The Innovation Judge Protocol

tphanson@kambria.io
mtsing@kambria.io
Kambria, 2019 January 3

Abstract

Currently, winners of hackathons as well as high tech prize competitions (together, “Competitions”) are determined by a group of judges who select the winning team. Such judges may be invited by the organizers of hackathons and bounties based on their expertise, technical proficiency and/or relationships. Judges typically range from investors and sponsoring companies to technologists. However, this centralized system of i) judge selection and ii) judge discernment are problematic. First, judges may not possess sufficient knowledge of the theme or the subject matter of the Competition. In addition, non-technical judges may ask the wrong questions or they may not understand the intricacy of how code interplays with the business model. Furthermore, judges may easily be subjective in their discernment, even though they are often asked to judge teams based on creativity, innovation, purpose, quality of the code, the theme of the hackathon and business model among other criteria.

As a distributed ledger and without a central intermediary, blockchain allows for a consensus mechanism that disrupts the selection of judges, because better governance for judges would result in an outcome that takes into consideration quality, inclusion and fairness. Consensus mechanism generally validates the next block in the chain and prevents adversaries from attacking and forking the blockchain. Consensus models typically include Proof of Work, Proof of Stake, Delegated Proof of Stake and Practical Byzantine Fault Tolerance (PBFT). This paper outlines a consensus mechanism for the Kambria Network that implements a more reliable and better governance for judges.

In addition, this paper set forth terms, mathematics, probability and game theory that help to assure that judges adhere to a set of predefined rules. By taking into consideration potential conflict of interests, inherent bias and subjectivity and possible malevolent intent, the protocol contains two prongs that would keep it robust and help maintain a fault tolerant of ½ through our consensus model. The consensus will fell in case of fault tolerant greater than ½. However, it is possible to keep this parameter less than the threshold of ½ by a set of predefined rules. Thereafter, the consensus will be reliable and entirely able to implement in practice.

1. Introduction

Hackathons are often designed to create functional software or hardware to solve a particular problem by the end of the event. Software developers and other subject matter experts form teams and collaborate intensively at the beginning of the morning throughout the day. At the end of hackathons, each team would demonstrate their solution and present their results. A panel of judges would select the winning teams for the prize. For example, Salesforce had run a hackathon with a prize of $1 million to the winners and TechCrunch had offered a $250,000 prize for a social gaming hackathon. Such hackathons leverage the collective technical minds of the community to solve a problem utilizing a new field of technology, to develop new software technologies within a short amount of time, and/or to locate innovation for funding and further development.

In the same spirit as hackathons, many organizations run public competitions to tap the power of the tech collective, looking for solutions from everything from AI to spacecraft. For example, NASA sponsored its Unmanned Aircraft Systems Airspace Operations Challenge that focuses on a drone's ability to sense and avoid air traffic. During the Google Lunar XPRIZE, a team must place a robot on the moon's surface successfully that would explore at least 500 meters of the moon and transmit images and video back to Earth. Likewise, the Brain Preservation Foundation had announced a cash prize for the first individual to preserve an entire human brain for over 100 years such that every neuronal process and synaptic connection must remain traceable and intact using electron microscopic technologies.

A critical component of hackathons and public competitions is the role of judges. At many hackathons, the judges are composed of organizers and sponsors. Similarly, judges for public competitions also comprise of organizers and sponsors. Their task is to evaluate each team and their solutions prior to selecting and determining a winning team for the prize. Ideally, criteria for each team includes the business value and technical complexity of their solution, their relative wow factor, user experience and design, whether their solution is functional, and whether they are innovative. Such criteria serves as a guideline for judges especially those who have never been a judge nor been to an event to know what to expect. Despite a list of criteria such as usefulness, originality, impact and complexity however, the rating by the judges may not be fair.

The biggest challenge with hackathons and public competitions lie the governance for judges and the judging process. For instance, are hackathons merely a way for organizations and their sponsors to crowdsource ideas from the tech community without compensation? Are organizations running a public competition to harvest intellectual property from the crowd with no intent whatsoever to select a winner? Is there an inherent conflict of interest if the organizers of hackathons or public competitions also serve as judges? Would the judges award the prize to a polished functional solution even though it seemed to be obviously created prior to the hackathon?

To solve the risk of subjectivity and malevolent intent of the judges, this paper proposes a new consensus that resolves the existing limitations of governance.

2. Definitions

- A “Backer” is an entity or individual that organizes and/or funds a Competition.
- A “Bounty” is a prize given to the winning team from a hackathon or a public competition.
- A “Competition” is a hackathon or a public competition.
- A “Developer” is a developer that is part of a Competition.
- A “Judge” is an individual that judges the Competition and takes part in the determination of a winner of the Bounty.
- A “Stake” is a deposit made by a Judge in order to qualify as a Judge.
- A “Backer’s Judge” (J_B) is a Judge selected by the Backer.
- A “Developer’s Judge” (J_D) is a Judge that is selected by the teams in the Competition.
- A “Malicious Developer’s Judge” (J_{MD}) is a Developer’s Judge with malicious intent.
- A “Honest Developer’s Judge” (J_{HD}) is a Developer’s Judge with no malicious intent that would judge the Competition fairly.
- “No winner” / “Having winners” (nw/hw) means no winner or having winners submitted by a Judge.
- “No winner” / “Having winners” (NW/HW) means no winner or having winners as determined by the consensus and represents the truth. Note that “no winner” / “having winners” will be lowercase when it is submitted by a Judge and uppercase when it is the truth.
- A “Bribery-attack” means that group A exchanges some interests with a group B for B to release some decisions that are advantage to group A in some consistent contexts.
- A “Bribed Developer’s Judge” (J_{BD}) is a Developer’s Judge bribed by a Backer.

3. Judge Selection

Ideally, a Judge would be an expert in the field that corresponds to the theme of the Competition. Because a Backer deeply understands the requirements of the Competition and Bounty, he or she could judge the teams more effectively. Therefore, the consensus allows the Backer to be a Judge.

However, the final outcome of the Judges will not be fair and may be subjective if the Backer selects the rest of the Judges or if the Judges are selected randomly especially if the Backer predetermines that he or she does not want anyone to win. For example, the Backer has decided that the Competition has no winner so he would not have to provide a Bounty even though some of the teams’ projects meet the criteria, then the Backer has learned the solutions without paying the Bounty. To protect the best interest of the teams of the Competition, the consensus let the teams select the rest of the Judges.

To ensure the highest integrity of the Judges, the Protocol requires a Judge to provide a Stake as a prerequisite. Such Stake includes fiat, coin or token. For the purposes of this paper, we will use token as the Stake.

A Judge with a big Stake has higher responsibility and accountability than a Judge with a small Stake as they have more to lose. Objectively, the Protocol does consider that the power of a Judge's determination is the same - it must be based on the amount of a Judge's staked token. $|X|$ symbolizes the number of staked token that is equal to the power of Judge X.

For example: If Judge A with 1000 staked token and judge B with 700 staked token then,

$$|A| = 1000$$
$$|B| = 700$$

If A choose r_1 as the result, B choose r_2 as the result, the final result will be r_1 because of $|A| > |B|$.

** From now, for brief expression, this paper will use X to mean Judge X and power $|X|$.*

4. The Implications of Malicious Judges

A J_B has a tendency of protecting the interest of the Backer by allowing the result be "no winner" (nw) (1.1). For instance, the Backer may intend to learn from qualified teams without having to pay out a Bounty. On the contrary, a J_D has the best interest of the team and protect the team's interests and regularly submits "having winners" (hw) (1.2) to the final result. For instance, Developers may want a Bounty even if they could not find satisfactory solutions to the Competition. In addition, some teams may bribe some of the J_D, ensuring that the Bounty is given despite failing to meet Competition requirements and criteria.

We will place the J_D into two subgroups: i) the malicious Developer's Judge (J_{MD}) who would not judge the Competition in good faith and ii) the honest Developer's Judge (J_{HD}) with no malicious intent who would judge the Competition fairly.

Developer's Judge equation: $J_D = J_{MD} + J_{HD}$

Notice that when comparing (1.1) of J_B and (1.2) of J_{MD}, their interests always conflict with each other. In other words, J_B and J_{MD} can not be harmonized. When it benefits J_B, then J_{MD} will be disadvantaged and vice versa. On the other hand, J_{HD} has a pure intent, for J_{HD} always respects the truth. By using that analysis, the consensus will provide safeguards against the motivation of malicious actions.

5. Bribery-attack

In the context of Competitions and Bounties, Bribery-attack may be in two forms:

- Form 1: A team brides a J_D and it attracts the J_{MD}.
- Form 2: A Backer bribes a J_D to generate a new subgroup - J_{BD} - and this Bribed Developer's Judge belongs to J_D although it protects the interests of the Backer.

In Form 1, the structure of the problem that this paper is examining is not modified because this paper considers this as an obvious factor to be analyze herein.

In Form 2, with the arrival of a new subgroup in J_D, the structure of the original problem is destroyed. The J_{BD} must to be separated and observed discreetly.

Developer's Judge equation with backer-bribery-attack: $J_D = J_{BD} + J_{MD} + J_{HD}$

Assuming that every party who joins the Competition is selfish, it means they do not have the best interest of the others at heart and merely wish to maximize their own self-interest.

Backers may conduct a bribery-attack when they want to learn from qualified teams without having to pay a Bounty or when they suspect that Developers have already operated a bribery-attack and intend to conduct another bribery-attack against the attack of the Developer.

A Developer may conduct a bribery-attack when they want to grab the Bounty without producing any satisfactory project or when they suspect that the Backer has already done a bribery-attack and they intend to conduct another attack to protect their interest.

Conventions,

- Total profit of satisfied projects is equal to the value of Bounty and equal to 2.
  If it does not have any qualified project, then that total profit is equal to 0.

$$if\ HW : \Sigma \left| projects_{HW} \right| = 2$$
$$if\ NW : \Sigma \left| projects_{NW} \right| = 0$$

- The cost for a bribery-attack is 1 and this is the same for the Backer and the Developer.

$$cost = 1$$

- The Backer can learn from qualified teams through getting in touch with such teams in the Competition and the value is 3/2 (For example, listening to the representation, discussing with other judges or questioning teams). The value is 3/2 because the Backer spends a cost (which equals to 1) to reject paying a Bounty and obtains insights from learning from qualified projects. Therefore, the advantage of the insights should be greater than 1. Without the direct support from team after the Competition or not owning the original project, the Backer can not gain 100% advantages from the projects (which

is equal to 2); therefore, the advantages of insights should be less than 2. Finally, the value of 3/2 is understandable when $2 > 3/2 > 1$.

$$\Sigma |insight| = 3/2$$

- The advantages or disadvantages of the Backer and Developer are deterministic such as whether or not there will be a bribery-attack because they will happen. Thus, we can skip them in examination.

As set forth by the Conventions discussed above, the payoff matrix can be deduced in two cases, HW and NW:

(Developer, Backer)	Malicious	Honest
Malicious	$(-1, -1)$	$(1, -2)$
Honest	$(0, -1)$	$(0, 0)$

Table 1: Payoff matrix in case of NW

In case of NW:

When both the Backer and the Developer are honest, they do not obtain any personal advantage or disadvantage so the payoff is 0.

$$Payoff(Developer) = 0$$
$$Payoff(Backer) = 0$$

When the Developer is malicious and the Backer is honest, the Developer will grab the Bounty by spending a few cost for a Bribery-attack and the Backer will lose their Bounty with valueless projects.

$$Payoff(Developer) = -cost - \Sigma \left|projects_{NW}\right| + bounty = -1 - 0 + 2 = 1$$
$$Payoff(Backer) = -bounty + \Sigma \left|projects_{NW}\right| = -2 + 0 = -2$$

When the Developer is honest and the Backer is malicious, the attack of the Backer is nonsensical with spending a cost for Bribery-attack to change nothing (the Backer creates a Bribery-attack to let the result be nw).

$$Payoff(Developer) = 0$$
$$Payoff(Backer) = -cost = -1$$

When the Backer and the Developer are also malicious, then the attack of the Backer is against the attack of Developer to protect the Bounty, then the Developer cannot take the Bounty.

$$Payoff(Developer) = -cost = -1$$
$$Payoff(Backer) = -cost = -1$$

(Developer, Backer)	Malicious	Honest
Malicious	$(-1, -1)$	$(-1, 0)$
Honest	$(-\frac{3}{2}, \frac{1}{2})$	$(0, 0)$

Table 2: Payoff matrix in case of HW

In case of HW:

When both the Backer and the Developer are honest, the Backer will release the Bounty to qualified teams and the Developer will submit the project and receive the Bounty.

$$Payoff(Developer) = -\Sigma\left|projects_{HW}\right| + bounty = -2 + 2 = 0$$
$$Payoff(Backer) = -bounty + \Sigma\left|projects_{HW}\right| = -2 + 2 = 0$$

When the Developer is malicious and the Backer is honest, the attack of Developer is nonsensical to change nothing far away from the truth.

$$Payoff(Developer) = -\Sigma\left|projects_{HW}\right| + bounty - cost = -2 + 2 - 1 = -1$$
$$Payoff(Backer) = -bounty + \Sigma\left|projects_{HW}\right| = -2 + 2 = 0$$

When Developer is honest and the Backer is malicious, the Backer is willing to pay a small cost and take advantage of the project partly without paying a Bounty and the Developer observes the Backer's action as negative.

$$Payoff(Developer) = -\Sigma|insight| = -3/2$$
$$Payoff(Backer) = -cost + \Sigma|insight| = -1 + 3/2 = 1/2$$

When the Backer and the Developer are also malicious, then the attack of the Developer is against the attack of the Backer. In other words, Backer cannot reject paying the Bounty.

$$Payoff(Developer) = -\Sigma\left|projects_{HW}\right| + bounty - cost = -2 + 2 - 1 = -1$$
$$Payoff(Backer) = -bounty + \Sigma\left|projects_{HW}\right| - cost = -2 + 2 - 1 = -1$$

Using the two tables above and p is the possibility of HW, then $1-p$ is the possibility of NW:

(Developer, Backer)	Malicious	Honest
Malicious	$(-1,\ -1)$	$(1-2p,\ 2p-2)$
Honest	$(-\frac{3p}{2},\ \frac{3p}{2}-1)$	$(0,\ 0)$

Table 3: General payoff matrix

At $p = \frac{1}{2}$,

(Developer, Backer)	Malicious	Honest
Malicious	$(-1,\ -1)$	$(0,\ -1)$
Honest	$(-\frac{3}{4},\ -\frac{1}{4})$	$(0,\ 0)$

Table 4: $p = \frac{1}{2}$ payoff matrix

With $p = \frac{1}{2}$, Nash equilibrium is at $(Honest,\ Honest)$. Thus, the problem of bribery could be solved if the consensus keeps $p = \frac{1}{2}$ with the assumption that all the parties tend to maximize their payoff.

In detail, $p = \frac{1}{2}$ means no party knows or decides NW/HW in advance. Coming back to the traditional problem, to select the prizes, it needs to answer 2 main questions:

- Are there winners?
- What is the order of winners?

Pertaining to the question “Are there winners?”, this is the main factor which affects the parameter p - if the answer is “yes” so $p = 1$, if the answer is “no” so $p = 0$. In order to keep $p = \frac{1}{2}$, the consensus must keep it in secret until at the end of the Bounty after all the Judges submit their decisions by the commitment scheme set forth in Section 6.

Comment 1: *In the process of submission, some Judges that have not yet submit results can estimate the parameter p by basing on the results that the other Judges had submitted. Thus, all the results will be kept in secret but can be verified if it is changed after all Judges have submitted their results.*

6. The Commitment Scheme

The commitment scheme is a cryptographic primitive which allows one to commit to certain message m as commitment c, with the ability to later “open” c and show that m was honestly committed. The commitment scheme provides two important properties:

- Hiding: Given c, it is infeasible to learn any information about m.
  (This prevents Judges from gaining any knowledge about the selections in advance.)
- Binding: It is infeasible to open c to two distinct messages m_0, m_1.
  (This prevents Judges from changing their selections after they have provided their submissions.)

For a rock-scissors-paper between two Judges on the blockchain for example, the commitment scheme would work as follows:

- Round 1: the Judges must submit their hash of combination of their selection and a secret random number.
- Round 2: the Judges must submit their selection and the secret random number in raw.

The secret random number here is as a zero-knowledge factor. Because the sample space of game is just 3 selections, it is too small and feasible to guess the result. The hash becomes a commitment and it is based on the secure hash functions. Therefore, this scheme ensures fairness:

- The two Judges cannot gain any knowledge about the selections in advance (hiding).
- After disclosure phase, the two Judges cannot change their selections (binding).

7. 2-step Judge Sequencing Algorithm

After the analysis in Section 5, it is easy to see that to optimize the security of consensus, the order of judging should follow 2 steps:

- Step 1: Arrange the serial order of all projects based on its quality
- Step 2: Decide whether or not there are winners

Competitions generally define how many winners are at the beginning. Therefore, in Step 2, if the answer is yes, all prizes will be occupied and if the answer is no, then no one could be a winner. For example, in a Bounty, a rule may indicate that there are 3 prizes: the first, second and third. Either all prizes will have winning teams or nothing. It cannot be that the first and second have winning teams but not the third.

a. The serial order of teams

In Step 1, J_B could make two decisions. First, J_B could submit a correct result. Secondly, J_B could submit an incorrect result. The decision in Step 1 can affect Step 2 as well as the final J_B payoff. This is illustrated in Table 5 below:

(Outcome Step 2, Decision of J_B at Step 1)	Malicious	Honest
HW	$\frac{1}{2}$	0
NW	-1	0

Table 5: J_B Payoff

If J_B was honest in Step 1, no matter what the result in Step 2 is, J_B would always be satisfied. Therefore, the outcome should be 0. In terms of J_B being malicious, that means J_B chose the concrete team who would return the Bounty back to the Backer. The assumption is that all values are the same in Section 5. Deducing:

$$Payoff(HW,\ Malicious) = -cost + \Sigma|insight| = -1 + 3/2 = 1/2$$

$$Payoff(NW,\ Malicious) = -cost = -1$$

With the probability of NW happening and HW happening is equal to $p = \frac{1}{2}$, so:

$$Payoff(Honest) = \frac{Payoff(HW, Honest) + Payoff(NW, Honest)}{2} = 0$$

$$Payoff(Malicious) = \frac{Payoff(HW, Malicious) + Payoff(NW, Malicious)}{2} = -1/4$$

$$\Rightarrow Payoff(Honest) > Payoff(Malicious)$$

Comment 2: *J_B tends to be honest in Step 1 with condition $p_{NW} = p_{HW} = \frac{1}{2}$. To assure that, the consensus will force J_B out of the Step 2, which means J_B cannot decide NW or HW for the final result.*

Regarding the process of submitting the serial order of teams based on the quality of projects, the consensus rules that all Judges must submit in form of array from lowest quality to highest quality. For example, A is a team with the highest quality of project, after that B, C and D, so the submitted result must be $[D, C, B, A]$. Based on the index of team in array, the score of them is equal to index, $score(D) = 1,\ score(C) = 2,\ score(B) = 3,\ score(A) = 4$, and so on for all arrays. After having the scores from all of the Judges, the consensus keeps calculating to return the final score.

** There are many options to calculate, but in range of the paper we focus only on the average and the median.*

With the average, more precisely, it is the expected value. The final score of a team shall be the average of all scores of Judges for that team. That value represents "a medium value of scores when countless number of Judges (including honest and malicious judges)". Unlike "expected value", that value usually has a very low probability of occurrence. Because of these reasons, the average is not sufficient to find the final score.

In regard to the median, it is a value as the cumulative equation is established with cumulative distribution function P :

$$P(x \leq median) = P(x \geq median) = 0.5$$

Let the score of 1 in 10 teams joined the Bounty, because it has 10 teams so the array will be 10-length and then the possible score of a team should be from 1 to 10. See Table 6.

Score	1	2	3	4	5	6	7	8	9	10
Weighted	0	900	2020	5339	608	0	0	0	6337	0

Table 6: Scores of a team in 10 teams

The weight W of score s is the total power of the Judges who voted score s to that team. We can deduce the probabilities from Table 6 in Table 7:

Score	1	2	3	4	5	6	7	8	9	10
Weighted	0	900	2020	5339	608	0	0	0	6337	0
Probability	0	0.060	0.133	0.351	0.040	0	0	0	0.416	0

Table 7: Probability based on weighted score of a team in 10 teams

Keep calculating from table 7:

$$\mu = the\ expected\ value = 5.867$$

$$P(s \leq 4) = 0.544 > 0.5$$

$$P(s \leq 3) = 0.193 < 0.5$$

Then, $3 < m = median < 4$ at $P(x \leq m) = P(x \geq m) = 0.5$

Although $s = 9$ is the value with the highest probability by most probability of scores is laid around $s = [2, 5]$, therefore the median will be pulled to near area $s = [2, 5]$.

Comment 3: *The median represents the typical value (majority) in probability. Because of those reasons, the median seems meaningful to the Step-1 problem.*

It's not enough to conclude $3 < m < 4$, because Step 1 needs a specific value of m. To define m, the discrete probabilities table should be defined continuously. That means,

$\forall s \in [0, 10],\ there\ exists\ p(s)\ and \int_0^{10} p(x)\,dx = 1 \Leftrightarrow \forall s \in [0, 10],\ \exists w(s)$ with $p(s)$ is the probability and $w(s)$ is the weight at score s of a team.

Thus, to calculate the real positive numbers, the algorithm will be used is Inverse Distance Weighting Interpolation.

In detail, with the specific number of samples n, let $x = \{x_1, ..., x_n\}$ be the existing samples:

$$u(x) = \begin{cases} \frac{\Sigma_{i=1}^{n} \omega_i(x) u(x_i)}{\Sigma_{i=1}^{n} \omega_i(x)} & if\ \forall i, d(x, x_i) \neq 0 \\ u(x_i) & if\ \exists i, d(x, x_i) = 0 \end{cases}$$

$$\omega_i(x) = \frac{1}{d(x, x_i)^4}$$

$$d(x, x_i) = |x - x_i|$$

Basing on the algorithm above, the consensus can calculate m precisely.

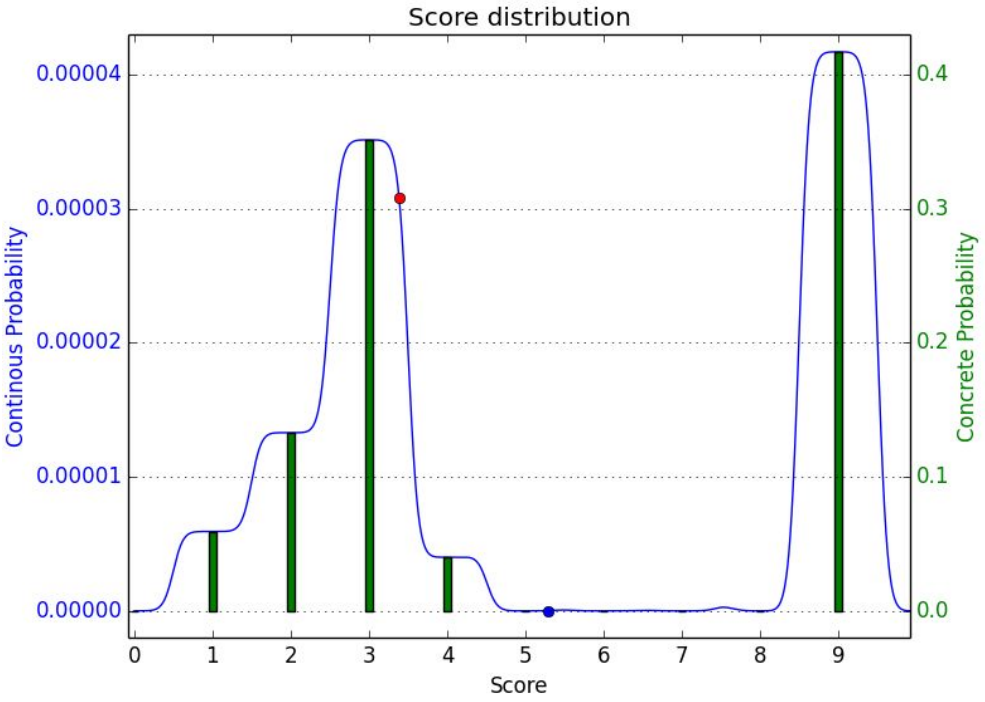

Figure 1: Determined median by using IDW interpolation

In Figure 1, the red point is the median, the blue point is the expected value, the green columns are the original probabilities, and the blue line is the interpolated probabilities.

In cases where there is a tie for first place, the submit time would be referenced as a parameter to sequence those teams.

b. HW or NW

Pertaining to comment 2, the consensus does not allow J_B to join this step; therefore, the decision of NW/HW only relies on J_D.

And now, following the summary of analysis in Section 5, the consensus accepts that there is no malicious Judge any more. To decide whether or not to having winners, the consensus will compare the weight of NW and HW and then select the higher as the final result.

8. What if bribery-attack still happen?

When Judges do not follow the regular rules of game theory, they deflect and become a free-rider, so what would happen to the consensus? Is it still secure?

Notation:

- *The values 0 and 1 mean "not exists" and "exists" respectively.*
- *S1, S2: Step 1, Step 2.*
- *H and M are total power of honest Judges and malicious Judges in each round.*
- *End: the end of the protocol with correct result.*

At the value 1010, it contains no malicious Judge in the protocol, so the Judge can be operated successfully.

At the value 1110, to do a successful attack then $J_{BD} > \frac{J_D}{2}$ (8.1). In case of $J_{BD} < \frac{J_D}{2}$, the attack will not prevail.

At the value 1011, the same as 1110, to do a successful attack then $J_{MD} > \frac{J_D}{2}$. (8.2)

Finally, at the value 1111, in order for J_{BD} and J_{MD} to implement a successful attack, then one of them must own a power greater than the sum of power of J_{HD} plus the other (either J_{BD} or J_{MD}), then $|J_{BD} - J_{MD}| > J_{HD}$.

Doing some transformations:

$$|J_{BD} - J_{MD}| > J_{HD}$$

With $J_{BD} > J_{MD} \Rightarrow |J_{BD} - J_{MD}| > J_{HD}$

$\Leftrightarrow J_{BD} - J_{MD} > J_{HD} \Leftrightarrow J_{BD} > J_{HD} + J_{MD}$

$\Leftrightarrow J_{BD} > J_D - J_{BD} \Leftrightarrow J_{BD} > \frac{J_D}{2}$ (8.3)

With $J_{MD} > J_{BD} \Rightarrow |J_{BD} - J_{MD}| > J_{HD}$

$\Leftrightarrow J_{MD} - J_{BD} > J_{HD} \Leftrightarrow J_{MD} > \frac{J_D}{2}$ (8.4)

From (8.1), (8.2), (8.3) and (8.4), deducing the condition of successful attacks:

$$J_{BD} > \frac{J_D}{2} \vee J_{MD} > \frac{J_D}{2}$$

From the calculations above, the attack seems to be 51% attack. Thus, if the total power of J_D is bigger, the attack is harder to happen. The consensus can use this information as an estimation of security and reliability of the Judge.

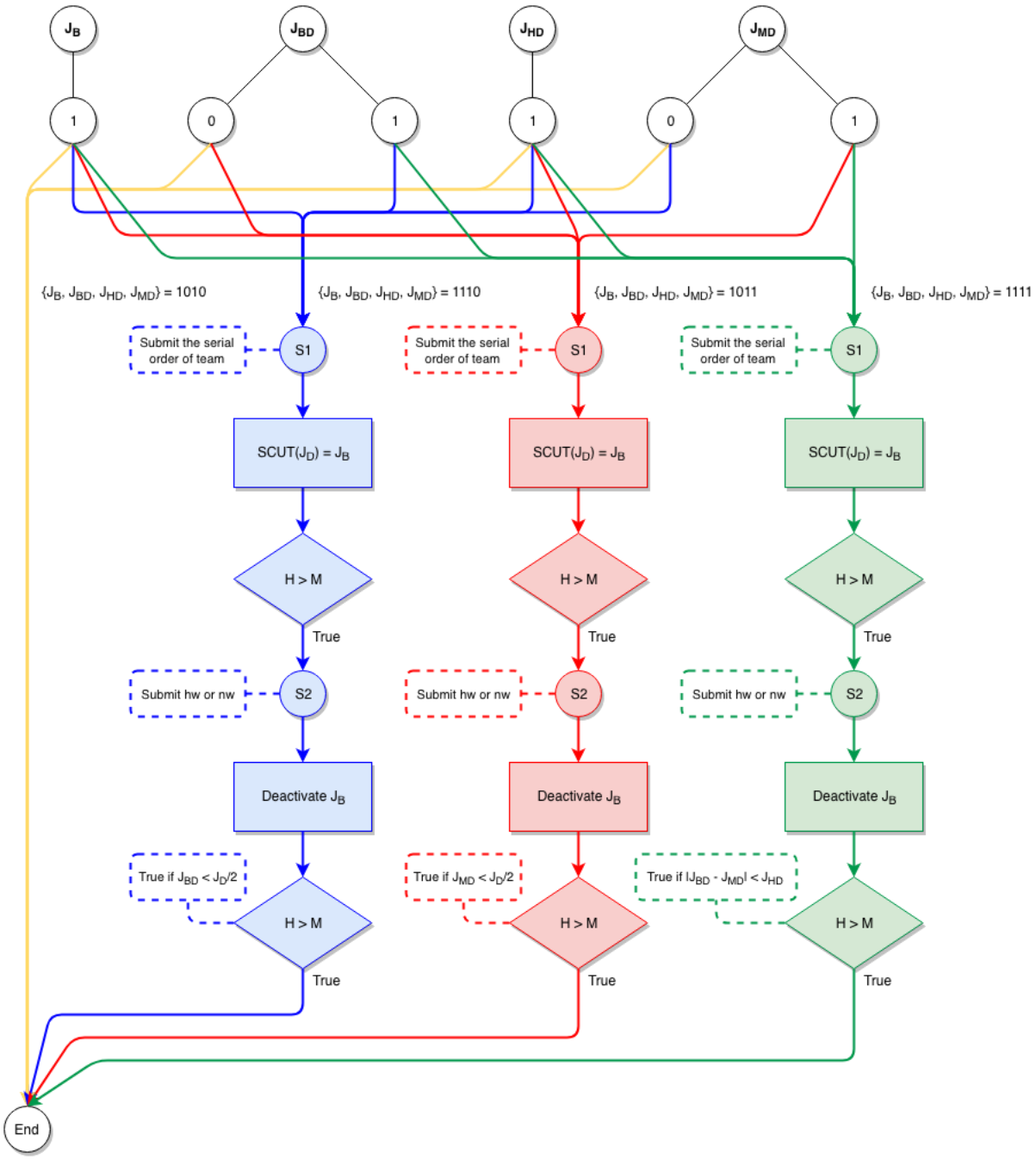

Figure 2: If bribery-attack still happen.

9. The Innovation Judge Protocol

- In first phase, the Bounty must be determined including the value of the Bounty, the requirements of the Bounty, m prizes, et al.
- The community may become a Judge candidate by staking their tokens.

- The Backer may select one Judge from the list of candidates. The Developer may also select Judges from the list of candidates. A Bounty will be regarded highly secure, if the value of Stake and the number of the Judges is large.
- The Stake will be returned to candidates who do not become Judges.
- After the teams submit their projects, every Judge has to decide separately and independently based on the quality of projects and the Bounty's requirements:
  1. The serial order of teams
  2. Whether or not there are winners

Assuming the form of decision as:

$$result_i = [isWon_i,\ team_{i1},\ team_{i2}, ...,\ team_{in}]$$

Therein, the result of Judge is an array $result_i$ with $isWon_i = \{1\ if\ hw,\ 0\ if\ nw\}$ and the serial order of n teams is $team_{i1},\ team_{i2}, ...,\ team_{in}$.

- Judge publishes the value $Hash(result_i + prn_i)$ where prn is pseudo-random number.
- After all Judges publish their hash of decision, they submit $result_i$ and prn_i. The community will verify those values with $Hash(result_i + prn_i)$. If it is incorrect, the consensus will punish them with their Stake.
- Scale the power J_B equal to J_D. Call the score of $team_j$ from $result_i$ submitted by $judge_i$ is $score_{ji}$ with $probability_{ji}$ and the final score $score_j$ will be calculated by getting the median from returned value of IDW basing with sample $\{score_{ji},\ probability_{ji}\}_{i\ \in\ the\ judges}$:

$$score_{ji} = index\ of\ team_j\ in\ result_i \Rightarrow score_j = median(IDW(\{score_{ji},\ probability_{ji}\}_{i\ \in\ the\ judges}))$$

Arrange team position by $score_j$.

- Deactivate J_B. Only J_D can define the value $isWon$.

$$isWon = \{1\ if\ \left|isWon_i == 1\right| \geq \left|isWon_i == 0\right|,\ 0\ if\ \left|isWon_i == 1\right| < \left|isWon_i == 0\right|\}$$

If $isWon = 1$, return m highest teams. If $isWon = 0$, return $null$.

10. Conclusion

We believe that this Innovation Judge Protocol will decentralize the judging of Competitions, ensuring its fairness and consistently and preventing conflicts of interests and malicious actions from Backers and/or Developers who may not have the best interest of the Competitions. We hope to maintain the integrity of the Competitions, ensuring that Developers will not be taken advantage of if they participate in good faith and ensuring that Backers will benefit from qualified teams if they intend to pay out the Bounty to the winning teams. We are confident that this Protocol would protect all.

11. References

1. Anirban. 2011. Bribery – a game theoretic approach. [https://aghatak.wordpress.com/2011/04/21/bribery/]. Accessed November 17, 2018.
2. Basu, K. 2011. Why, for a Class of Bribes, the Act of Giving a Bribe should be Treated as Legal. [http://www.kaushikbasu.org/Act_Giving_Bribe_Legal.pdf]

3. Bentov, I., Gabizon, Ariel. and Mizrahi, A. 2017. Cryptocurrencies without Proof of Work. [https://arxiv.org/pdf/1406.5694.pdf]
4. Brassard, G., Chaum, D. and Crépeau, C. 1988. Minimum Disclosure Proofs of Knowledge. *Journal of Computer and System Sciences*. **37**, pp.156-189.
5. Castro, M. and Liskov, B. 1999. Practical Byzantine Fault Tolerance. *Proceedings of the Third Symposium on Operating Systems Design and Implementation*. pp.173-186.
6. Lamport, L., Shostak, R. and Pease, M. 1982. The Byzantine Generals Problem. *Journal ACM Transactions on Programming Languages and Systems*. **27**(2), pp.228-234.
7. Tendermint. 2018. Byzantine Consensus Algorithm. [https://github.com/tendermint/tendermint/wiki/Byzantine-Consensus-Algorithm]. Accessed October 9, 2018.
8. Token Curated Registry. 2018. What is a Token Curated Registry?. [https://medium.com/@tokencuratedregistry/a-simple-overview-of-token-curated-registries-84e2b7b19a06]. Accessed October 23, 2018.
9. Verma, P. and Sengupta, S. 2015. Bribe and punishment: An evolutionary game-theoretic analysis of bribery. PLoS One 10, e0133441 (2015).
10. Wikipedia. 2018. Commitment scheme. [https://en.wikipedia.org/wiki/Commitment_scheme]. Accessed November 19, 2018.