Modelling Evolving Voting Behaviour on Internet Platforms
Stochastic Modelling Approaches for Dynamic Voting Systems

Shikhar Raje[1], Navjyoti Singh[1] and Shobhit Mohan[2]
[1]*Center for Exact Humanities, International Institute of Information Technology, Hyderabad, Hyderabad, India*
[2]*Department of Economics, Hyderabad Central University, Hyderabad, India*

Keywords: Preference Aggregation, Stochastic Modelling, Dynamic Voting, Markov Decision Processes, Computational Complexity, Algorithm Analysis.

Abstract: Markov Decision Processes (MDPs) and their variants are standard models in various domains of Artificial Intelligence. However, each model captures a different aspect of real-world phenomena and results in different kinds of computational complexity. Also, MDPs are recently finding use in the scenarios involving aggregation of preferences (such as recommendation systems, e-commerce platforms, etc.). In this paper, we extend one such MDP variant to explore the effect of including observations made by stochastic agents, on the complexity of computing optimal outcomes for voting results. The resulting model captures phenomena of a greater complexity than current models, while being closer to a real world setting. The utility of the theoretical model is discussed by application to the real world setting of crowdsourcing. We address a key question in the crowdsourcing domain, namely, the Exploration Vs. Exploitation problem, and demonstrate the flexibility of adaptation of MDP-based models in Dynamic Voting scenarios.

1 INTRODUCTION

The Internet exhibits a variety of voting and preference aggregation schemes. This is immediately evident from the wide use of such schemes in ranking product features, ranking of songs and artists, etc (Altman and Tennenholtz, 2005). In many of these settings the aggregated ranking is dynamic, i.e. the system announces the ranking at each particular point, and may revise it when new agents arrive to the system and announce their votes, or when existing agents change their votes. Therefore, the Internet calls for additional study of voting and preference aggregation schemes, that goes beyond the classical models. We refer to this setting as *Dynamic Voting* (Tennenholtz, 2004).

A recent treatment of Dynamic Voting is due to (Parkes and Procaccia, 2013). The authors modelled preference aggregation as a discrete-space, discrete-time evolutionary model, which was then modelled as a stochastic process. In order to construct a tractable algorithm for their model, a symmetry-based contraction was used to reduce the state space of the problem. The results generated from their approach differed from traditional results in Social Choice since an axiomatic approach was traded for specifying the behaviour of the system formally. The study focused on the dynamic behaviour of the system as opposed to axiomatic insights.

In this paper, we extend stochastic modelling techniques to scenarios in Dynamic Voting. The rest of the paper is organized as follow. We begin Section 2 by surveying the approach taken in (Parkes and Procaccia, 2013). Section 3 extends the groundwork to newer stochastic models to deal with more complex settings. Specifically, we introduce a model that allows us to account for observations made by voters about their surroundings (including the votes of fellow voters) and interpret the findings from using the model in that setting. In Section 4, a high-level application of this model to existing scenarios presents potential benefits, thereby justifying the use of such a novel approach. We conclude in Section 5.

2 DYNAMIC VOTING AS AN MDP

In a voting or preference aggregation scenario, the uncertainty stems from the possible action that an agent may take from a set of actions, given a particular set of inputs. Current methods in Dynamic Voting focus on relating these actions to the future evolution of the

Raje, S., Singh, N. and Mohan, S.
Modelling Evolving Voting Behaviour on Internet Platforms - Stochastic Modelling Approaches for Dynamic Voting Systems.
DOI: 10.5220/0006073502390244
In *Proceedings of the 8th International Joint Conference on Computational Intelligence (IJCCI 2016) - Volume 1: ECTA*, pages 239-244
ISBN: 978-989-758-201-1

system. This relation is achieved by defining the possible actions that a voter can take as a random variable for constructing a stochastic process. Random variables from multiple voters can then be aggregated into a single random variable that represents the aggregate uncertainty of the entire system of voters.

In (Parkes and Procaccia, 2013), the class of stochastic processes chosen to model Dynamic Voting are *Markov Decision Processes* (Puterman, 2014), (Howard, 1960). A Markov Decision Process (MDP) is a tuple $M =< S,A,R,T >$. S represents the finite state space of the process. A is the finite set of actions that can be taken to change the state of the system from one state to another. $R : S \times A \rightarrow \mathbb{R}$ is a reward function that where, for $s \in S$ and $a \in A$, $R(s,a)$ is a reward obtained by taking action a in state s. $T : S \times (S \times A) \rightarrow \mathbb{R}$ is the transition function, where $T(s'|s,a)$ (or $T(s,a,s')$) is the probability that the system will move to state $s^{`}$ when action a is taken in state s. Notice that the rewards for the voter are defined only in terms of the action taken in the current state, which is known as the *Markov property*. A *deterministic optimal policy* maps set of preferences from multiple voters to a single aggregated preference that satisfies some optimality criteria.

2.1 Formal Model of Dynamic Voting as an MDP

Formally, the approach taken by the authors is to model individual voters as MDPs, then combine the individual models into an aggregate model. This model is then optimized for computation of the optimal policy. The voter MDPs are defined as follows:

- The state space of the voter is the set of preference orderings that the voter can vote for. For m alternatives, S is a set of size $m!$.
- The actions that the voter can take is to change his current ordering over the various preferences. Therefore, A is also a set of size $m!$, since the agent can change to any of the other orderings.
- T and R are provided as input. In a real-world setting, these functions could come from machine learning techniques over past data for the voter.

In the aggregate setting, S becomes a state space of size $(m!)^n$, since every one of the n voters can be in one of the individual $m!$ states. A, while still remaining a set of size $m!$ now represents the system declaring a particular preference ordering as the winner of the lot. Since each voter evolves independently, the aggregate T for a state s and s' is the product of the probabilities of each voter to move from s to s'. The rewards are simply the sum total of rewards for each voter in the state change.

Finally, the authors also note that the system can be designed to comply with certain desirable behaviour. Briefly, they propose modifying the aggregate reward function to heavily penalize certain actions in certain states.

3 PARTIALLY OBSERVABLE MDPs IN DYNAMIC VOTING

A Partially Observable MDP (POMDP) differs from basic MDPs in that, the state of the evolving agent is not completely visible to the agent. The agent, instead of being in a particular state, maintains a *belief vector* or probabilities of being in various states. The agent then receives certain observations on every state change, through which it updates its belief vector on which state it is likely to be in. Formally, a Partially Observable Markov Decision Process (POMDP) (Kaelbling et al., 1998), (Sondik, 1971) is a tuple $< S,A,\Omega,T,O,R >$. Besides the elements already defined as part of an MDP, Ω is a finite set of observations that the voter can receive on a state change, and $O(o|s)$ (or $O(s,o)$) is a probability of making a given observation in a particular state.

3.1 Formal Model of Dynamic Voting as a POMDP

Formally, the voter POMDPs are defined the same way as they are for MDPs, with the only difference coming in the voter state space. In our model, we propose the voter state space as an $(m!)^n$ set, containing all the possible preference orderings of the other agents, as well. Therefore, if the voter does not have perfect information about the preferences reported by all the other voters in the setting, then he is unsure about his present state. The voter therefore maintains his current state as a probability distribution or *belief vector* b over the state space, where $b(s)$ is the agent's belief that it is in state s. This lack of perfect information is a reasonable assumption in most real-world scenarios involving Dynamic Voting.

Additionally, we define the observation set as a subset of the preference orderings reported by the other voters. That is, we say that voter i observes the votes of some subset $L_i = \{1,\cdots,l\}$ of the other agents. Therefore, the observation space becomes a set of size $(m!)^l$ for each agent. We observe that, definitions of the observation set and observation function need not be limited to observations of the behaviour of other voters. It may not even be a single observable phenomena. Multiple classes of observations can be combined into a unified observation function and observation set through Bayesian operations. As we discuss later, this allows for greater adaptability in using this model in real-world settings. Also, as for transition and reward functions, observation functions are also entered as input.

The aggregate POMDP can be obtained from the individual voter POMDPs following the process described in section 2.2. We observe that, in defining the observation set for the aggregate model, since each agent makes an independent observation, the size of the observation set becomes $|\Omega| = \prod_{i \in N} (m!)^{|L_i|} = (m!)^{\sum_{i \in N} |L_i|}$. Finally, at the combination stage, a binary constraint function $C(s,a)$ is also added, which is used to enforce desirable behaviour on the system. $C(s,a)$ returns 1 if an action a is not allowed in a particular state, and 0 otherwise.

3.1.1 Computation of the Optimal Policy

There exists a large body of literature on the challenges and methods for computation of optimal policies for POMDPs. Readers are directed to (Kaelbling et al., 1998), (Sondik, 1971), (Kaelbling et al., 1996) and (Amato et al., 2014) for further reading. For the purpose of our study, however, we choose the algorithm presented in (Undurti and How, 2010). We do this since the algorithm deals with constrained POMDPs, is reasonably simple in complexity and the authors emphasise the tractability of the algorithm through offline, pre-computation methods. An adapted version of the algorithm is presented in the appendix. The algorithm computes the optimal policy by computing future belief states (encapsulated in the τ function, which comes from (Kaelbling et al., 1998)), while using a discount factor γ for deciding the impact of future rewards on current optimal actions. We observe that, in the computation process for the optimal policy, the system anticipates each of the possible observations in the aggregate observation set, and calculates an expected reward in the event of making that observation, making the computational complexity of the algorithm polynomial in $|\Omega|$. However, as defined earlier, this observation set grows exponentially in the number of voters that can be observed by each voter (assuming the number of alternatives remains constant).

3.1.2 Model Optimization

An assumption that we can make in the design of our model, which would improve tractability, is that each individual voter, instead of observing the preferences of a fixed subset L_i of all the other voters, simply observes $|L_i|$ votes. That is, the reported preferences are disassociated from the people reporting the preference. Intuitively, one way to interpret this simplification is the difference between aggregation behaviour on social networks (where we can observe which of our connections “liked” or “followed” an alternative) versus aggregation behaviour on e-retail websites (where some people rated a product 5 stars, some others rated it 4 stars, etc.). This clearly reduces the size of the voter observation sets, and consequently, the aggregate observation set.

The exact size of the observation set in this new setting can be calculated as follows. If we assume an “alphabet” of size $m!$ (where each “letter” is a preference ordering), then we wish to know how many sets of length $|L_i|$ can be formed from this alphabet. An established result in combinatorics allows us to compute this result as $\prod_{i \in N} {}^{m!+|L_i|-1}C_{|L_i|}$. Again, assuming the number of alternatives to be fixed (for example, 3), then the expression evaluates to a polynomial of order 5, or $O(|L_i|^5)$ complexity.

4 CROWDSOURCING AS DYNAMIC VOTING

In this section, we apply the model to a real-world setting and discuss the advantages. We apply this model to *crowdsourcing* (Slivkins and Vaughan, 2014). We begin by defining the problem in crowdsourcing platforms, briefly analysing existing methodologies, and comparing our model to these methodologies.

We identify three interacting components in a crowdsourcing platform, namely, the workers, the requesters and the platform matching these two components to one another. Two observations about crowdsourcing encourage an approach from a Dynamic Voting perspective. Firstly, with each iteration, workers and requesters can form an insight into how the other group is making decisions. The platform can generate profiles to predict the kinds of tasks that workers might choose, or their performance in the completion of tasks, or it might match requesters to a set of workers which give the requesters the most optimal output.

Secondly, we observe the presence of two different strategic elements in the system. In the first *local* element, members of individual components are seeking to maximize their payoff with strategic interactions. These interactions can be between workers and requesters, such as when workers are debating how much effort to put in for the amount offered by the requester (Mason and Watts, 2010), or by requesters who want to incentivize good behaviour through mechanisms such as reputation systems (Dellarocas, 2005). In the second *global* element, we observe that, for long-term sustainability of a crowdsourcing platform, it should have certain desirable characteristics that are guaranteed in it's performance over a horizon of repeated decisions.

These observations make crowdsourcing a good fit for the constrained POMDP Dynamic Voting model presented in the earlier sections.

4.1 Current Perspectives

A general model for the study of crowdsourcing is due to Cesa-Bianchi et al., known in literature as the *multi-armed bandit* model (Auer et al., 1995), (Slivkins and Vaughan, 2014). The problem statement for the model is stated as follows. In a simple scenario, an agent plays against a Vegas-style slot machine (colloquially referred to as a 'one-armed bandit'). In such a setting, the agent does not know the probabilities of payoff with repeated play. The multi-armed scenarios extends this setting to k arms or machines, with each machine having a unique probability distribution for the payoffs assigned.

This model addresses a key challenge in existing crowdsourcing literature known as the *Exploration Vs. Exploitation* problem. This is a challenge wherein, given a set of repeated plays against multi-armed bandits, an agent has to decide whether a particular play should be allotted to the task of exploration or exploitation. An in-depth presentation of the problem is due to Slivkins and Vaughan. (Slivkins and Vaughan, 2014). Emphasis on the Exploration Vs. Exploitation problem has the effect of localizing the study of crowdsourcing to a per-agent basis. That is, "efficiency" and "optimal behaviour" of such models are defined in terms of maximizing payoff for only the task requesters and workers of the system, and not in terms of desiderata which exist beyond those distributions.

While a few studies have been done which factor in desiderata from literature on Social Choice theory (Lee et al., 2014), (Mao et al., 2013), our model differs from these in two ways. Firstly, we focus on integrating Dynamic Voting models into existing crowdsourcing models. This separates our work from the work done by Lee et al. (Lee et al., 2014), where the end product is a new model based on an alternative definition of "exploration vs. exploitation". Secondly, our model aims to be closer to real-life situations by factoring in the aspect of repeated decision-making, which is not covered by the work of Mao et al. (Mao et al., 2013).

4.2 An Initial Model

While the action and state spaces have a direct definition as per section 3, the definition of the observation space provides some more insight. Defining observations and relating them to the progression of the model is a challenge, since the original multi-armed bandit model proposed updations of the static and dynamic elements of the MDP with each observation. Essentially, this resulted in models of the multi-armed bandit problem being constructed as one-state MDPs, with every iteration changing the probabilities of outcomes from playing. The POMDP version fixes this issue by adding all the information of the system that can be known as a static set, and the effects of an observation made at one iteration are manifested in the value function of the next iteration, not in the elements of the POMDP itself.

4.3 Adding Exploration Vs. Exploitation

Adding elements to model the question of Exploration Vs. Exploitation can now be achieved by modifying the action space and reward function of the agent POMDP.

1. We redefine the action space by adding a redundancy. Let one action space $A_{exploration}$ be an $m!$ sized set of the preference orderings when the agent is taking an action from an exploration perspective. Similarly, another action space $A_{exploitation}$ be the $m!$ sized set of all the preference orderings when the agent is exploiting the current system with his current knowledge. The transition function for an agent is also redefined for this new action space. For example, a more exploration-prone agent would have greater transition probabilities for an action in $A_{exploration}$ than for the same action in $A_{exploitation}$.
2. To reflect the trade-off between the two actions, we redefine the reward function. In a trivial case, if, for an action a, the reward function in all states is $R(a_{exploration}, s) = R(a_{exploitation}, s)$, then the agent is playing a *purely random strategy*, where the information from the observations are not playing a role in deciding the next action that the agent should take. However, without loss of generality, we can define the relationships between the reward functions in the two cases as $R(a_{exploration}, s) = R(a_{exploitation}, s) + \Delta_{a,s}$, where the $\Delta_{a,s}$ term defines

the extra value that the agent gains in terms of information about the system, in addition to the reward from taking the action in that particular state.

3. Now, the Exploration Vs. Exploitation problem can be expressed by modifying the Value Function and Optimal Policy. We define this problem in terms of the difference between $V_{ideal}(s)$, the ideal or maximum value that an agent can gain from the current state, if he were playing with full observations and complete knowledge of the system and $V_{actual}(s)$. We now assert that the Exploration Vs. Exploitation problem is to minimize the distance between this value, and the value that the agent gains in it's own iterations i.e.

$$\begin{aligned} V_{actual}(s) \;=\; \min_{a\,\in\,A}[V_{ideal}(s) \;-\; R(s,a) \\ -\,\gamma \sum_{s' \in S} T(s'|s,ca).V_{actual}(s')] \end{aligned} \tag{1}$$

Essentially, this optimality criteria asserts that the best possible outcome occurs when the agent's behaviour converges towards that shown by an identical agent operating under ideal conditions. The value of $V_{ideal}(s)$, and the method for solving of the recurrence relation is explained under the chapter on Discounted MDPs (chapter 6) of (Puterman, 2014).

4.4 Insights

While the usage of a POMDP presents a steep cost in terms of tractability over the single-state, multi-armed bandit model used in literature, we note the benefits that this model presents. Firstly, we observe that the definition of general observation sets and functions provides a distinct modelling advantage for real-world settings over even the most widely-used generalization of the multi-armed bandit model.

The second advantage is that the process followed here to adapt the model can be generalized and used to extend Dynamic Voting POMDPs in ways similar to the various extensions of the multi-armed bandit model. For example, to study regret minimization in a system and reward based reputation systems (Auer et al., 1995).

In conclusion, we propose that models for crowdsourcing based on dynamic voting approaches can supersede current multi-armed bandit based models. The core phenomena being modelled by the latter can be incorporated into the former, while the former provides a more generalized and easily extensible framework for the crowdsourcing scenario.

5 RESULTS

In this paper, we began with the problem of modelling the evolving of preferences of agents on internet platforms. Current approaches focus on evolving preferences over a horizon of repeated decision-making. This was a novel approach, since it studied aggregate decision-making by constructing it as an evolving system, rather than using the standard logic-based approaches that were the mainstay of the domain so far. These approaches focused on a single instance of decision aggregation, and questions asked were along the lines of how best to map the individual inputs to the output, to maximize overall social utility (Moulin et al., 2016).

Therefore, with this work, this paper builds a case for further investigation into the use of MDP variants and stochastic modelling techniques in real world Dynamic Voting scenarios. We show that, stochastic modelling yields greater insight into the relation between factoring a real-world aspect (such as the observation a voter makes about the votes of others), and the resulting change in computational complexity.

For example, *Semi Markov Decision Processes*, a class of MDPs where different state changes are completed in different times, presents a unique challenge at the aggregation step. A workaround from (Gosavi, 2014) reduces SMDPs into regular MDPs for this aggregation step. Therefore, while SMDP based models would be as tractable as the basic MDP model (and much more tractable than the POMDP model), the process for conversion changes the optimal policy for the system. Additionally, we have the novel insight that, factoring in voter behaviour related to speed of changing one's own decisions has no effect on computational complexity, while making observations about other voters results in more complex computation.

Similarly, (Gmytrasiewicz and Doshi, 2004) and (Wiering et al., 2007) introduce *interactive POMDPs* and *Multi-objective MDPs*. IPOMDPs extend POMDPs by allowing for interactions between agents, and MOMDPs use multiple optimality criteria, instead of the straightforward criteria used here, which maximizes only the payoff of the entire system.

Finally, another key line of research would use these theoretical models on actual platforms, with real data. Of particular interest would be a study on how different methods for approximating policies would translate in a Dynamic Voting scenario, in terms of the tradeoff between optimal rewards and computational tractability.

REFERENCES

Altman, A. and Tennenholtz, M. (2005). Ranking systems: the pagerank axioms. In *Proceedings of the 6th ACM conference on Electronic commerce*, pages 1–8. ACM.

Amato, C., Konidaris, G. D., and Kaelbling, L. P. (2014). Planning with macro-actions in decentralized pomdps. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1273–1280. International Foundation for Autonomous Agents and Multiagent Systems.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (1995). Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 322–331. IEEE.

Dellarocas, C. (2005). Reputation mechanism design in online trading environments with pure moral hazard. *Information Systems Research*, 16(2):209–230.

Gmytrasiewicz, P. J. and Doshi, P. (2004). Interactive pomdps: Properties and preliminary results. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1374–1375. IEEE Computer Society.

Gosavi, A. (2014). *Simulation-based optimization: parametric optimization techniques and reinforcement learning*, volume 55. Springer.

Howard, R. (1960). *Dynamic programming and Markov processes*. MIT Press.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134.

Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, pages 237–285.

Lee, D. T., Goel, A., Aitamurto, T., and Landemore, H. (2014). Crowdsourcing for participatory democracies: Efficient elicitation of social choice functions. In *Second AAAI Conference on Human Computation and Crowdsourcing*.

Mao, A., Procaccia, A. D., and Chen, Y. (2013). Better human computation through principled voting. In *AAAI*. Citeseer.

Mason, W. and Watts, D. J. (2010). Financial incentives and the performance of crowds. *ACM SigKDD Explorations Newsletter*, 11(2):100–108.

Moulin, H., Brandt, F., Conitzer, V., Endriss, U., Lang, J., and Procaccia, A. D. (2016). *Handbook of Computational Social Choice*. Cambridge University Press.

Parkes, D. C. and Procaccia, A. D. (2013). Dynamic social choice with evolving preferences. In *AAAI*.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Slivkins, A. and Vaughan, J. W. (2014). Online decision making in crowdsourcing markets: Theoretical challenges. *ACM SIGecom Exchanges*, 12(2):4–23.

Sondik, E. J. (1971). The optimal control of partially observable markov processes. Technical report, Ph.D Thesis, Stanford University.

Tennenholtz, M. (2004). Dynamic voting. In *EC'04: Proceedings of the 5th ACM Conference on Electronic Commerce, New York, New York, USA, May 17-20, 2004*, page 230. Association for Computing Machinery.

Undurti, A. and How, J. P. (2010). An online algorithm for constrained pomdps. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3966–3973. IEEE.

Wiering, M., De Jong, E. D., et al. (2007). Computing optimal stationary policies for multi-objective markov decision processes. In *Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on*, pages 158–165. IEEE.

APPENDIX

```
Procedure OnlinePOMDPsolver
  $b$:current belief state of the system
  $T$: A tree-like data structure to
      contain the current state of the
      system and possible future
      transitions
  $D$:Expansion depth for lookahead
  $b \leftarrow b_0$
  $T \leftarrow b$
  WHILE ExecutionTerminationCondition
      DO
    Expand($b,D$)
    Execute action $a^*$ returned by
        Expand
    Receive observation $o$
    Update $b$ to reflect new belief
        state of the system
    Update tree $T$
  END WHILE

Procedure Expand(b, d)
  IF $\{d = 0\}$
    $V(s) \leftarrow 0$
  ELSE
    $V(s) \leftarrow -\infty$
  END IF
  FOR $a \in |A|$ AND $C(s,a) = 0$ DO
    $P(o|b,a) \leftarrow \sum_{s' \in S} O(o|s') \sum_{s \in S} T(s'|s,a) b(s)$
    $V(s,a) \leftarrow \sum_{n \in N} [R(s,a) + \gamma \sum_{o \in \Omega} P(o|b,a)$Expand(
        $\tau(b_n,a,o), d-1$)]
    IF $\{V(s,a) > V(s)$ AND $C(s',a) = 0\}$
      $V(s) = V(s,a)$
      $a^* \leftarrow a$
    END IF
  END FOR
  RETURN $a^*$, $V(s)$
```