Formalization and Analysis of Automated Dispute Resolution via Tokenholder Tribunal on the Blocklancer Platform

Michael Kaiser and Kevin Kaiser

contact@blocklancer.net,
WWW home page: https://www.blocklancer.net

Abstract. Blocklancer is a decentralized and autonomous freelancing platform on the Ethereum Blockchain [7]. In this yellow-paper a mathematical model of the dispute resolution process is presented. At first the concepts of the Blocklancer platform are described and formalized. Then the dispute resolution process and the Tokenholder Tribunal are formalized. The Tokenholder Tribunal is analyzed by means of decision theory, game theory and probability theory. Finally it is shown that the Tokenholder Tribunal is designed in a way that making competent decisions is most profitable for Tokenholders.

Keywords: Cryptocurrency, Game Theory, Decision Theory, Probability Theory, Ethereum, Blockchain

1 Introduction

Blocklancers Tokenholder Tribunal is built on Executable Distributed Code Contracts (EDCC) offered by the Ethereum Network. The Tokenholder Tribunal is a crypto-economic framework to allow automated dispute resolution by the Tokenholders. Tokenholders are allowed to vote on disputes and get incentivized for participating in the dispute resolution process between clients and freelancers. Building such voting mechanics is not easy, because it has to be guaranteed that the votes of the participants converge to the correct solution. If there is a flaw in the voting protocol, it could lead to a situation where participants get rewarded for guessing and the solution is not useable at all. The Tokenholder Tribunal of Blocklancer is such a voting mechanism and therefore the protocol for voting has to be designed very careful. In the following sections it will be shown that the Tokenholder Tribunal converges to a correct solution. Before the analysis, a formal model that fits the actual process has to be developed. In the following, the components of the Tokenholder Tribunal will be defined and formalized and finally the Tokenholder Tribunal is modeled as a Partially Observable Markov Decision Process (POMDP). This formal model can then be further analyzed. The analysis shows that the optimal strategy for the agents, participating in the Tokenholder Tribunal, is to make competent decisions. This shows that Blocklancer incentivizes Tokenholders that are actively participating in the Tokenholder Tribunal and making competent decisions the most. In the following section 2, some formal definitions will be given.

2 Preliminaries

Definition 1. *(The Lancer Token)*
Λ is the amount of all LNC Tokens. LNC is the internal Token of the Blocklancer platform.

Definition 2. *(Tokenholders)*
T is the set of all Tokenholders $\tau_{i \in N}$ with $|T| = N$. Every Tokenholder $\tau_{i \in N}$ holds a certain amount of Tokens. $v(\tau_{i \in N}, t)$ denotes the amount of Tokens $\tau_{i \in N}$ holds at timestamp t.

Definition 3. *(Dispute Cycle)*
A dispute cycle o has a starting time $t_s(o)$ and ending time $t_e(o)$. After the dispute cycle the collected fees $\xi(o)$ are distributed between the Tokenholders T_o, who actively voted in dispute cycle o. $|T_o|$ denotes the amount of active Tokenholders in dispute cycle o.

Definition 4. *(Dispute)*
A dispute δ can be initiated by a Freelancer ϕ or a Client κ. The dispute δ has a starting time $t_s(\delta)$ and ending time $t_e(\delta)$ and belongs to a dispute cycle o. Δ_o is the set of all disputes belonging to dispute cycle o. $\delta \in \Delta_o$ if $t_e(\delta) > t_s(o)$ and $t_e(\delta) \leq t_e(o)$.

Definition 5. *(Dispute Resolution Groups)*
Tokenholders are partitioned to form Dispute Resolution Groups (DRG). Algorithm 1 depicts the procedure for generating n Dispute Resolution Groups. $m(g)$ returns all the members of group g. G denotes the set of all $g_{i \in |G|}$. Moreover it applies that $\bigcup_{i=1}^{|G|} g_i = T$. G is a partition of T and it applies that $x \sim_G y :\Rightarrow \exists g \in G : x \in g, y \in g$.

3 Dispute Instantiation

The goal of this section is to formalize the process of disseminating the disputes among the Tokenholders. The dissemination of disputes should not be

Algorithm 1 Generate Dispute Resolution Groups

procedure GENERATE_DISPUTE_RESOLUTION_GROUPS(n) ▷ n is number of groups to generate
  $group\_size \leftarrow \frac{|T|}{n}$
  $tokenholder\_stack \leftarrow T$
  $G \leftarrow \{g, \ldots, g_n\}$ ▷ Set of groups
  while $|tokenholder\_stack| \neq 0$ **do** ▷ While there are Tokenholder
    $g_{now} \leftarrow uniform\_rand(G)$ ▷ Get a random group
    while $|g_{now}| = group\_size$ **do** ▷ While group is full
      $g_{now} \leftarrow uniform\_rand(G)$
    $g_{now} \leftarrow g_{now} \cup tokenholder\_stack.pop()$
  return G ▷ The Dispute Resolution Groups

predictable by Tokenholders, Clients and Freelancers. The Client κ and the Freelancer ϕ both have the opportunity to instantiate a dispute. If a dispute δ is instantiated, δ and all the information attached to δ is messaged to a Dispute Resolution Group g with $g \sim U(G)$. $g \sim U(G)$ denotes that g is uniformly drawn from G. The Tokenholders $g_{\Upsilon} = \tau \in (m(g) - (\kappa \cup \phi))$ get notified and a new Tokenholder Tribunal Υ forms. If $\kappa \in m(g)$, κ is not allowed to vote in Υ, also if $\phi \in m(g)$ then ϕ is not allowed to vote in Υ.

Because $g \sim U(G)$ and g is constructed randomly (see Alg. 1), κ and ϕ cannot know, who is eligible to vote in Υ.

Definition 6. *(Collusion)*
An actor x is colluding with actor y, if y is always voting for x if $x = \kappa$ or $x = \phi$ for any Υ. Collusion is symmetric and denoted by $x \leftrightsquigarrow y$.

This means an actor x has to collude at least with $|G|$ actors to get a vote through collusion in every possible g. It has to be true that $\forall g \in G\ \exists y \in m(g) \wedge x \leftrightsquigarrow y$. The probability of collusion diminishes with an increase in $|G|$.

4 The Tokenholder Tribunal

The goal of this section is to show that competent voting is incentivized the most and that guessing is not profitable for the Tokenholders. At first, the Tokenholder Tribunal will be formalized. A game theoretical definition of the majority rule will be given. Then the decision process will be modeled as a Markov Decision Process (MPD) [4]. The MDP will be analyzed and an ϵ-optimal policy will be computed and presented. Furthermore, the MDP will be extended to a POMDP [3] [2] [5] and finally the optimal policy of the POMDP will be analyzed.

Definition 7. *The Tokenholder Tribunal is a 7-tuple $\Upsilon = \langle g, \delta, P, V, F, K, c\rangle$ with*

$$
\begin{aligned}
g &\coloneqq \textit{dispute group resolving dispute } \delta,\\
\delta &\coloneqq \textit{dispute that has to be resolved},\\
P &\coloneqq \textit{a set } \{\kappa, \phi\} \textit{ containing the dispute parties } \kappa \textit{ and } \phi,\\
V &\coloneqq \textit{a function } g \times P \to \{0,1\} \textit{ with } v(\tau, p) = \begin{cases} 1 & \textit{if } \tau \textit{ voted for } p \\ 0 & \textit{otherwise} \end{cases},\\
F &\coloneqq \textit{a set } \{\tau | \tau \in g \wedge v(\tau, \phi) = 1\} \textit{ containing the Tokenholders, who voted for } \phi,\\
K &\coloneqq \textit{a set } \{\tau | \tau \in g \wedge v(\tau, \kappa) = 1\} \textit{ containing the Tokenholders, who voted for } \kappa,\\
c &\coloneqq \textit{a function } \{\phi, \kappa\} \to \mathbb{R}_{\geq 0} \textit{ being the number of votes for } \kappa \textit{ or } \phi.
\end{aligned}
$$

The Tokenholders from the Tokenholder Tribunal Υ are taking part in a cooperative game [6].

Definition 8. *(Cooperative Game)*
A cooperative game is a 2-tuple $\langle N, v \rangle$ *with*

$$
\begin{aligned}
N &\coloneqq \textit{the set of players, called the grand coalition},\\
v &\coloneqq \textit{the characteristic function with } 2^N \to \mathbb{R}.
\end{aligned}
$$

More specific the cooperative game from the Tokenholder Tribunal is a simple [6] majority rule game.

Definition 9. *(Simple Game)*
A game $\langle N, v \rangle$ *is simple, if for every coalition* $S \subset N$*, either* $v(S) = 1$ *or* $v(S) = 0$*.*

Definition 10. *(Majority Rule Game)*
A majority rule game is a simple game where

$$
v(S) = \begin{cases} 1 & \textit{if } |S| > \frac{n}{2} \\ 0 & \textit{otherwise} \end{cases} \tag{1}
$$

.

Definition 11. *(Positive and Negative Votes)*
A vote is denoted as positive, if the Tokenholder, who voted is part of the winning coalition. If the Tokenholder, who gave the vote, is not part of the winning coalition, the vote is denoted as negative.

In the Tokenholder Tribunal the probability for a vote to be a positive vote should be higher, if the Tokenholder makes a competent decision. If $v(S) = 1$, S is called the winning coalition. The majority game is not weighted, which means that each Tokenholder has exactly one vote. The amount of LNC a Tokenholder holds is not affecting the weight of the vote. Otherwise there would be the risk that Tokenholders could dominate dispute groups, because of the natural Pareto-distribution of Tokens. Figure 1 shows the distribution of the Augur REP Tokens. It can be seen that the Tokens are Pareto-distributed. This distribution is natural for most Tokens and therefore this distribution has to be considered in the design of the dispute resolution procedure. Tokenholders are taking part in a sequence $\langle\langle N, v\rangle_1 \dots \langle N, v\rangle_i \dots \langle N, v\rangle_T\rangle$ of games with $i \in \mathbb{N}$ being the step of majority rule games in each dispute cycle. This sequence of games can be modeled as finite-state MDP.

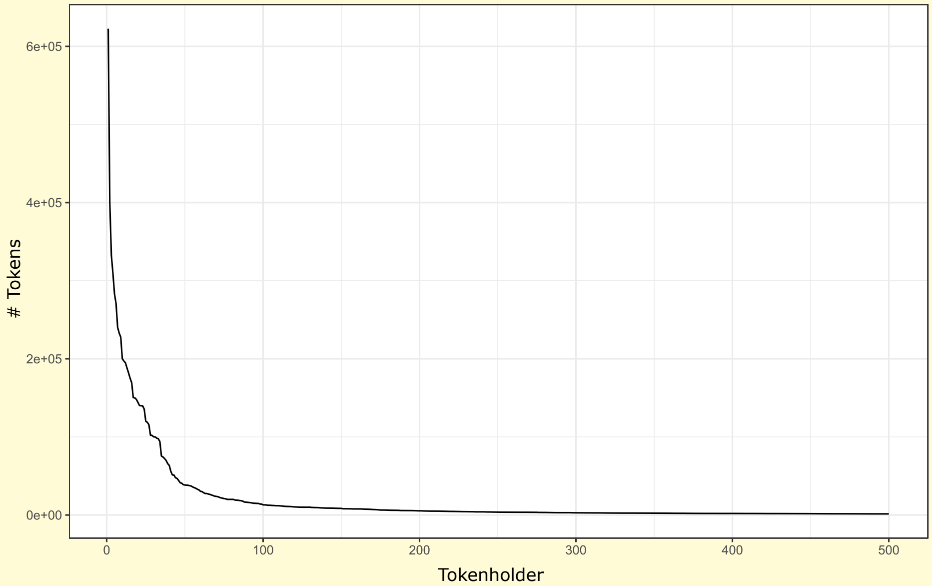

Fig. 1: Pareto-distribution of Augur REP

Definition 12. *(Finite-State Markov Decision Process)*
Formally a finite-state MDP is a 5-tuple $\langle S, A, P, R, \lambda\rangle$ with

$S \coloneqq$ *finite set of states,*

$A \coloneqq$ *set of actions,*

$P \coloneqq$ *the characteristic function with* $S \times S \times A \to [0, 1]$*, that* $P(s, a, s') = \mathbb{P}(s'|s, a)$ *is the probability that* a *leads to* s' *after staying in state* s*,*

$R \coloneqq S \times S \times A \to \mathbb{R}$ *the immediate reward function,*

$\lambda \coloneqq \lambda \in [0, 1]$ *the discount factor.*

The MDP for the Tokenholder Tribunal is denoted as MDP_{Υ} and is defined in Definition 13.

Definition 13. *(MDP_{Υ})*
Formally MDP_{Υ} is a 5-tuple $\langle S, A, P, R, \lambda \rangle$ with

$$
\begin{aligned}
&S \coloneqq \{s, q_{winning\_coalition}, q_{losing\_coalition}, q_{tie}\},\\
&A \coloneqq \{decide, random, inactive\},\\
&P \coloneqq \textit{describes the probability of belonging to the majority or not},\\
&R \coloneqq \textit{describes the earned vote share or penalty},\\
&\lambda \coloneqq \lambda \in [0,1] \textit{ the discount factor.}
\end{aligned}
$$

State	**Description**
s	The starting state
$q_{winning\_coalition}$	The state of belonging to the winning coalition
$q_{losing\_coalition}$	The state of belonging to the losing coalition
q_{tie}	The state when reaching a tie

Table 1: States of MDP_{Υ}

Action	**Description**
decide	The Tokenholder is using the information of the dispute to make an informed decision
random	The Tokenholder votes random
inactive	The Tokenholder is not voting

Table 2: Actions of MDP_{Υ}

A description of the states and actions can be found in Tables 1 and 2. Figure 2 depicts the structure of MDP_{Υ}.

Next, the probability of a tie is computed. The probability of tie can be calculated by a Binomial-distribution. The probability mass function of a Binomial-distribution is

$$\mathbb{P}(k; n, p) = \mathbb{P}(X = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{2}$$

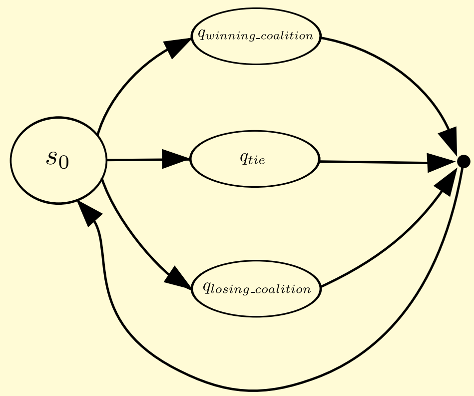

Fig. 2: Structure of MDP_{Υ}

with $n \in \mathbb{N}$ being the number of trials, $k \in \mathbb{N}$ being the number of successes and $p \in [0, 1]$ being the probability of a success.

In the Tokenholder Tribunal, p is the probability of belonging to the winning coalition. For simplicity, assume that belonging to the winning coalition is completely random therefore $p = 0.5$[1]. n is the number of Tokenholders in the grand coalition N, therefore $n = |N|$. To exemplify, set $|N| = 200$ and therefore $n = |N| = 200$. So the probability of a tie with $n = 200, p = 0.5$ is

$$\mathbb{P}(X = \frac{200}{2}) = \binom{200}{100} 0.5^{100}(1 - 0.5)^{200-100} = 0.05634848. \tag{3}$$

In the following, the probability of a tie is specifically denoted as $\mathbb{P}_{tie}(k; n, p)$.

For further analysis it is assumed that the probability of belonging to the winning coalition becomes higher, when making an informed decision rather than a random decision. Therefore, the probability of belonging to the winning coalition with action $a =$ decide is

$$\mathbb{P}(q_{winning\_coalition}|s, decide) = 0.6 - \frac{\mathbb{P}_{tie}(100; 200, 0.5)}{2} \tag{4}$$

and the probability of belonging to the losing coalition with action $a =$ decide is

$$\mathbb{P}(q_{losing\_coalition}|s, decide) = 1 - 0.6 - \frac{\mathbb{P}_{tie}(100; 200, 0.5)}{2}. \tag{5}$$

If the Tokenholder makes a random dicision the probability of belonging to the winning coalition and losing coalition is

$$\mathbb{P}(q_{winning\_coalition}|s, decide) = 0.5 - \frac{\mathbb{P}_{tie}(100; 200, 0.5)}{2} = \mathbb{P}(q_{losing\_coalition}|S, random). \tag{6}$$

[1] It can be assumed that in reality it is likely that $p > 0.5$.

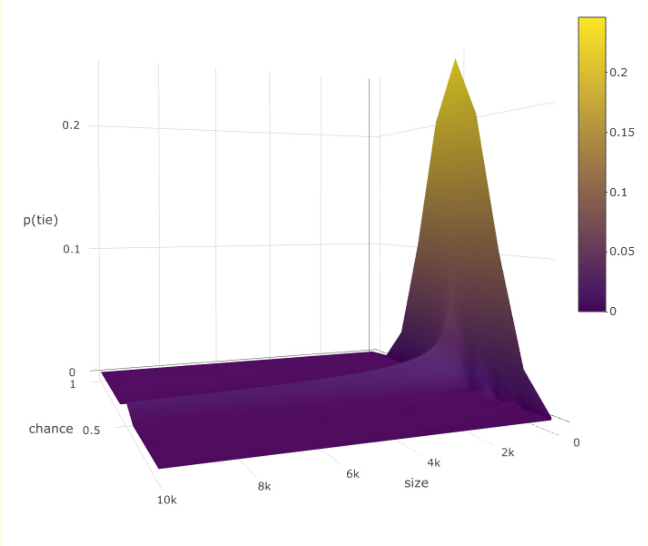

Fig. 3: The probability of tie depends on the number of voters n and the probability of belonging to the winning coalition p.

If the Tokenholder belongs to the winning coalition, the Tokenholder is rewarded with *vote shares*. The more vote shares a Tokenholder earns, the higher will be the Tokenholders share of the fees in the end of the dispute cycle. On the other side, a Tokenholder, who belongs to the losing coalition is penalized. The reward model therefore consists of

1. the reward function that decides, how many vote shares the Tokenholder will get,
2. the penalty function that decides, how the Tokenholder is penalized and
3. the cost funcion that takes the costs for making an informed decision into account.

The reward function depends on the size of the dispute group g and the size of the grand coalition N. It is defined as

$$r(g, N) = \frac{\max(|g| - |N|, 1)^3}{100|g|^3}. \tag{7}$$

The penalty function is

$$p(g, N) = 2r(g, N), \tag{8}$$

and the cost function is

$$c(g, N) = \frac{r(g, N)}{10}. \tag{9}$$

It can be seen that Equation 7 is the difference of $|N|$ and $|g|$ normalized by $100|g|^3$ so that the range of r is $]0, 100]$. Figure 4 depicts that the smaller the grand coalition $|N|$ is in relation to $|g|$ the higher is the reward. This guarantees that voting on disputes with little participation is incentivized. The penalty is the doubled reward. The penalty makes it unattractive for Tokenholders to vote randomly, because there is a risk of losing vote shares. $vs(\tau, \delta)$ denotes the vote shares τ earned for δ. The cost function is the reward function divided by 10 and represents costs[2] involved in making an informed decision. Table 3 shows the MDP_Υ. Table 3 shows all possible transition. All other transitions have a probability of 0.

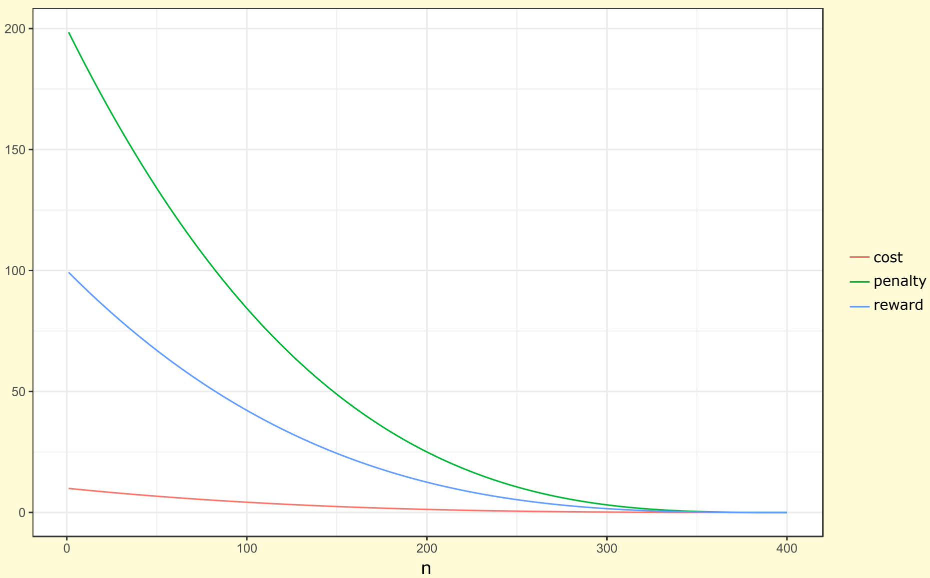

Fig. 4: Reward function, cost function, penalty function with $|g| = 400$

$$vs(\tau, \delta) = \begin{cases} r(g, N) & \text{if } \tau \in S \wedge v(S) = 1 \\ p(g, N) & \text{if } \tau \in S \wedge v(S) = 0 \end{cases} \tag{10}$$

Table 3 depicts that P and R of MDP_Υ depend on N and g therefore a specific MDP_Υ with defined N and g is referred to as $MDP_\Upsilon^{(N,g)}$. N has a significant impact on P and R, therefore it is important to create a decision model with varying N at each step i in the decision process.

[2] Reading all the information, reading code, looking for additional material etc.

Action × State	State'	Probability	Reward
S, decide	$q_{winning\_coalition}$	$\mathbb{P}(q_{winning\_coalition}\|S, decide)$	$r(g,N) - c(g,N)$
	$q_{losing\_coalition}$	$\mathbb{P}(q_{losing\_coalition}\|S, decide)$	$-c(g,N) - p(g,N)$
	q_{tie}	$\mathbb{P}_{tie}(\frac{\|N\|}{2}; \|N\|, 0.5)$	$-c(g,N)$
S, random	$q_{winning\_coalition}$	$\mathbb{P}(q_{winning\_coalition}\|S, random)$	$r(g,N)$
	$q_{losing\_coalition}$	$\mathbb{P}(q_{losing\_coalition}\|S, random)$	$-p(g,N)$
	q_{tie}	$\mathbb{P}_{tie}(\frac{\|N\|}{2}; \|N\|, 0.5)$	0
S, inactive	S	1	0
$\{q_{winning\_coalition}, q_{losing\_coalition}, q_{tie}\}$	S	1	0

Table 3: Tabular form of MDP_{Υ}

This can be achieved by creating a MDP that contains different sub-processes with different configurations of N and g. Which sub-process to activate is controlled by a *Nature State* as new starting state. This new MDP is referred to as $MDP^{\star}_{\Upsilon}$ and is able to model varying Ns at each step. The Nature State is denoted as S_0. S_0 has transitions to starting states $S_{(N_i,g)}$ of multiple $MDP^{(N_i,g)}_{\Upsilon}$s with different configurations for N_i. So the state set of $MDP^{\star}_{\Upsilon}$ is $S_{MDP^{\star}_{\Upsilon}} = S_0 \cup \{S_{(N_1,g)}, \ldots, S_{(N_{|g|},g)}\} \cup \{q_{winning\_coalition}, q_{losing\_coalition}, q_{tie}\}$. Figure 5 depicts the structure of $MDP^{\star}_{\Upsilon}$.

For the following analysis it is assumed that $|g| = 200$ and that N is a discrete random variable and has a Poisson distribution with $\lambda = \frac{|g|}{2}$. The probability mass function of N is given by

$$f(k;\lambda) = \mathcal{P}(N = k) = \frac{\lambda^k e^{-\lambda}}{k!}. \tag{11}$$

Now the probability for each $N_i \in \{1, \ldots, |g|\}$ can be calculated by $f(N_i, \lambda)$ and the transition probability from S_0 to $S_{(N_i,g)}$ is

$$\mathcal{P}(S_{(N_i,g)}|S_0, \cdot) = f(N_i, \lambda). \tag{12}$$

Based on $MDP^{(N_i,g)}_{\Upsilon}$ an ϵ-optimal policy $\pi^{\star}$ can be calculated. For calculating $\pi^{\star}$ the Policy-Iteration Algorithm was used. Table 4 depicts $\pi^{\star}$.

Table 4 shows that whether the Tokenholder chooses the action *decide* or *inactive* sometimes depends on the condition if N is even or odd. For example, the Tokenholder is inactive for $N = 2$, but decides for $N = 3$. The reason for this behaviour is that with $N\%2 = 0$ there is the risk of a tie. In the following, it will be shown that this behaviour only appears, if it is assumed that every Tokenholder exactly knows the value of N. In reality N is unknown to the Tokenholders, therefore the model has to be adapted. During the dispute resolution

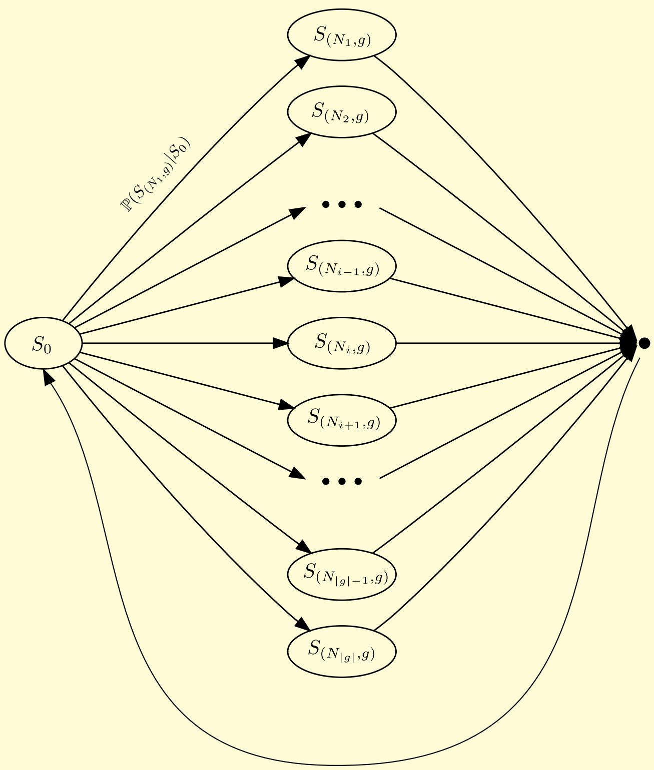

Fig. 5: Structure of $MDP^{\star}_{\Upsilon}$

State	$S_{(1,g)}$	$S_{(2,g)}$	$S_{(3,g)}$	$S_{(4,g)}$	$S_{(5,g)}$	$S_{(6,g)}$	$S_{(7,g)}$	$S_{(8,g)}$	$S_{(9,g)}$	$S_{(10,g)}$	$S_{(11,g)}$
action	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide
	$S_{(12,g)}$	$S_{(13,g)}$	$S_{(14,g)}$	$S_{(15,g)}$	$S_{(16,g)}$	$S_{(17,g)}$	$S_{(18,g)}$	$S_{(19,g)}$	$S_{(20,g)}$	$S_{(21,g)}$	$S_{(22,g)}$
	inactive	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(23,g)}$	$S_{(24,g)}$	$S_{(25,g)}$	$S_{(26,g)}$	$S_{(27,g)}$	$S_{(28,g)}$	$S_{(29,g)}$	$S_{(30,g)}$	$S_{(31,g)}$	$S_{(32,g)}$	$S_{(33,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(34,g)}$	$S_{(35,g)}$	$S_{(36,g)}$	$S_{(37,g)}$	$S_{(38,g)}$	$S_{(39,g)}$	$S_{(40,g)}$	$S_{(41,g)}$	$S_{(42,g)}$	$S_{(43,g)}$	$S_{(44,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(45,g)}$	$S_{(46,g)}$	$S_{(47,g)}$	$S_{(48,g)}$	$S_{(49,g)}$	$S_{(50,g)}$	$S_{(51,g)}$	$S_{(52,g)}$	$S_{(53,g)}$	$S_{(54,g)}$	$S_{(55,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(56,g)}$	$S_{(57,g)}$	$S_{(58,g)}$	$S_{(59,g)}$	$S_{(60,g)}$	$S_{(61,g)}$	$S_{(62,g)}$	$S_{(63,g)}$	$S_{(64,g)}$	$S_{(65,g)}$	$S_{(66,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(67,g)}$	$S_{(68,g)}$	$S_{(69,g)}$	$S_{(70,g)}$	$S_{(71,g)}$	$S_{(72,g)}$	$S_{(73,g)}$	$S_{(74,g)}$	$S_{(75,g)}$	$S_{(76,g)}$	$S_{(77,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(78,g)}$	$S_{(79,g)}$	$S_{(80,g)}$	$S_{(81,g)}$	$S_{(82,g)}$	$S_{(83,g)}$	$S_{(84,g)}$	$S_{(85,g)}$	$S_{(86,g)}$	$S_{(87,g)}$	$S_{(88,g)}$
	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide	decide
	$S_{(89,g)}$	$S_{(90,g)}$	$S_{(91,g)}$	$S_{(92,g)}$	$S_{(93,g)}$	$S_{(94,g)}$	$S_{(95,g)}$	$S_{(96,g)}$	$S_{(97,g)}$	$S_{(98,g)}$	$S_{(99,g)}$
	decide	decide	decide	decide	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide
	$S_{(100,g)}$	$S_{(101,g)}$	$S_{(102,g)}$	$S_{(103,g)}$	$S_{(104,g)}$	$S_{(105,g)}$	$S_{(106,g)}$	$S_{(107,g)}$	$S_{(108,g)}$	$S_{(109,g)}$	$S_{(110,g)}$
	inactive	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**
	$S_{(111,g)}$	$S_{(112,g)}$	$S_{(113,g)}$	$S_{(114,g)}$	$S_{(115,g)}$	$S_{(116,g)}$	$S_{(117,g)}$	$S_{(118,g)}$	$S_{(119,g)}$	$S_{(120,g)}$	$S_{(121,g)}$
	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide
	$S_{(122,g)}$	$S_{(123,g)}$	$S_{(124,g)}$	$S_{(125,g)}$	$S_{(126,g)}$	$S_{(127,g)}$	$S_{(128,g)}$	$S_{(129,g)}$	$S_{(130,g)}$	$S_{(131,g)}$	$S_{(132,g)}$
	inactive	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**
	$S_{(133,g)}$	$S_{(134,g)}$	$S_{(135,g)}$	$S_{(136,g)}$	$S_{(137,g)}$	$S_{(138,g)}$	$S_{(139,g)}$	$S_{(140,g)}$	$S_{(141,g)}$	$S_{(142,g)}$	$S_{(143,g)}$
	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide	**inactive**	decide
	$S_{(144,g)}$	$S_{(145,g)}$	$S_{(146,g)}$	$S_{(147,g)}$	$S_{(148,g)}$	$S_{(149,g)}$	$S_{(150,g)}$	$S_{(151,g)}$	$S_{(152,g)}$	$S_{(153,g)}$	$S_{(154,g)}$
	inactive	decide	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**
	$S_{(155,g)}$	$S_{(156,g)}$	$S_{(157,g)}$	$S_{(158,g)}$	$S_{(159,g)}$	$S_{(160,g)}$	$S_{(161,g)}$	$S_{(162,g)}$	$S_{(163,g)}$	$S_{(164,g)}$	$S_{(165,g)}$
	inactive	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**
	$S_{(166,g)}$	$S_{(167,g)}$	$S_{(168,g)}$	$S_{(169,g)}$	$S_{(170,g)}$	$S_{(171,g)}$	$S_{(172,g)}$	$S_{(173,g)}$	$S_{(174,g)}$	$S_{(175,g)}$	$S_{(176,g)}$
	inactive	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**
	$S_{(177,g)}$	$S_{(178,g)}$	$S_{(179,g)}$	$S_{(180,g)}$	$S_{(181,g)}$	$S_{(182,g)}$	$S_{(183,g)}$	$S_{(184,g)}$	$S_{(185,g)}$	$S_{(186,g)}$	$S_{(187,g)}$
	inactive	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**
	$S_{(188,g)}$	$S_{(189,g)}$	$S_{(190,g)}$	$S_{(191,g)}$	$S_{(192,g)}$	$S_{(193,g)}$	$S_{(194,g)}$	$S_{(195,g)}$	$S_{(196,g)}$	$S_{(197,g)}$	$S_{(198,g)}$
	inactive	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**	**inactive**
	$S_{(199,g)}$	$S_{(200,g)}$									
	inactive	**inactive**									

Table 4: Optimal policy for MDP_{Υ}

procedure, the Tokenholders never know the real value of N, but they can observe the difficulty of the dispute. Because Tokenholders have all the information about the job and the dispute, Tokenholders have an expectation whether the dispute is hard to solve or easy. If the dispute seems hard to solve, Tokenholders may expect that not many Tokenholders have the expertise to vote in the Tokenholder Tribunal and therefore N would be low. On the other hand, if the dispute seems easy Tokenholders may expect that many Tokenholders have the knowledge to vote and therefore N is high. This behaviour of the Tokenholders can be modeled by a Partially Observable Markov Decision Process (POMDP). A POMDP is a MDP, where the current state is not known by the agent. The agent just makes observations and has a belief of the current state.

Definition 14. *(Partially Observable Markov Decision Process)*
Formally a POMDP is a 7-tuple $\langle S, A, P, R, \lambda, \Omega, O\rangle$ *with*

$S :=$ *finite set of states*,

$A :=$ *set of actions*,

$P :=$ *the characteristic function with* $S \times S \times A \to [0,1]$*, that* $P(s,a,s') = \mathbb{P}(s'|s,a)$
is the probability that a *leads to* s' *after staying in state* s,

$R := S \times S \times A \to \mathbb{R}$ *the immediate reward function*,

$\lambda := \lambda \in [0,1]$ *the discount factor*,

$\Omega :=$ *a set of observations*,

$O :=$ *a set of conditional observation probabilities with* $O(o|s',a)$
being the probability of observing o *after entering state* s' *with action* a.

The POMDP for the Tokenholder Tribunal Υ is denoted by $POMDP^*_\Upsilon$ and S, A, P, R and λ are the same like for MDP^*_Υ. This means that the model just has to be extended by observations Ω and observation probabilities O. Ω is $\Omega \in \{\text{easy}, \text{hard}\}$, which means that the Tokenholders can observe easy disputes or hard disputes. The probability O directly depends on N. For calculating O a measurement for the difficulty of disputes is needed.

Definition 15. *A dispute is more difficult, the less Tokenholders have the expertise to vote in the dispute, therefore the difficulty diff directly depends on* N*. The difficulty can be calculated based on a state* $S_{(N_i,g)}$ *by* $\mathit{diff}(S_{(N_i,g)}) = \frac{|g|}{N_i}$*. This means, the smaller* N_i*, the more difficult is the dispute. The vector* D *contains all difficulty values for the states* $\{S_{(1,g)}, \ldots, S_{(|g|,g)}\}$ *and* $D_i = \mathit{diff}(S_{(N_i,|g|)})$*.*

To compute probabilities based on D, the *Softmax function* is used. $\text{softmax}(D)$ contains probabilities and $\sum_{i=1}^{|g|} \text{softmax}(D)_i = 1$. Now O can be defined by

$$O(hard|S_{(N_i,g)}, \cdot) = \text{softmax}(D)_i \text{ and}$$
$$O(easy|S_{(N_i,g)}, \cdot) = 1 - \text{softmax}(D)_i.$$

In a POMDP the agent has beliefs $b(s)$ over states s. $b(s)$ is the probability of being in state s. At each step the agent has a belief state over all $s \in S$.

Definition 16. *(Belief State)*
The belief state B is defined by $B = \{b(s) | s \in S\}$.

$POMDP^*_{\Upsilon}$ is solved with *pomdp-solve*[3]. The result are 3 alpha vectors $\{\alpha_{decide}, \alpha_{random}, \alpha inactive\}$ for each action. The function $\alpha(a)$ returns the alpha vector for a given action $a \in A$. The alpha vectors are used for computing the optimal action for a given belief state B. The optimal action a^* for given alpha vectors and a belief state B is computed by

$$a^* = \text{argmax}(\alpha(a) \cdot B), \tag{13}$$

where $\cdot$ denotes the dot-product.

In the MDP^*_{Υ} the Tokenholders always know the number of Tokenholders in the grand coalition and whether N is even or odd. With the $POMDP^*_{\Upsilon}$ it can be modeled that the Tokenholders have uncertainty about N and whether it is even or odd. To create a new policy, the states $S_{(N_i,g)}$ are transformed into belief states. $B(S_{(N_i,g)})$ denotes the belief state, where a Tokenholder beliefs that N is most probable N_i. Algorithm 2 depicts the procedure for computing $B(S_{(N_i,g)})$. Figure 6 depicts the Belief State for $N_i = 100$. For each $B(S_{(N_i,g)})$ an optimal action can be computed with Equation 13. Table 5 depicts the optimal actions for each belief state $B(S_{(N_i,g)})$.

Algorithm 2 Generate Belief State

procedure GENERATE_BELIEF_STATE($S_{(N_i,g)}$, surrounding)
- $b(S_0) \leftarrow 0$
- $b(q_{winning\_coalition}), b(q_{losing\_coalition}), b(q_{tie}) \leftarrow 0$
- $b(S_{(N_{i-surrounding},g)}), \ldots, b(S_{(N_i,g)}), \ldots, b(S_{(N_{i+surrounding},g)}) \leftarrow 100$
- **for** $i \in \{1, \ldots, 100\}$ **do**
  - $b(S_{(N_{i-surrounding-i},g)}) \leftarrow 100 - i$
  - $b(S_{(N_{i+surrounding+i},g)}) \leftarrow 100 - i$
- $B = b(S_0) \cup \{b(S_{(1,g)}), \ldots, b(S_{(|g|,g)})\} \cup \{b(q_{winning\_coalition}), b(q_{losing\_coalition}), b(q_{tie})\}$
- $B = softmax({}^{1.1}\log B)$

Comparing Table 4 and 5 it can be seen that in Table 5 the Tokenholder is not distinguishing between odd and even N. It can be seen that with $N > 173$, the Tokenholder decides to take the action *inactive*. This is due to the decreasing reward with higher N. With higher N the decision of the Tokenholder

[3] http://www.pomdp.org/code/index.html

State	$B(S_{(1,g)})$ decide	$B(S_{(2,g)})$ decide	$B(S_{(3,g)})$ decide	$B(S_{(4,g)})$ decide	$B(S_{(5,g)})$ decide	$B(S_{(6,g)})$ decide	$B(S_{(7,g)})$ decide	$B(B(S_{(8,g)}))$ decide	$B(S_{(9,g)})$ decide	$B(S_{(10,g)})$ decide	$B(S_{(11,g)})$ decide
	$B(S_{(12,g)})$ decide	$B(S_{(13,g)})$ decide	$B(S_{(14,g)})$ decide	$B(S_{(15,g)})$ decide	$B(S_{(16,g)})$ decide	$B(S_{(17,g)})$ decide	$B(S_{(18,g)})$ decide	$B(S_{(19,g)})$ decide	$B(S_{(20,g)})$ decide	$B(S_{(21,g)})$ decide	$B(S_{(22,g)})$ decide
	$B(S_{(23,g)})$ decide	$B(S_{(24,g)})$ decide	$B(S_{(25,g)})$ decide	$B(S_{(26,g)})$ decide	$B(S_{(27,g)})$ decide	$B(S_{(28,g)})$ decide	$B(S_{(29,g)})$ decide	$B(S_{(30,g)})$ decide	$B(S_{(31,g)})$ decide	$B(S_{(32,g)})$ decide	$B(S_{(33,g)})$ decide
	$B(S_{(34,g)})$ decide	$B(S_{(35,g)})$ decide	$B(S_{(36,g)})$ decide	$B(S_{(37,g)})$ decide	$B(S_{(38,g)})$ decide	$B(S_{(39,g)})$ decide	$B(S_{(40,g)})$ decide	$B(S_{(41,g)})$ decide	$B(S_{(42,g)})$ decide	$B(S_{(43,g)})$ decide	$B(S_{(44,g)})$ decide
	$B(S_{(45,g)})$ decide	$B(S_{(46,g)})$ decide	$B(S_{(47,g)})$ decide	$B(S_{(48,g)})$ decide	$B(S_{(49,g)})$ decide	$B(S_{(50,g)})$ decide	$B(S_{(51,g)})$ decide	$B(S_{(52,g)})$ decide	$B(S_{(53,g)})$ decide	$B(S_{(54,g)})$ decide	$B(S_{(55,g)})$ decide
	$B(S_{(56,g)})$ decide	$B(S_{(57,g)})$ decide	$B(S_{(58,g)})$ decide	$B(S_{(59,g)})$ decide	$B(S_{(60,g)})$ decide	$B(S_{(61,g)})$ decide	$B(S_{(62,g)})$ decide	$B(S_{(63,g)})$ decide	$B(S_{(64,g)})$ decide	$B(S_{(65,g)})$ decide	$B(S_{(66,g)})$ decide
	$B(S_{(67,g)})$ decide	$B(S_{(68,g)})$ decide	$B(S_{(69,g)})$ decide	$B(S_{(70,g)})$ decide	$B(S_{(71,g)})$ decide	$B(S_{(72,g)})$ decide	$B(S_{(73,g)})$ decide	$B(S_{(74,g)})$ decide	$B(S_{(75,g)})$ decide	$B(S_{(76,g)})$ decide	$B(S_{(77,g)})$ decide
	$B(S_{(78,g)})$ decide	$B(S_{(79,g)})$ decide	$B(S_{(80,g)})$ decide	$B(S_{(81,g)})$ decide	$B(S_{(82,g)})$ decide	$B(S_{(83,g)})$ decide	$B(S_{(84,g)})$ decide	$B(S_{(85,g)})$ decide	$B(S_{(86,g)})$ decide	$B(S_{(87,g)})$ decide	$B(S_{(88,g)})$ decide
	$B(S_{(89,g)})$ decide	$B(S_{(90,g)})$ decide	$B(S_{(91,g)})$ decide	$B(S_{(92,g)})$ decide	$B(S_{(93,g)})$ decide	$B(S_{(94,g)})$ decide	$B(S_{(95,g)})$ decide	$B(S_{(96,g)})$ decide	$B(S_{(97,g)})$ decide	$B(S_{(98,g)})$ decide	$B(S_{(99,g)})$ decide
	$B(S_{(100,g)})$ decide	$B(S_{(101,g)})$ decide	$B(S_{(102,g)})$ decide	$B(S_{(103,g)})$ decide	$B(S_{(104,g)})$ decide	$B(S_{(105,g)})$ decide	$B(S_{(106,g)})$ decide	$B(S_{(107,g)})$ decide	$B(S_{(108,g)})$ decide	$B(S_{(109,g)})$ decide	$B(S_{(110,g)})$ decide
	$B(S_{(111,g)})$ decide	$B(S_{(112,g)})$ decide	$B(S_{(113,g)})$ decide	$B(S_{(114,g)})$ decide	$B(S_{(115,g)})$ decide	$B(S_{(116,g)})$ decide	$B(S_{(117,g)})$ decide	$B(S_{(118,g)})$ decide	$B(S_{(119,g)})$ decide	$B(S_{(120,g)})$ decide	$B(S_{(121,g)})$ decide
	$B(S_{(122,g)})$ decide	$B(S_{(123,g)})$ decide	$B(S_{(124,g)})$ decide	$B(S_{(125,g)})$ decide	$B(S_{(126,g)})$ decide	$B(S_{(127,g)})$ decide	$B(S_{(128,g)})$ decide	$B(S_{(129,g)})$ decide	$B(S_{(130,g)})$ decide	$B(S_{(131,g)})$ decide	$B(S_{(132,g)})$ decide
	$B(S_{(133,g)})$ decide	$B(S_{(134,g)})$ decide	$B(S_{(135,g)})$ decide	$B(S_{(136,g)})$ decide	$B(S_{(137,g)})$ decide	$B(S_{(138,g)})$ decide	$B(S_{(139,g)})$ decide	$B(S_{(140,g)})$ decide	$B(S_{(141,g)})$ decide	$B(S_{(142,g)})$ decide	$B(S_{(143,g)})$ decide
	$B(S_{(144,g)})$ decide	$B(S_{(145,g)})$ decide	$B(S_{(146,g)})$ decide	$B(S_{(147,g)})$ decide	$B(S_{(148,g)})$ decide	$B(S_{(149,g)})$ decide	$B(S_{(150,g)})$ decide	$B(S_{(151,g)})$ decide	$B(S_{(152,g)})$ decide	$B(S_{(153,g)})$ decide	$B(S_{(154,g)})$ decide
	$B(S_{(155,g)})$ decide	$B(S_{(156,g)})$ decide	$B(S_{(157,g)})$ decide	$B(S_{(158,g)})$ decide	$B(S_{(159,g)})$ decide	$B(S_{(160,g)})$ decide	$B(S_{(161,g)})$ decide	$B(S_{(162,g)})$ decide	$B(S_{(163,g)})$ decide	$B(S_{(164,g)})$ decide	$B(S_{(165,g)})$ decide
	$B(S_{(166,g)})$ decide	$B(S_{(167,g)})$ decide	$B(S_{(168,g)})$ decide	$B(S_{(169,g)})$ decide	$B(S_{(170,g)})$ decide	$B(S_{(171,g)})$ decide	$B(S_{(172,g)})$ decide	$B(S_{(173,g)})$ decide	$B(S_{(174,g)})$ **inactive**	$B(S_{(175,g)})$ **inactive**	$B(S_{(176,g)})$ **inactive**
	$B(S_{(177,g)})$ **inactive**	$B(S_{(178,g)})$ **inactive**	$B(S_{(179,g)})$ **inactive**	$B(S_{(180,g)})$ **inactive**	$B(S_{(181,g)})$ **inactive**	$B(S_{(182,g)})$ **inactive**	$B(S_{(183,g)})$ **inactive**	$B(S_{(184,g)})$ **inactive**	$B(S_{(185,g)})$ **inactive**	$B(S_{(186,g)})$ **inactive**	$B(S_{(187,g)})$ **inactive**
	$B(S_{(188,g)})$ **inactive**	$B(S_{(189,g)})$ **inactive**	$B(S_{(190,g)})$ **inactive**	$B(S_{(191,g)})$ **inactive**	$B(S_{(192,g)})$ **inactive**	$B(S_{(193,g)})$ **inactive**	$B(S_{(194,g)})$ **inactive**	$B(S_{(195,g)})$ **inactive**	$B(S_{(196,g)})$ **inactive**	$B(S_{(197,g)})$ **inactive**	$B(S_{(198,g)})$ **inactive**
	$B(S_{(199,g)})$ **inactive**	$B(S_{(200,g)})$ **inactive**									

Table 5: Optimal actions for Belief States

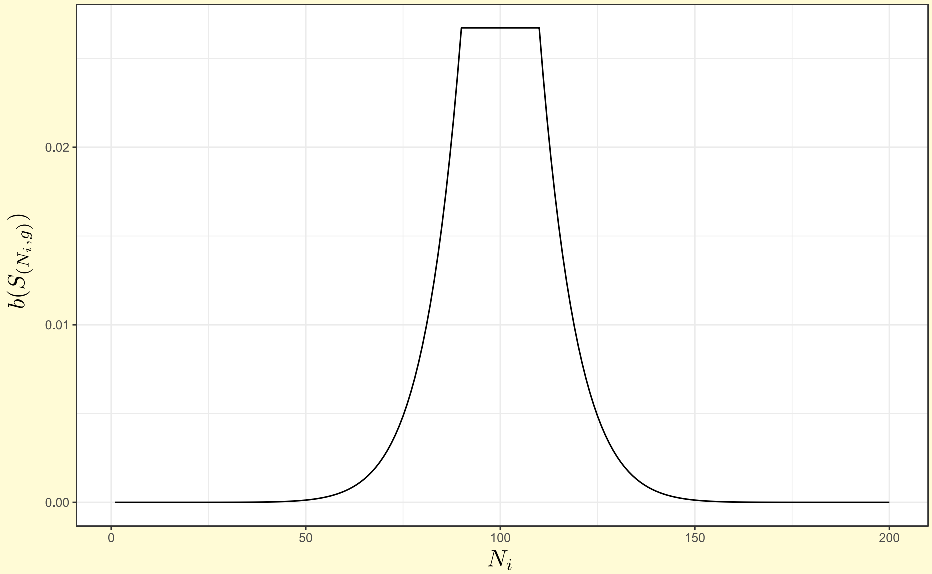

Fig. 6: Belief State for $N_i = 100$

becomes similar to the *El Farol Bar problem* [1]. The solution to this problem is to allow *mixed strategies*. This means that the Tokenholders will select probabilistic actions.

5 Distribution of the Fees

In this section the distribution of the fees among the Tokenholders is described. Each Tokenholder, who actively participates in the Tokenholder Tribunal gets a share of the fees $\xi(o)$ collected in a given dispute cycle o. Remind that the vote share for a given dispute δ and Tokenholder τ is $vs(\tau, \delta)$. The vote share for a given dispute cycle o is

$$vs(\tau, o) = \sum_{\delta \in \Delta_o} vs(\tau, \delta) \tag{14}$$

which is the sum of all vote shares from disputes $\delta \in \Delta_o$. Tokenholders τ, with higher $v(\tau)$ should have a higher vote share, therefore the weighted vote share $vs_w(\tau, o)$ is computed by

$$vs_w(\tau, o) = vs(\tau, o)v(\tau, t_e(o)). \tag{15}$$

Finally, to compute the share of the fees, the sum of all vs_w for $\tau \in T_o$ is computed by

$$VS_w^o = \sum_{\tau \in T_o} vs_w(\tau, o). \tag{16}$$

Now the portion of fees τ gets in vote cycle o is

$$fees(\tau, o) = \frac{vs_w(\tau, o)}{VS_w^o}\xi(o). \tag{17}$$

6 Conclusion

In this yellowpaper a formalization of the Tokenholder Tribunal was given. Moreover, the Tokenholder Tribunal was analyzed and the results were discussed in detail. The analysis showed that the optimal policy for Tokenholders is to make competent decisions. Finally the computation of the fees, that are distributed among the Tokenholders was presented.

References

1. W Brian Arthur. Inductive reasoning and bounded rationality. *American Economic Review*, 84(2):406–11, 1994.
2. Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 2)*, AAAI'94, pages 1023–1028, Menlo Park, CA, USA, 1994. American Association for Artificial Intelligence.
3. Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99 – 134, 1998.
4. Lodewijk Kallenberg. *Finite State and Action MDPS*, pages 21–87. Springer US, Boston, MA, 2002.
5. Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML'95, pages 362–370, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
6. Martin J. Osborne and Ariel Rubinstein. *A course in game theory.* The MIT Press, Cambridge, USA, 1994. electronic edition.
7. Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger, 2014.