Harnessing the Intelligence of Crowds

A software concept using an ad hoc, random board protocol
to conduct referendums and juries

Abstract

Co-operating with groups of people, whether for business or community projects, often requires intense discussion for decision making consensus. Proposals must be drafted and aligned with the community mission, and the best proposals demand extra analysis, in order to make the best decisions. The challenges of tapping the communities intelligence are work load per voter and bias. Filtering bias allows the group to operate without exposure to accusations of bias. Referendums are a common strategy which are cheaper now with new technology. Juries are another common strategy because they introduce randomness. Using software, a group can create and administer *referendums* and *juries* quickly, transparently and using a predictable protocol customized according to context. Some of the potential use cases include transparency, charity, arbitration, auditing, design, feedback, community consensus or business management. Communities could use this protocol to create a programmable culture which can manage itself. Regulators could remove opportunities for bias from their operational functions. Employees could both own and control capital without requiring external laws to regulate their compliance. This protocol aims to minimize both work and bias by introducing ad hoc, random boards to support *jury* and *referendum* administration and other tools to assist groups drafting, raising and implementing decisions.

June 2015

arb@keemail.me

Objective

There are two common ways to tap crowd intelligence: referendums and juries. Referendums read the whole community whereas juries consist of only a random subset. There are different applications for each. Referendums survey popularity and are not suited to small teams because proposals can be consistently blocked by small factions. For example, if you have a team of 20 people and they decide to authorize all big spends by a referendum vote greater or equal to 80% then it only requires a faction of 5 to consistently block every proposal. Instead using juries of 4 with 3 votes required to pass means factional blocking is much more difficult. Also referendums demand judgment from every member so maximum attention is required. Each individual must perform analysis and engage with proposals many of which they may not be interested in. Alternatively if no voting is required and proposals are only authorized by interested parties then there are too many opportunities for bias. With a jury, a random subset of people are selected to reduce the work load and the risks of bias. So when juries are used they stifle factions and reduce the inefficiencies of performing a global review.

Leveraging the intelligence of crowds is now a lot easier because of software and the internet. A software protocol can support administration of referendums and juries with features for discussing proposals, ad hoc voting and inserting randomness. A protocol to achieve this is described here and is called ARB because proposals are passed by voting with *Adhoc Random Boards*, which is a random jury formed adhoc. When each board is formed for a vote it has as inputs, 1) a detailed proposal drafted by interested members of the community which includes the expert explanations *for* and *against* the proposal as well as, 2) community polling results from this discussion. The ARB members can investigate the proposal raised by the community and then submit their vote. The whole process is transparent and means the culture can be standardized and refined at a protocol level.

The proposal process

The protocol will automate the work flow of proposals from *open*, to *draft*, to *raised*, to *passed or rejected* status. Firstly, individual members create proposals aligned with the community mission which default to **open** status. Community members are allocated limited *proposal points* which they can spend to score these proposals. Scores may be positive or negative. The member's score feeds into their weighted influence on draft discussions. In this way a member's influence is limited though they can tailor it toward the proposals they care about the most. If an open proposal reaches a score >= the *raise limit* then its **draft** period begins. Once drafted the proposal points are locked thereby regulating the influence members may have on other proposals. All drafted proposals stay as drafts until the draft period expires. Members may vote their preference to extend the draft period if they decide more discussion is required. If enough members vote to extend the draft period then it is extended by the default draft period.

The draft period allows time for other members to notice the proposal is closer to being raised. This gives time for consideration and discussion. Discussion is classified under two sections: *for* and *against*. The discussion points are voted up and down and each member's influence is weighted relative to the proposal points they scored. Each discussion point may have many comments which are also prioritized by weighted voting. When the draft period expires if the score of a draft proposal is >= the raise limit then the proposal is automatically **raised** for voting by an ARB. Otherwise its score is below the raise limit and so it is automatically moved to **rejected** status.

A **raised** proposal automatically causes the formation of an ARB. The size of the ARB and the votes required to pass are pre-configured. After the ARB votes the proposal is set to **passed** or **rejected** and the locked proposal points return to members. The system has a variety of configurable parameters which can be used to customize the process:

community size	The number of members in the community
initial proposal points	The amount of proposal points issued to new members
raise limit	The proposal points required to raise an ARB
default draft period	The period which a proposal will stay a draft before being raised
extend draft period	yes/no per member to indicate preference to extend the draft period of a proposal
board size	The number of members randomly selected for an ARB
pass limit	The number of votes required to pass a proposal
board bias filter	yes/no to exclude members involved with drafted proposals from the ARB
raising fee	A price for automatically raising an ARB
member availability	yes/no per member to indicate when they are available for an ARB

The engine of the protocol is represented by the *member availability* as it is a period of work, perhaps hours, days or a number of ARBs, that each member commits to perform analysis and decide on their vote with an explanation. Members can measure the probability that their committed work will be triggered by their random selection into an ARB. At one raised proposal per day it is the average rate of days between board selection: *proposals raised per day / probability of being selected*. So for example, if a member commits to 20 hours of work in a community with 1000 members and an ARB size of 12 then the probability of being selected is 1.2% (12/1000). If there is one ARB raised per day then there would be 83 days (1/1.2%) on average between the member's selection on ARB to perform their committed work. This is 0.24 hours per day (20/83).

Special proposals

Standard proposals are those focused upon the community mission. There are also some special proposals to be considered and configured. Some of these proposals may be automated which means the protocol creates the proposal instead of being manually created by a member. Proposals may also be configured to be raised for an ARB automatically.

Meta-proposals are a special kind which if passed require a change to the protocol configuration. If meta-proposals are enabled then all of the settings of the protocol are exposed to being updated by a successfully passed proposal.

Member proposals are a special kind which indicate proposals relating to specific members such as recommendations for white or black listing.

Audit proposals are a special kind which are automated to follow up on the performance of other members and giving an auditing score to each board member submission. After repeated low scores a member may be exposed to automated penalties such as suspension from ARB selection or frozen proposal points.

Applications

The ARB protocol is an efficient way of reading crowd intelligence using a process to minimize bias. It's not good at creating quality proposals. Therefore the quality of the output is dependent on the quality of proposals. *Community members need to be incentivized to create quality proposals aligned with the community mission and bind themselves to the outcomes.* Some communities may need to limit membership or require qualifications, while others may be unconditionally open. Each instance of the protocol must be configured with custom settings depending on the objectives and testing results. Not all communities need tools to minimize bias but any community could annex the ARB protocol by using the software.

A company may want to survey certain ideas that filter out bias instead of focusing on popularity. An open source software project could implement ARB among user base to guide development. A social community may want to hold referendums and juries to share management decisions. A startup of crowd funders could keep control of project funds and authorize project spends as a team. A regulatory board could implement it to offer regulation free from accusations of bias. For example the ARB community could be 100 architects that sign-off building plans by allocating juries of 3 architects to each submission for a fee. A charity could implement it so their spending is regulated. For example, a Red Cross division of 100 employees would channel spending authorization through the ARB software and give access to every employee. The spending protocol could then be configured with various authorization levels :

Spend	***Board Size***	***Pass Limit***
<$50	1	1
<$2k	6	4
< $10k	12	8
< $100k	24	16
>= $100k	100	70

Now the Red Cross division's spending is regulated by their employees with different levels of randomized involvement depending on the spend size. The employees should only authorize transactions that are aligned with the Red Cross mission. The settings are always customized to suit the application context.

Probabilities of work and bias

The primary function of the ARB protocol is to minimize work and bias. Only ARB work is measured since proposal drafting must always be done even without the protocol. A faction of biased members may conspire to achieve an outcome not aligned with the community mission. Some probabilities of work and bias can be calculated given specific assumptions:

N	the size of the ARB community
I	the number of initial points issued to each member for scoring proposals
r	the *raise limit* is the proposal points score being targeted to trigger an ARB
F	the number of members in a faction conspiring to insert bias
b	the number of members selected for each ARB
p	the *pass limit* is the number of votes required to pass an ARB
P(W)	the probability of a member being selected for work on a random ARB
1/P(W)	the average rate of a member being selected for work on an ARB
$P(F_{raise})$	the probability of a faction raising a biased proposal given no scoring from non-faction members
$P(F_{ARB})$	the probability of a faction controlling a random ARB sufficiently for passing
$1/P(F_{ARB})$	the average rate that a faction can control a random ARB sufficiently for passing
$P(F_{pass})$	the probability of a faction raising a biased proposal given no negative scoring from non-faction members *and* controlling its ARB sufficiently for passing, given the *board bias filter* option is disabled
$1/P(F_{pass})$	the average rate that a faction can pass a raised proposal
$P(F_{block})$	the probability of a faction biasing an ARB to block the proposal
$1/P(F_{block})$	the average rate that a faction can block a raised proposal

$$P(W) = b / N$$

$$P(F_{raise}) = IF\ (I \times B / r) < 100\%\ THEN\ 0\%\ ELSE\ 100\%$$

To pass a biased ARB requires >= p faction members to be selected. In the formula below all combinations of faction members getting selected onto an ARB in numbers equal to or greater than the pass limit is divided by the total number of combinations for board selection. C{n/k} represents the binomial coefficient for calculating combinations when the order is not important as in selecting members for a board. In excel it is the COMBIN() function. The factorial formula is: C{n/k} = n! / k! (n- k)! .

$$P(F_{ARB}) = (C\{F / (p + i_{0})\} \times C\{(N-F) / (b-(p + i_{0}))\} + ... + C\{F / (p + i_{max})\} * C\{(N-F) / (b-(p + i_{max}))\}) / C\{N / b\}$$

where *i* is a set of integers in the range {0 .. *MIN(b-p,F)*}

The above applies only to a random proposal but if faction members are trying to raise a biased proposal and get it passed they need to bias the ARB of the specific proposal they raised. This requires the *bias board filter* to be disabled since if it were enabled members who score proposals are automatically excluded. For a faction to pass a biased proposal they must first raise a biased proposal and then control the ARB in numbers sufficient to pass.

$$P(F_{pass}) = P(F_{ARB}/F_{raise}) \times P(F_{raise})$$

Blocking is similar to passing but usually is much easier since biased members need only >= b – p + 1 votes to block the proposal, instead of >= p for passing. So to minimize blocking it is best to configure an appropriate gap between the size of ARBs (*b)* and the pass limit (*p)*.

$$g = b – p + 1$$

$$P(F_{block}) = (C\{F / (g + i_{0})\} \times C\{(N-F) / (b-(g + i_{0}))\} + ... + C\{F / (g + i_{max})\} * C\{(N-F) / (b-(g + i_{max})\}) / C\{N / b\}$$

where *i* is a set of integers in the range {0 .. *MIN(b-g,F)* }

Here is a table with a variety of configuration parameters and their probabilities of work and bias :

#	N	I	r	F	b	p	P(W)	1/P(W)	$P(F_{raise})$	$P(F_{ARB})$	$P(F_{pass})$	$1/P(F_{pass})$	$P(F_{block})$	$1/P(F_{block})$
1	20	1	4	5	20	16	100%	1	100%	0%	0%	0	100%	1
2	20	1	12	6	4	3	20%	5	0%	6%	0%	0	34%	3
3	100	1	10	12	6	3	6%	17	100%	2.1%	2.1%	46	0.1%	606
4	100	1	75	20	10	5	10%	10	0%	2.5%	0%	0	0.4%	254
5	1000	1	550	200	10	5	1%	100	0%	3.2%	0%	0	0.6%	163
6	1000	1	550	400	30	20	3%	33	0%	0.25%	0%	0	54%	1.85
7	10000	1	5000	2000	30	20	0.3%	333	0%	TINY	0%	0	2.5%	39
8	10000	1	2000	2000	30	20	0.3%	333	100%	TINY	TINY	HUGE	2.5%	39

N the size of the ARB community
I the number of initial points issued to each member for scoring proposals
r the *raise limit* is the proposal points score being targeted to trigger an ARB
F the number of members in a faction conspiring to insert bias
b the number of members selected for each ARB
p the *pass limit* is the number of votes required to pass an ARB
P(W) the probability of a member being selected for work on a random ARB
1/P(W) the average rate of a member being selected for work on an ARB
$P(F_{raise})$ the probability of a faction raising a biased proposal given no scoring from non-faction members
$P(F_{ARB})$ the probability of a faction controlling a random ARB sufficiently for passing
$1/P(F_{ARB})$ the average rate that a faction can control a random ARB sufficiently for passing
$P(F_{pass})$ the probability of a faction raising a biased proposal given no negative scoring from non-faction members *and* controlling its ARB sufficiently for passing, given the *board bias filter* option is disabled
$1/P(F_{pass})$ the average rate that a faction can pass a raised proposal
$P(F_{block})$ the probability of a faction biasing an ARB to block the proposal
$1/P(F_{block})$ the average rate that a faction can block a raised proposal

On line #3 in the table there are 100 members (*N*) with 1 proposal point (*I*) each therefore at least 10 members (*r*) are required to raise a proposal. There is a risk of bias with 12 members being part of a faction (F). Each proposal will be decided by an ARB made up of 6 members (b) and 3 are required to pass the proposal (p). Each member has a 6% chance (*P(W)*) of being selected to work per ARB which is every 17th (*1/P(W)*) one. The chance that the faction can raise their own biased proposal is 100% ($P(F_{raise})$) since there are 12 in the faction with 1 proposal point each when 10 points are needed to raise a proposal. The chance that 3 members (p) of the faction are selected onto the same random ARB is 2.1% ($P(F_{ARB})$). The chance that 3 members (p) of a faction are selected onto a specific biased proposal they raised on their own is 2.1% ($P(F_{pass})$). That is every 46th proposal ($1/P(F_{pass})$). All these probabilities are best chance so they assume other members always refrain from scoring proposals. If other members do score proposals then all these probabilities are reduced, especially $P(F_{pass})$ versus $P(F_{ARB})$. The chance that 2 faction members (b – p + 1) are selected onto the same random ARB giving them blocking power is 0.1% ($P(F_{block})$) which is every 606th one ($1/P(F_{block})$).

Enabling the *board bias filter* means members that discuss or score proposals before they are raised will be excluded from ARB selection. So this may not be possible for small communities. Enabling the option changes quite a few of the numbers but generally means $P(F_{ARB})$, $P(F_{pass})$ *and* $P(F_{block})$ would be lower. If a proposal is very popular and the *board bias filter* is enabled then there may not be enough members available so there may be a limit where it cannot apply even if enabled. Otherwise the number of members involved with a proposal may also be limited to prevent this issue.

In certain instances the proposal process may be bypassed by paying fees and the members available for ARB selection will never be part of the same pool as the members who are paying to raise proposals. For example, if 100 architects are regulating building plans, builders will pay a fee to the architects to automatically raise an ARB. In this case the $P(F_{ARB})$ is the relevant formula and N becomes the number of architects available for ARB instead of the size of the whole community. Also $1/P(F_{ARB})$ becomes the average rate that a faction can pass a biased proposal, since $P(F_{raise})$ is irrelevant.

For simplicity the above table only includes examples of $I=1$. However, in actual instances more proposal points allow members to contribute to multiple proposals at the same time. It also gives members the ability to distribute varying influence between proposals, since within proposals discussion is prioritized by weighted voting based on the member's score.

Proof of stake networks allow ARB community members to be additionally incentivized by distributing signing control to cooperating nodes. If their influence is attached to the value of their stake, members have stronger incentives to generate quality proposals and commit to work on ARBs.

Blockchain Adhoc Randomized Boards (BARB)

The following is a proposal for a particular ARB protocol instance achieved with a peer to peer software application that manages a special kind of *blockchain* database (BARB). A distributed blockchain database guarantees the record of events are immutable, auditable and transparent. A record of every event and message generated by the protocol is available so all members can audit the process including the randomness algorithms, the protocol process and the probabilities of bias. Thereby requiring less dependence upon trust between members. The BARB software application facilitates the following features:

1) registration of members with the capacity to :
   i. create and share proposals,
   ii. score proposals,
   iii. score reputation of other members,
   iv. track proposals, reputations and board history,
   v. configure member settings;
   vi. private, anonymous messaging between members,

2) automate attaching proposals to adhoc, randomized boards of members with the capacity to :
   i. vote on the proposal either pass or reject;
   ii. submit an explanation of their vote reasoning;
   iii. performing auditing by scoring other ARB submissions;

3) track key collective metrics including :
   i. average reputation score
   ii. average proposal score
   iii. average raised proposals per week
   iv. history of proposals

In the example below members are the investors of a successful crowd fund and are each allocated proposal points in proportion to their investment, although on average each investor contributed $1000. The crowd fund is regulated by the BARB protocol so transactions need authorization signed by an ARB. The distributed nature of the protocol and blockchain database means investors can audit the processes of their collaboration. The following global parameters are configured for the community :

community size	1000 members		
spend vs net worth	board size	pass limit	pass ratio
0% to 30%	10	5	>= 50%
>30%	30	20	>= 66%
>50%	1000	700	>= 70%

initial proposal points	0.01 points x $1000 = 100
raise limit	5000 points
board bias filter	yes, members who score proposals are excluded from the ARB
member availability	all yes configured per member to indicate ARB availability

The metrics for this community are seen on line #6 of the table in the previous section. So firstly, members register and the size of the community is fixed to 1000 members, the original investors. Then proposals are created by individual members to be detailed and shared through the community. If members each start with 100 proposal points on average and the raise limit is 5000 then it would take about 500 members to raise a proposal. If a proposal can reach a score above the raise limit then it moves to draft status. It stays in this status for priority discussion until the draft period expires with options to extend the period. If the score remains greater than the raise limit when the draft period expires then the proposal is automatically raised for voting by an ARB. Since the *board bias filter* is yes therefore members who scored or discussed the proposal are excluded. This is automated by randomly selecting other available members and issuing to each of them signing control of a crypto-account associated with the proposal. For a spend >30% of the communities net worth a pass requires 20 of 30 signatures (>=66%). For a spend >50% of the net worth all members are called to vote so its just like a referendum, and in this case 70% are required to authorize the spend.

Corrupting boards for spends >30% is difficult as even with a faction of 400 out of 1000 the chance of them getting enough places on a random board of 30 requiring 20 to pass their proposal is 0.25% and even less with the *board bias filter* option enabled. Although with such a large faction 54% of proposals can be blocked or if the faction members can convince others to positively score their target proposals they might find ways to pass other biased proposals. Perhaps, they could try to influence the other board members on the fly. Their communications may happen offline so their actions might not be transparent. Generally fraud would be a lot easier without the regulation of an ARB protocol. And if the ARB is configured correctly problems can be minimized. The parameters would need testing to achieve the best configurations for each instance.

A bitcoin example

In the above example, some smaller group of people still need to be trusted to control the community funds and honor the ARB outcomes. Using bitcoin it is possible to put the funds directly into the control of many members of an ARB community, thereby minimizing fraud by distributing signing trust with an ad hoc and random protocol. So a basic bitcoin account is controlled by one person. A multi-signature account requires a fixed set of persons to sign transactions. A *BARB bitcoin account* requires a randomly changing set of members to sign transactions that are firstly raised via the proposal process of the ARB community.

Anyone could take their bitcoin and send it to a multi-signature account administered by an ARB community instance. That bitcoin account would be controlled by the ARB protocol. The community formulates proposals for specific transactions and if any proposal is scored highly enough by the community it is raised before an ARB for voting. If they pass it then the transaction is signed and broadcasted.

Firstly a bitcoin *community account* needs to be created and bitcoin which is to be controlled by the ARB community is sent here. This account needs to be a multi-signature account requiring *m* of *N* signatures, where *m* is an appropriately high number close to *N*, the size of the community to allow for some lost or temporarily unavailable signing keys. Every member of the community has a signing key but only *m* members are required to sign. In case significant signatures are reported lost then bitcoin funds could be sent to a new account with a lower *m* value.

To spend their bitcoin a community needs to raise a proposal and specify the *destination public address* and *amount* for payment. If the ARB passes the proposal then the protocol prompts each member's device to sign and broadcast a transaction from the *community account* to the *destination public address*. Alternatively the ARB could have its own multi-signature account created at the time it is formed and the bitcoin amount specified in the raised proposal could be transferred there before voting.