Measuring Trust

Exploration of Relative Trust Scores Derived From a Web of Trust

Tyler Yasaka
contact@tyleryasaka.me

ABSTRACT

...."I'm not reading a good review on Yelp."

...."That's not good, hmm? Mmm, you know what? Fuck it, 'cause this is usually just one pissed off dude who had a bad experience and then wrote, like, 50 bad reviews."

This is an excerpt from the Netflix show *Love* (season 1, episode 5). It's a trivial example of the more general problem of *Sybil attacks*, situations where a malicious actor games a system that relies on the assumption of unique identity by creating numerous fake accounts.

On systems with centralized servers, there may exist methods to at least partially deter Sybil attacks. These methods might utilize IP addresses or phone verification to limit the number of identities a person can make (or at least make additional identities expensive), for example. My knowledge of the extent of the threat to centralized systems is limited; my concern is instead focused on decentralized, permissionless systems.

The concept of an open and permissionless system is philosophically appealing. However, there are certain applications that require the concept of trusted identities. At a minimum, all systems that involve voting rely on unique, trustworthy identities to cast a vote. This includes any consensus mechanism as well as any rating system. Such systems face a dilemma: how can we filter out bad actors without a centralized authority?

Existing solutions seem to generally rely economic incentives. Bitcoin (and other systems based on proof-of-work), for example, makes influence in the network computationally expensive. Proof-of-stake, as an alternative, requires tokens to be staked in exchange for influence. In other words, these solutions are pay-to-play; Sybil attacks are no longer feasible because they are expensive.

However, I am interested in exploring an alternative to economic-based systems. There are a couple of reasons for this. The first is philosophical; pay-to-play systems inherently favor the economically privileged, and are less accessible to those with less economic resources. I believe that true decentralization should not discriminate based on economic privilege.

The second reason I am interested in another solution is practical. I believe there exists at least one class of applications where economic incentives fall short. The class of applications I have in mind is *reputation*. I use the word *reputation* in a very broad sense; this could be anything from a personal reputation to a product rating. I believe that economic solutions to reputation fall short because they lack a key piece of information: trust. I contend that reputations, in the real world, are based on trust. More specifically, they are based on a peer-to-peer network of trust.

Thus my interest in an alternative solution to the Sybil problem: the web of trust. In this paper, I propose the concept of deriving relative *trust scores* using a given *trust metric*, one score for each identity from the perspective of another, in a web of trust. I then offer as examples multiple trust metrics, propose the concept of *relative reputation*, and explore the the idea of obtaining social consensus from a web of trust using trust scores.

1 Web of Trust

In this paper, *web of trust* refers to a simple concept that can be represented by a directed graph. In the graph, nodes point to nodes that they trust. The result is a web, where one can follow the flow of trust from one node to the next. This is an alternative to a centralized trust model, which would essentially be a graph with a central node to which all other nodes are pointing. The web of trust allows trust to be established among peers without a central authority.

2 A Relative Trust Score

Google's PageRank algorithm is famously able to assign ranking scores to web pages from a directed graph, where edges represent links from one website to another. One major problem with PageRank, however, is that it is vulnerable to Sybil attacks. Indeed, after Google implemented PageRank we witnessed the phenomenon of link farms - a form of Sybil attack that takes advantage of the fact that (in the pure form of PageRank) all pages are first-class citizens with equal ability to cast "votes" in the algorithm, in the form of web links. Dummy web pages could be created cheaply to deceive the algorithm. Of course, it is now widely known that Google uses a much more sophisticated version of PageRank that is less susceptible to link farms (though the algorithm itself is kept secret).

Deriving trust scores in an open web of trust is similar to the problem of deriving web page rankings in an open internet, in many respects. Both the web of trust and the internet can be represented as a directed graph, where edges represent votes from one node in favor of another. Indeed, the defunct website Advogato used a web of trust with designated "seed nodes" in order to generate trust scores for members of the site [1]. Advogato's solution was Sybil-resistant thanks to the seed nodes, and Google's updated ranking algorithm may very well use a similar concept of "seed" websites to strengthen to protect against link farms (though I am only speculating here).

There is a problem with seed nodes, however. Seed nodes are inherently more powerful than other nodes - an inequality of power that we do not want in a decentralized system. Seed nodes seem to be necessary when these two requirements exist for a graph-based system: (1) that it must produce absolute scores for each node (2) that it must be Sybil-resistant. We want to keep the second requirement, obviously. But what if we were to relax the first?

I will demonstrate that we can have a seed-free, Sybil-resistant, decentralized system that produces *relative* trust scores, or scores that will vary based on the observer. While relative scores may not be sufficient for all applications, there are may be some problems that can be solved merely with relative trust scores.

I will define a trust score as the result of a function that receives at least 3 parameters: the web of trust itself (represented as a directed graph), the *observing node*, and the *observed node*. For the same web of trust and the same observed node, but a different observing node, it is possible to produce different trust scores. Hence they are considered relative.

The implementation of the function itself is left open for now. I will refer to a trust score generation algorithm as a *trust metric* (borrowing from Advogato's terminology). This will allow us to discuss the concept of trust scores without being married to a particular implementation of them. Next I will present some example trust metrics, though I have no doubt that there are other interesting (and probably superior) metrics in addition to those that I am presenting.

3 Example Trust Metrics

3.1 Metric Example 1

In this metric, the trust is calculated by first taking all paths in the web of trust from the observer to the observed. For each path, we subtract one from the number of edges in the path, raise two to that power, and then take the inverse of that number. The trust score is the sum of these results. More precisely:

$$T(a,b) = \sum_{p \varepsilon P(a,b)} \frac{1}{2^{p-1}}$$

where *T* is the trust score, *a* is the observer, *b* is the observed, and *P(a, b)* is the set of the lengths of all paths from *a* to *b*.

3.2 Metric Example 2

I take this trust metric from the defunct Outfoxed browser extension that used a web of trust to generate ratings for websites [2]. I present it in its original form:

$$trust(s,t) = \frac{1}{1 + (hops\ from\ s\ to\ t)}$$

3.3 Metric Example 3

Here I present a recursive trust metric:

$$T(G,a,b) = \{1\ if\ a = b;\ else \sum_{c \varepsilon C(a)} \frac{T(G-a,c,b)}{|C(a)|}\}$$

where *G* is the graph representing the web of trust, *C(a)* is the set of child nodes of *a*, |*C(a)*| is the number of child nodes of *a*, and *G-a* is the subgraph of *G* where the node *a* (and all adjacent edges) have been removed.

3.4 Metric Example 4

So far, none of the presented examples have been Sybil-resistant. (I will define Sybil-resistance in the next section.) By making a minor adjustment to example 3, we can make it Sybil-resistant:

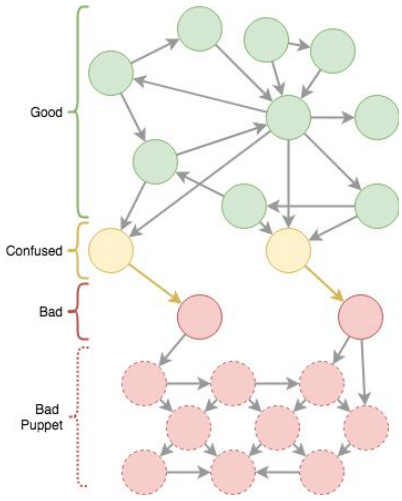
$$T(G,a,b) = \{1 \; if \; a = b; \; else \sum_{c \varepsilon C(a)} \frac{T(G-a,c,b)}{2 \times |C(a)|} \}$$

4 A Definition of Sybil-resistance

So far I have been using the term *Sybil-resistance* rather vaguely. Here I will provide a precise definition that applies to the manipulation of trust scores in a web of trust.

To do this, I will borrow the concept of *good nodes*, *confused nodes*, and *bad nodes* from Advogato. Good nodes in a web of trust are honest and will not attempt to manipulate trust scores. Confused nodes are also honest and will not attempt to manipulate trust scores, but may mistakenly trust one or more bad nodes. Bad nodes, then, are those that will create fake nodes in order to game the system in some way.

The definition of a confused node is rather open-ended, however. Does a confused node trust a single bad node, or multiple bad nodes, or an unlimited number of bad nodes? For the sake of clarity, I will define an edge from a confused node to a bad node as a *confused edge*. I will also define a *puppet node* as a fake identity in a web of trust created by an attacker for the purpose of gaming the system. More precisely: puppet nodes are the subset of bad nodes that are not adjacent to a confused edge.



I propose the following definition: a trust metric is Sybil-resistant if and only if the upper limit of the combined trust scores that can belong to bad nodes is bounded by the number of confused edges. Stated another way, a trust metric is Sybil-resistant if and only if there is a finite amount of combined trust an attacker can gain in a system merely by creating puppet nodes.

This definition alone is not sufficient when considering Sybil attacks. While it helps to know that the impact that a Sybil attack can have is bounded, we might also wish to know the *degree* of Sybil-resistance that a metric has. I will define the degree of Sybil-resistance as the maximum trust that an attacker can have with a single bad node, divided by the maximum combined trust that can be accumulated by an attacker with an unlimited number of puppet nodes. A metric where puppet nodes have no effect has a Sybil-resistance degree of one. A metric where puppet nodes can *increase* the combined trust score of an attacker has a Sybil-resistance degree *less than* one. And a metric where puppet nodes can increase the combined trust score of an attacker without bound has a Sybil-resistance degree of zero (i.e. is not Sybil-resistant).

5 A Proof of Sybil-resistance

I previously claimed that metric example 4 is Sybil-resistant. I will further claim that example 4 has a Sybil-resistance degree of ½. I will now supply a proof of these claims as an example of how this definition of Sybil-resistance can be applied.

To perform this proof, I will start by making an assumption that forbids a certain type of edge case, and then dealing with this edge case once sybil-resistance has been proven with the assumption. The assumption is that a puppet node cannot be trusted by more than one other node. This precludes the existence of multiple paths to any given node. Thus, looking at the equation, we can effectively ignore the summation as it adds the results from multiple paths to a node. For now we will assume just one path.

Let's say we have an observer node *n* and a single confused node in the observer's network, which points to an attacker node *a*. Let *t* represent the trust score of *a* with respect to *n*.

Next, I will describe the concept of a *generation*. Here I will refer to a generation *k* as a set of puppet nodes descended from the attacker where there are *k* edges in the path from the attacker to each of the descendents. The first generation will refer to all direct children of the attacker, and

the second generation will refer to all grandchildren of the attacker, etc.

With the definitions in place, we can proceed with the proof. First we will prove the base case, generation one. The first generation, of course, consists of the direct children of the attacker. If we assume a trust value of *t* for the attacker, the trust score for each child of the attacker will be $t \times \frac{1}{2 \times c}$ where *c* is the number of children of the attacker. Assuming the attacker has at least one child, the combined trust score for all children of the attacker will be $c \times (t \times \frac{1}{2 \times c}) = \frac{t}{2}$.

Now let us consider two consecutive generations, *k* and *k + 1*. Let us divide the nodes in generation *k* into groups by parent. For any of these groups, we can characterize the trust score of each member as $p \times \frac{1}{2 \times c}$ where *p* is the trust score of the parent and *c* is the number of children. Thus the combined trust score of each generation *k* group will be $c \times (p \times \frac{1}{2 \times c}) = \frac{p}{2}$. For each of these groups, each member node can have children of its own, which will belong to generation *k + 1*. Let us also group these children by parent. This leaves us with a nested group structure. Let us define the trust score for each generation *k + 1* group. For each node, the trust score is given by $q \times \frac{1}{2 \times d}$ where *q* is the trust score of the parent node in generation *k* and *d* is the number of children of that node. Substituting our equation for the trust score of a *k* node, we can represent the trust score of a *k + 1* node as $(p \times \frac{1}{2 \times c}) \times \frac{1}{2 \times d} = \frac{p}{4 \times c \times d}$. The maximum combined trust for each generation *k + 1* group is then $d \times (\frac{p}{4 \times c \times d}) = \frac{p}{4 \times c}$. Now let us consider the combined trust for generation *k + 1* groups that are nested under a given *k* group. This combined trust is given by $c \times \frac{p}{4 \times c} = \frac{p}{4}$.

Now, it is important to note that this value of *p* can vary between each generation *k* group. So we must represent the combined trust of generation *k* as a summation: $\sum_{p \varepsilon P} \frac{p}{2}$ where *P* represents the set of parents of each *k* group. Likewise, by summing the *k + 1* combined trust values for each *k* group, we can represent the combined trust of generation *k + 1* as $\sum_{p \varepsilon P} \frac{p}{4}$. Thus it is now trivial to see that the maximum combined trust of generation *k + 1* is exactly ½ that of generation *k*.

We have now demonstrated that the first generation of an attacker will have a maximum combined trust score of $\frac{t}{2}$. We have also demonstrated that the maximum trust score of any generation will be exactly half that of the preceding generation. We can thus denote the upper bound of the total combined trust of the attacker and the attacker's puppet nodes, by taking the upper limit of the sum of an infinite number of generations: $t + t \times \lim_{k \to \infty} \sum_{i=1}^{k} \frac{1}{2^i} = t + t = 2t$. This is a finite upper bound, demonstrating Sybil resistance. We can also calculate the *degree* of Sybil resistance by taking the attacker's trust and dividing it by the maximum that can be achieved through a Sybil attack: $\frac{t}{2t} = \frac{1}{2}$.

However, we have one loose end. So far we have assumed only one path from the attacker to a puppet node. As it turns out, multiple paths do not affect the results we obtained. This is because, in the original metric equation, the results of multiple paths are added together. If there are multiple paths to a node, we can equivalently represent this graph by making copies of the node (along with its descendants) and supplying one path to each copy. In the single-path graph, the scores from multiple paths will simply be added together to form the total. In the multiple-path graph, the same scores will be added together but will simply be attributed to a single node rather than being spread out over multiple nodes.

6 Use Case: Relative Reputation

Today's online systems use absolute scores to portray the reputation of identities. Commonly known examples are ratings for businesses on Yelp or upvotes on Stack Exchange answers. I refer to these scores as *absolute* because everyone sees the same scores. In contrast, *relative* scores vary based on the observer. I am not aware of any current system that uses relative reputation.

The web of trust provides an information layer that makes relative reputation possible. For example, Amazon product ratings could be weighted by the trust score of the author of each review. With absolute reputation scores, the rating of an item will simply be the average of all posted ratings. But using relative reputation and a web of trust, the rating of an item could be a weighted average of the ratings, based on the trust score of the author. Because trust scores vary from observer to observer, two people could see completely different ratings for the same product.

I have discovered one project that used a similar concept. The Outfoxed browser extension (now defunct) allowed users to enter their "informers" (people they trusted), which essentially generated a web of trust. Websites were then tagged according to how trustworthy they were based on the user's network of trust [3].

7 Use Case: Social Consensus

As mentioned previously, consensus is a process that requires some mechanism to defend against Sybil attacks. By consensus, I simply mean the process by which a group of individuals can agree upon a shared state. Current decentralized consensus mechanisms generally rely on economic incentives to defend against Sybil attacks and bad influence. When I use the word *economic* here, I mean that influence in the system can usually be purchased with money, either directly or indirectly.

However, I am not entirely convinced that economic incentives are ideal for decentralization. In fact, I believe that true consensus is social and based on shared trust, even when the consensus mechanism itself is economic. I agree with Vitalik Buterin's statement [4]:

> *....social considerations are what ultimately protect any blockchain in the long term.*

However, he went on to say:

> *....blockchain protected by social consensus alone would be far too inefficient and slow, and too easy for disagreements to continue without end.*

Perhaps this is true. Nonetheless, I would like to explore the possibility of achieving consensus through a web of trust.

Personally, I find Buterin's theoretical social defense mechanism against 51% attacks in proof-of-stake valid but also awkward. He states:

> *....Theoretically, a majority collusion of validators may take over a proof of stake chain, and start acting maliciously. However… if they try to prevent new validators from joining, or execute 51% attacks, then the community can simply coordinate a hard fork and delete the offending validators' deposits.*

This is a social consensus defense mechanism, but it is an awkward one. So, to defend a proof-of-stake protocol against a 51% attack, the solution is to fork the protocol itself? I understand that such an attack may be unlikely. Yet there is a part of me that seeks a more elegant solution. I also believe that it is hard to estimate the possibility of such attacks, and even harder to predict how a community would react to them. I like the spirit of Buterin's solution, but I wonder if we cannot do better.

I propose the following as food for thought: why can this forking process not be part of a protocol itself? More fundamentally, what if we had a protocol for social consensus? Perhaps this protocol would be "inefficient and slow"; perhaps more efficient layers would need to be built on top. Still, I think there may be significant value in a social consensus protocol. I also conjecture that if such a protocol were to exist, it would be based on a web of trust.

I will propose a naive concept as an example. Suppose some sort of efficient data sharing layer were to exist, one on which trust data between nodes could be easily stored and accessed. In this scenario, we could define the concept of a *community* as a set of nodes which mutually trust one another. That is, each node in the community would need to have a minimum trust of *t* as observed by all other nodes in the community. Each node in the community could be considered to have one vote in establishing a shared state for the community. Thus the community could achieve consensus.

I have not explored this proposal in depth. I can think of numerous concerns already. How would the data be stored and accessed reliably and efficiently? What trust metric and minimum trust score could be used to maximize inclusivity but deter malicious influence? I do not have answers to these questions at the moment. I am simply offering this proposal to stimulate thought and conversation around the idea of social consensus built on a web of trust.

REFERENCES

[1] *Advogato's Trust Metric*, https://web.archive.org/web/20170627230829/http://www.advogato.org/trust-metric.html.

[2] S. James, *Calculating Levels of Trust*, http://getoutfoxed.com/node/112

[3] S. James, *What is Outfoxed?*, http://getoutfoxed.com/about

[4] V. Buterin, *A Proof of Stake Design Philosophy*, https://medium.com/@VitalikButerin/a-proof-of-stake-design-philosophy-506585978d51