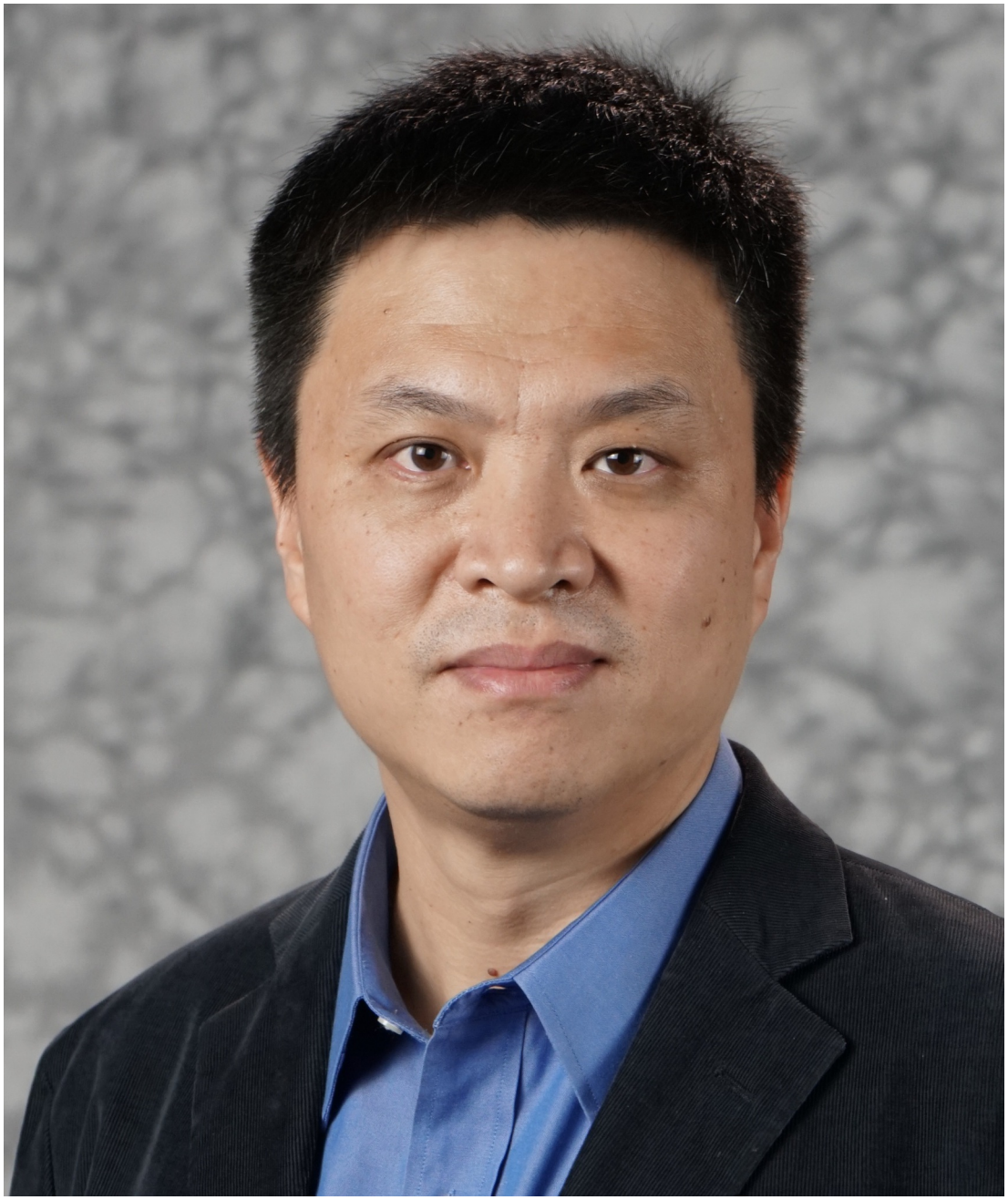


Fumin Zhang is a professor in the School of Electrical and Computer Engineering at the Georgia Institute of Technology. He received a PhD degree in 2004 from the University of Maryland (College Park) in Electrical Engineering, and held a postdoctoral position in Princeton University from 2004 to 2007. His research interests include mobile sensor networks, maritime robotics, control systems, and theoretical foundations for cyber-physical systems. He received the NSF CAREER Award in 2009 and the ONR Young Investigator Program Award in 2010.

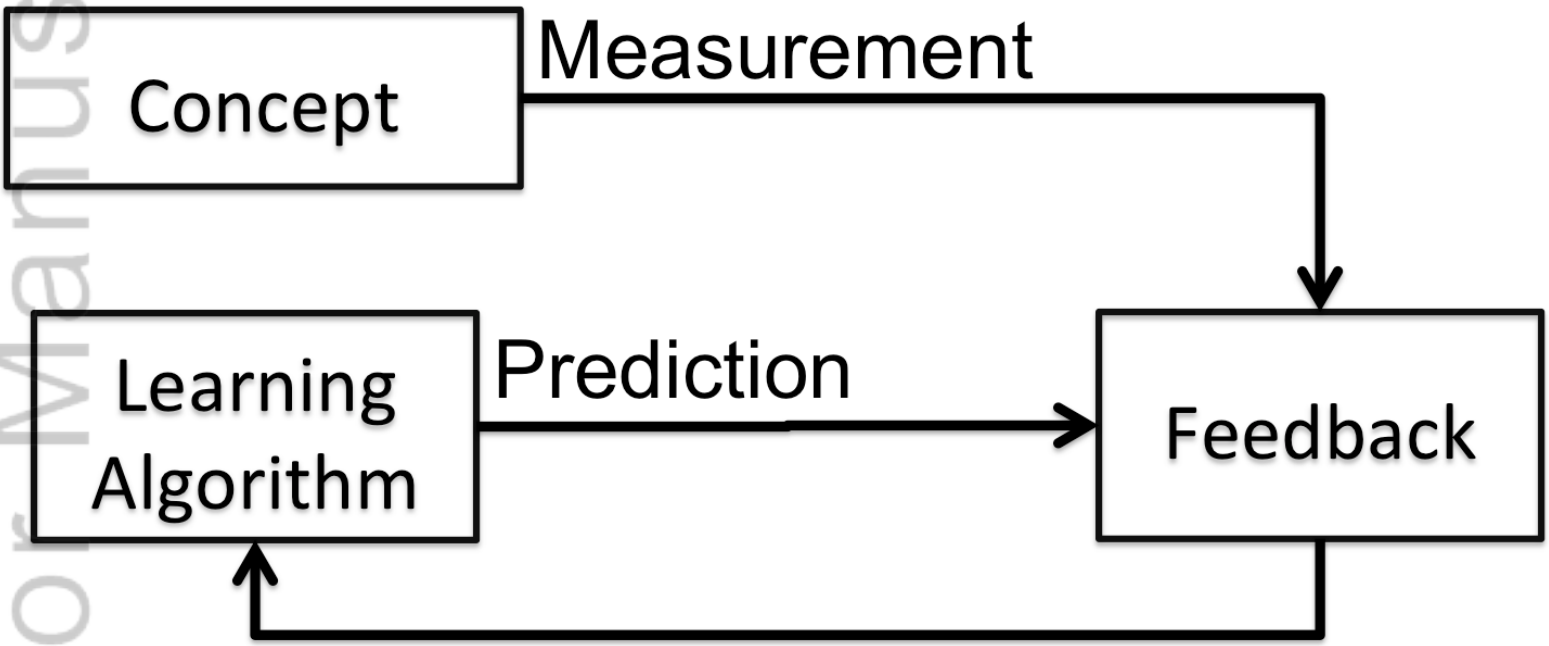


RNC_5292_Adapting_Standard.png

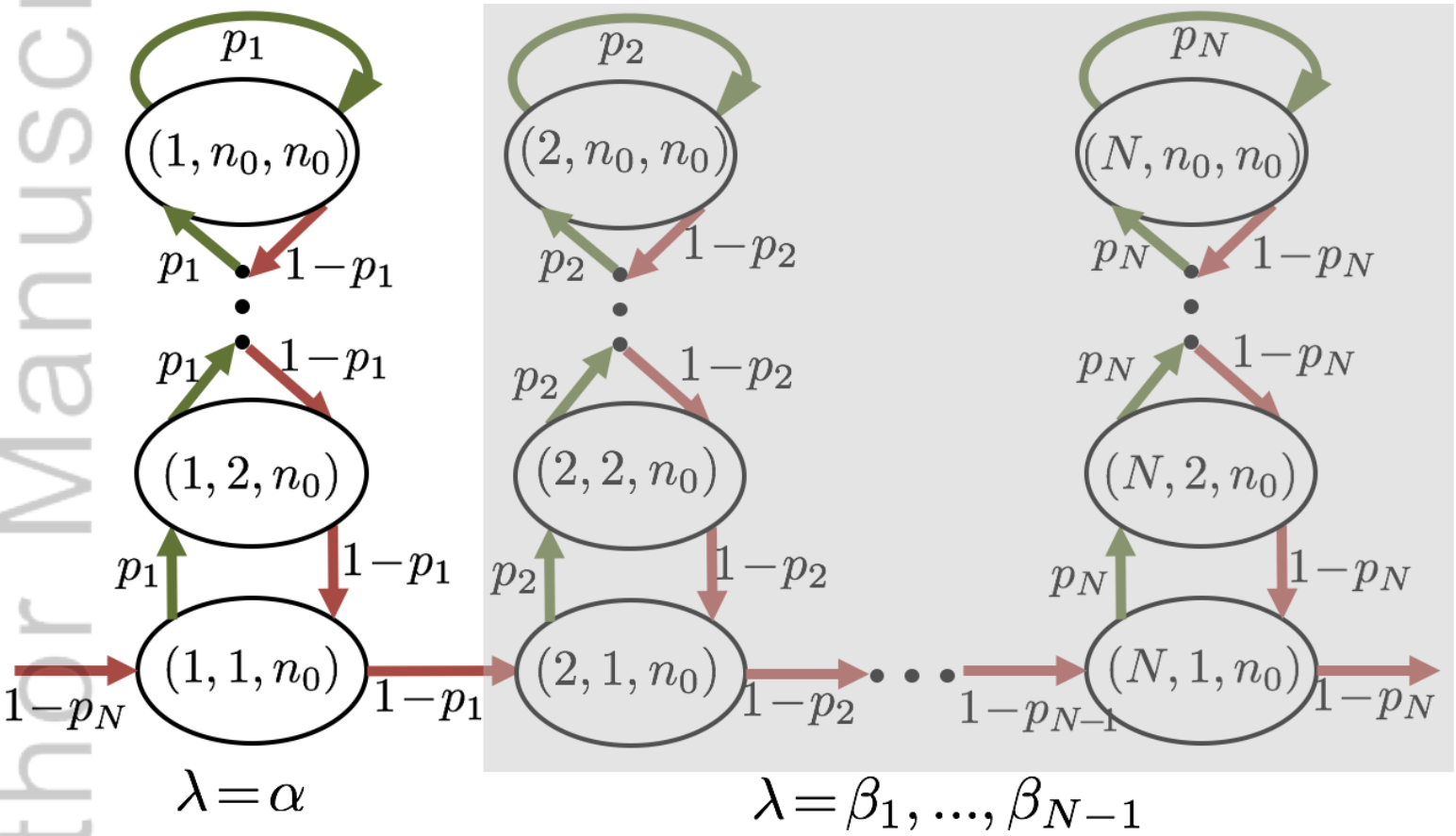
RNC_5292_Carol.png



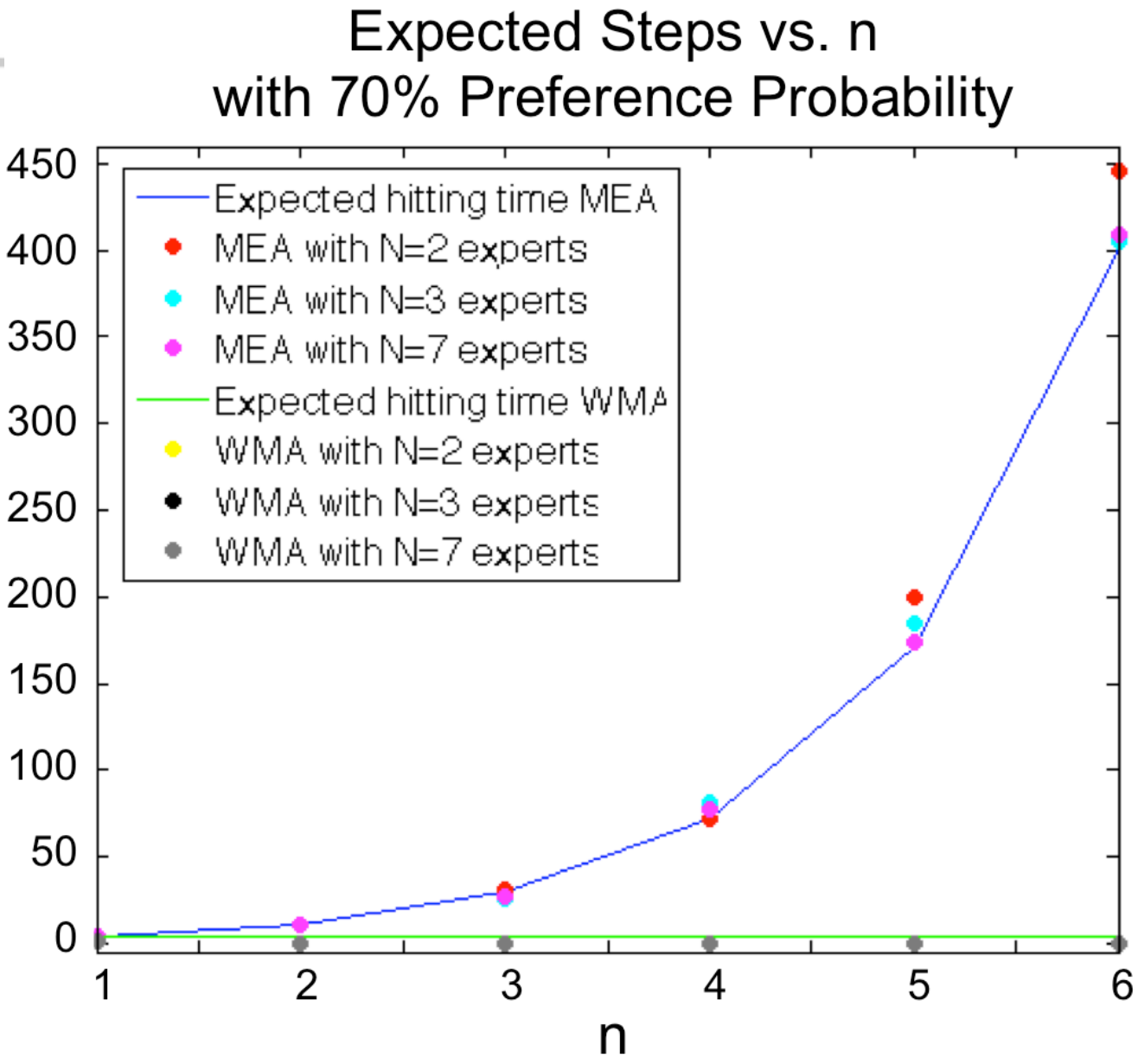
RNC_5292_Fumin_bio.jpeg



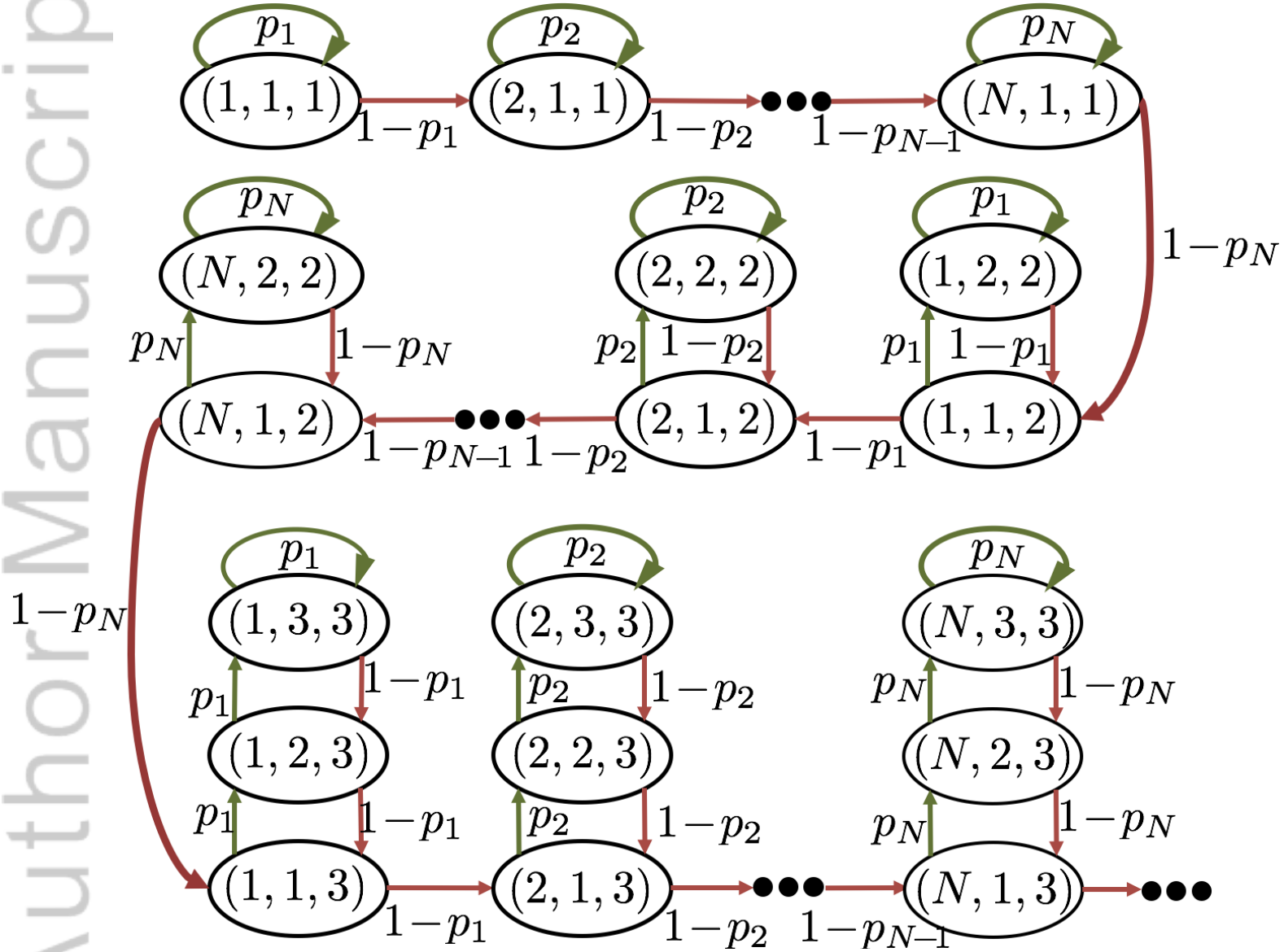
RNC_5292_LA_Journal.png



RNC_5292_MEA_Branch.png

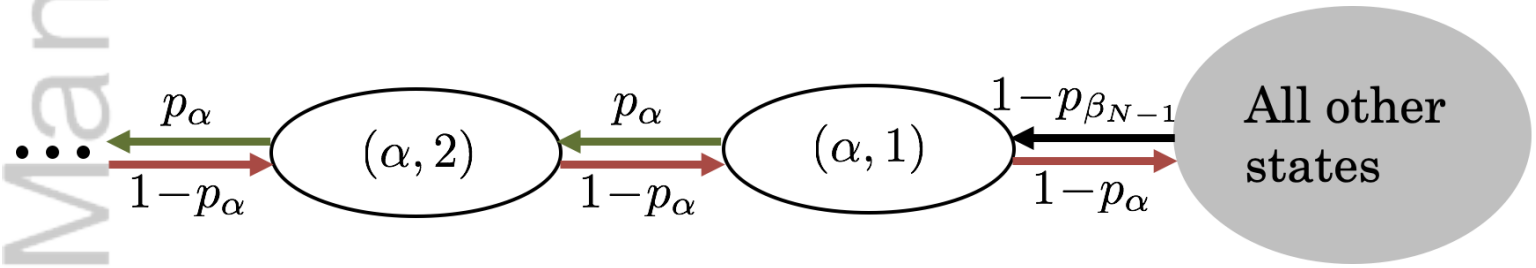


RNC_5292_MEA_CON_70.png

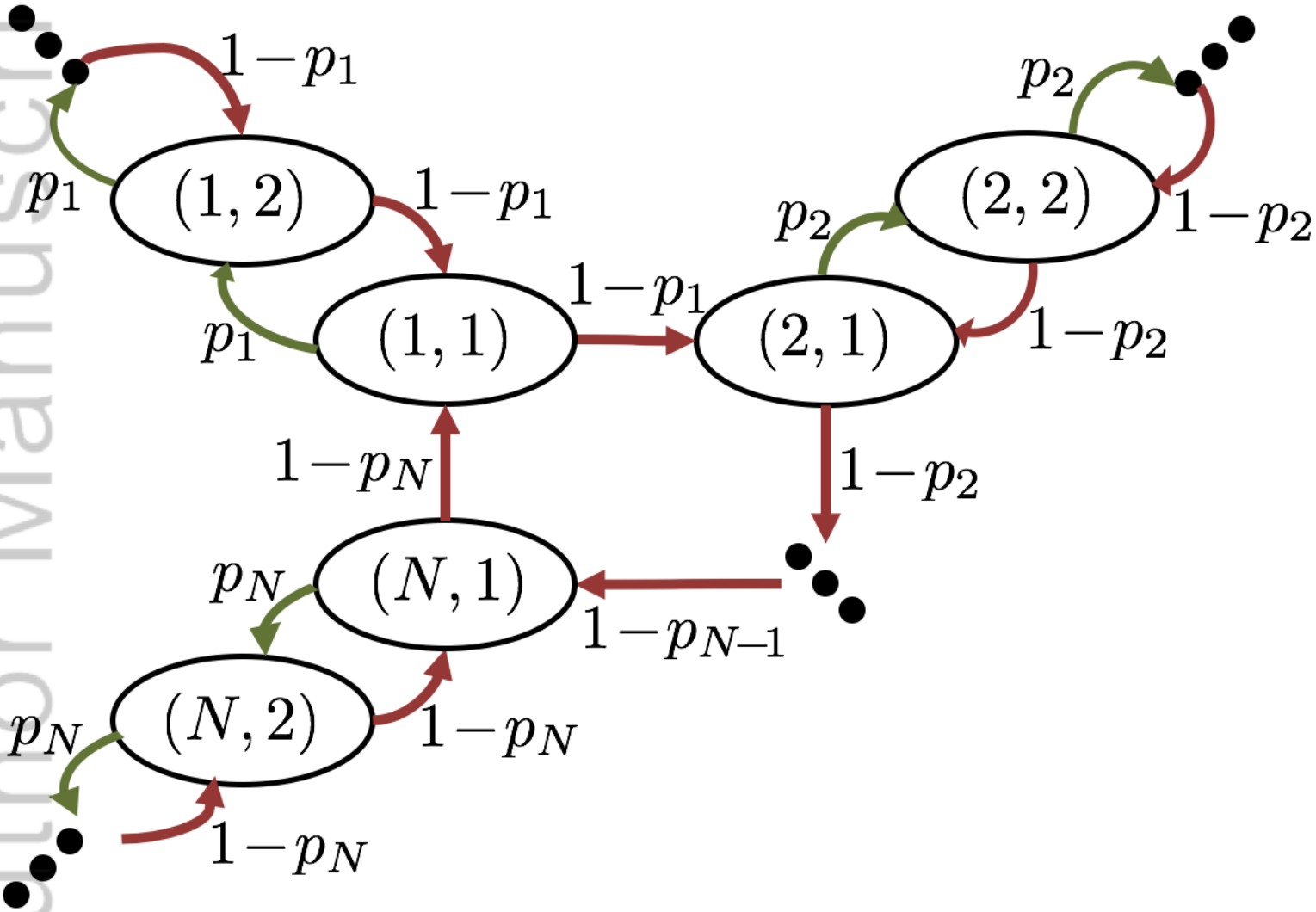


RNC_5292_MEA_FULL.png

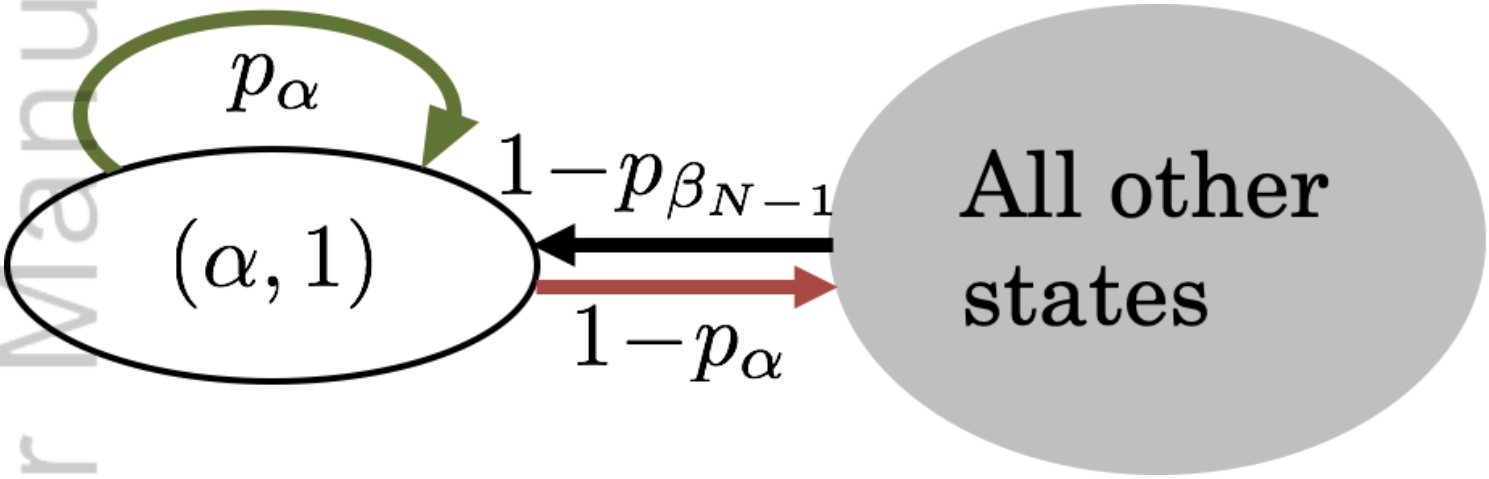
RNC_32-92_Ningshi.png



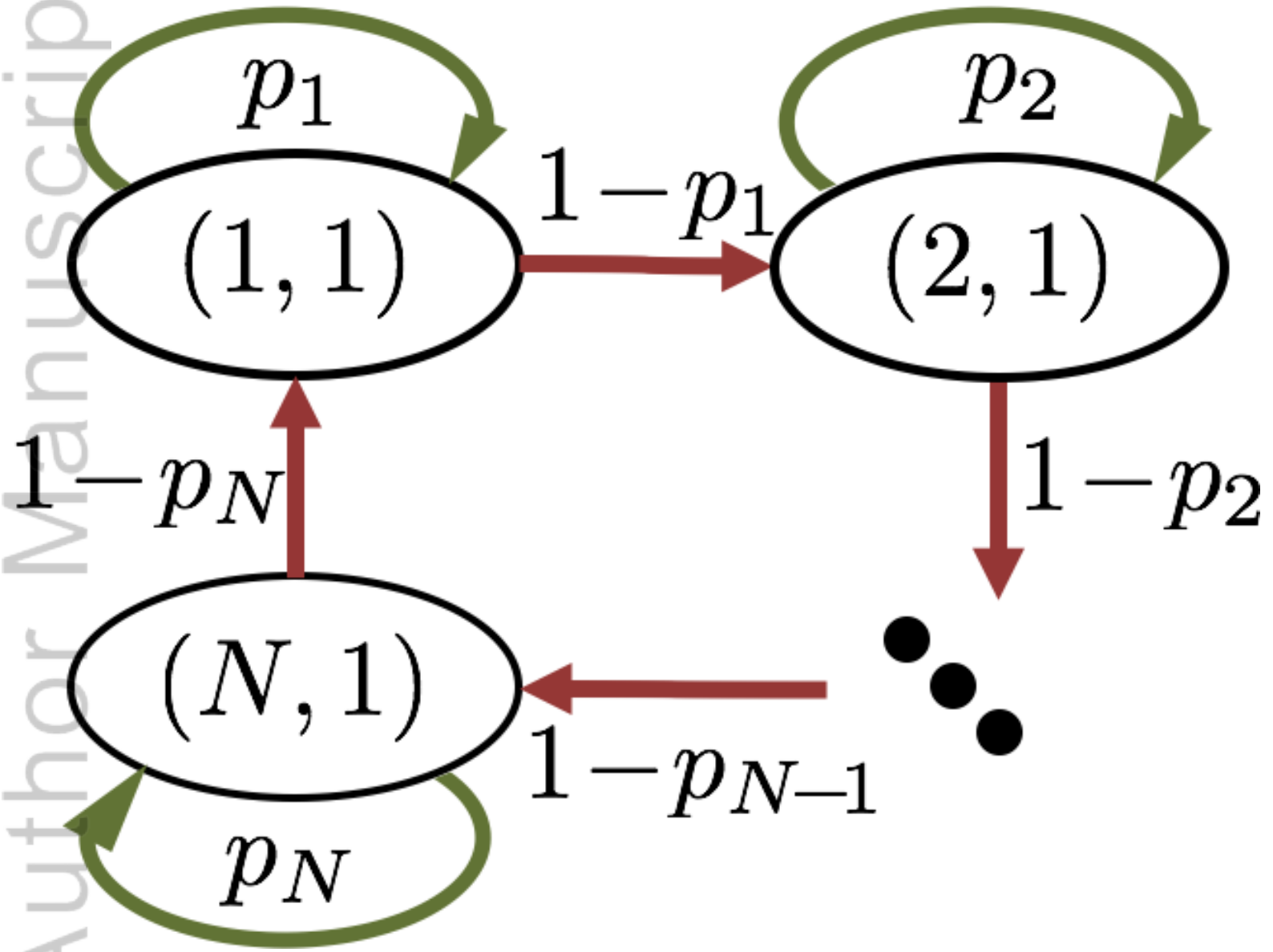
RNC_5292_WIN_Branch.png



RNC_5292_WIN_FULL.png



RNC_5292_WMA_Branch.png



RNC_5292_WMA_FULL.png