Generalized Impact Evaluators

Network Goods[1]
network-goods@protocol.ai

Abstract

Existing funding systems fail to sufficiently fund public goods and common goods due to insufficient mechanisms for coordinating various agents towards valuable outcomes. Relative to traditional capital systems that scalably organize activity around maximizing financial performance, impact funding remains underdeveloped, especially in the ability to reward high-upside, high-uncertainty work. Here, we propose Impact Evaluators (IEs) as a modular system for coordinating work by measuring, evaluating, and retrospectively rewarding the impact achieved towards specified valuable objectives. We present a structure to define Impact Evaluators as well as design schematics to facilitate their implementation. We then discuss implementation considerations, practical learnings from past experiments, and integration with the broader ecosystem of public goods and commons funding systems.

[1] Network Goods is an independent team within Protocol Labs

Content

1. Introduction

Purpose of this document

The goal of this document is to accelerate development of Impact Evaluator mechanisms by providing:

- Standard language to define and construct Impact Evaluators
- A framework to efficiently implement Impact Evaluators

What is an Impact Evaluator

An Impact Evaluator (IE) is a system or mechanism that coordinates groups of agents to work towards objectives by assessing impact against those objectives to retrospectively reward valuable work or outcomes (Benet, 2022a). An effective IE creates an incentive structure that guides potential contributors to cooperatively pursue the specific objectives.

Examples of this structure can be seen in block rewards, performance reviews, and performance-based contracts. We will explore several examples below to illustrate.

Why Impact Evaluators are important

Impact Evaluators can provide incentive or market mechanisms for highly effective "impact funding" (i.e. provision of public goods) that manifests in two major needs:

<u>Evaluating value creation:</u> At its simplest, an Impact Evaluator is an important function to indicate what *was* valuable and distribute a suitable reward (noting "valuable" is subjective). In traditional market scenarios this could be used to indicate an "ROI" on deployed resources, which is often the primary key performance indicator (KPI) to determine value and define future allocations of capital or labor. This feedback signal is underdeveloped in impact markets and Impact Evaluators address this.

<u>Reward high-potential/high-uncertainty work:</u> Existing funding systems, particularly in public goods, typically prospectively fund projects that have clear

paths to outcomes. This funding is also typically based on present and future cash flows. High uncertainty projects with high upside, particularly those without easily projected cash flows, appear to have a gap in funding systems. This manifests as missing incentives for the best talent to pursue these projects.

While we explore Impact Evaluators for “impact funding,” this mechanism can be very powerful for incentivizing uncertain tasks in general where typical financial methods fall short.

2. Designing an Impact Evaluator

To build intuition around Impact Evaluators, we will use an illustrative example throughout Section 2 in addition to the core theory. Note to the reader that our model systems used to explore IEs will draw heavily from open source software and blockchain experience, but that does not limit broader applicability.

Illustrative example: The WidgetMakers community

Suppose there exists a group of individuals that creates widgets, The WidgetMakers. These widgets are a classic “public good” - they are not sold, but many people in the broader community use them and find them valuable. The WidgetMakers have received a generous donation in the form of a $2M USD grant, and now want to grow the impact of their community, which they want to accomplish by bringing in new members. They allocate 50% through prospective funding over 10 years (e.g. referral fee, onboarding grants) but these options do not align incentives of new members with the success of the WidgetMaker community - this can lead to bad actors that will take funding but not perform as valuable members of the community.

To counteract this, WidgetMakers announces it will reward the remaining 50% to “valuable members” with a bonus based on the value of their contributions each year, starting with the previous year. This incentivizes existing members to consistently contribute and more members to join the community. Setting up this mechanism will require the ecosystem to:

- Define their **Objective** → intermediate goal of “more valuable members” in their community

- Define the **Scope** → which community members are eligible and the eligible indicators (actions / outcomes) counting toward the objective
- Define an **Interval** to evaluate the value created (in this case the last year)
- **Measure** the indicators (actions / outcomes) during the interval that provide the most information on value created, and identify entities (members) that contributed
- **Evaluate** the indicators measured and attribute to the entities (members) responsible
- Disburse an appropriate **Reward** to incentivize more valuable outcomes
- Apply this measure → evaluate → reward process repeatedly for future intervals

We will now develop each step in detail as we arrive at a formal definition of Impact Evaluators. More examples of IE design in practice are found in section 3.

Impact Evaluators: The Formal Definition

An Impact evaluator can be generally described by the tuple of objects:

$$Impact\ Evaluator\ IE\ =\ \{r,\ e,\ m,\ S\}$$

where

r is the reward function,
e is the evaluation function,
m is the measurement function on S, and
S is the scope of the IE.

The scope, S, constrains the domain of all past actions, outcomes, and entities to the subset that will be considered for the specific IE. The ordered application of these functions is used to compute the reward allocated to each entity. Additionally, all activities can be repeated in additional epochs or intervals, where

i is the interval (time or other discrete milestone measuring the IE cadence).

This (simplified) ordered application yields the reward disbursement

$R = r(e(m(S)))$

for which the IE is engineered to incentivize a set of desired objectives (*O)* within a scope *S*. The goal of an impact evaluator is that, over time, the value of objectives (*O*) achieved are greater than the value of resource inputs (operational cost + incentive rewards).

The remainder of Section 2 will expand on the individual components of an IE, concluding with a restatement of the reward disbursement that elucidates relative allocation between entities rewarded by IE.

Scope (*S*)

We start with the set of actions taken by entities (individuals or organizations) over all time and the outcomes of those actions. Within this set, we use Scope *S* to identify a subset that is intended to be measured, evaluated, and incentivized by the impact evaluator.

This subset could be be constrained by any boolean constraints on this set, including but not limited to: temporal constraints on the action or outcome (e.g. specifying hours or months), type of action (e.g. specifying a project name or type of activity), location (e.g. the amazon rainforest or github.com/ipfs/kubo), or any other designation.

To convey our intuition, we suggest but do not intend to formalize or prove the following

- This set could be considered a space by adding the additional metric where two actions are considered "close" if they have overlapping impact (or at least logically correlate with the same or similar, likely counterfactual, outcomes).
- We expect that all practical scopes will be convex subsets of this space.

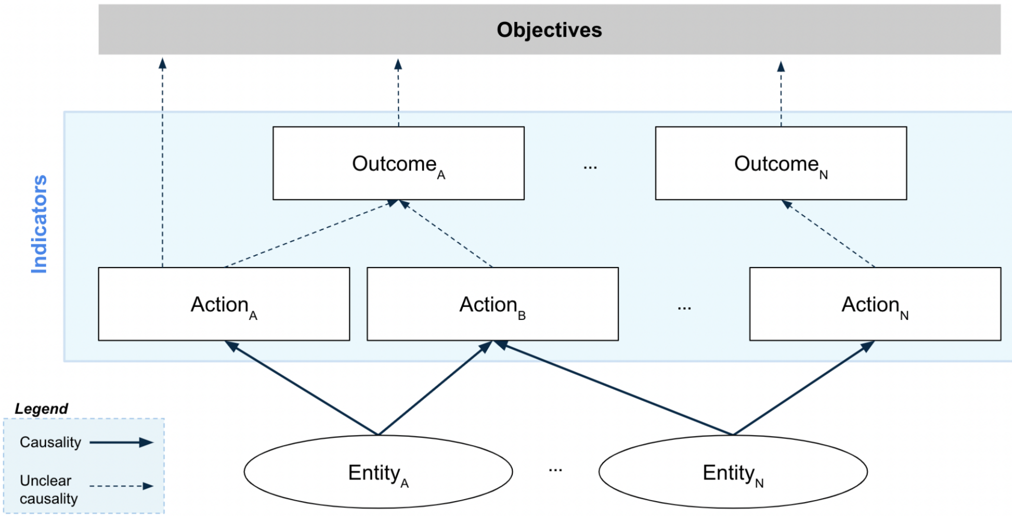

Fig 1: *Framework for linking entities, indicators (actions & outcomes) to specified objectives within Scope S. Realistically causality is not clear in complex systems, so a subset of the most informative indicators is selected.*

We choose to use the language *actions* and *outcomes* to convey that an IE can be designed to incentivize both actions (as a leading indicator of impact) and outcomes (as a signal or lagging indicator of impact). This is due to the fact that it can be intractable to directly measure objectives and establish causality with entities to properly design incentives. We therefore group *actions* and *outcomes* as *indicators* of the objectives we desire to achieve with an IE, which will be measured and attributed to entities.

An impact evaluator interacts with a specific scope S at various intervals i which can be tied to either time or milestones to incentivize the evolution of desired objectives (O), which logically must be a set within the expected outcomes and actions within scope S. The scope of the overall IE, S, is thus the union of subscopes, S_i, expressed

$$S = \bigcup_i S_i$$

As a fairness constraint, we also expect IEs to aim to minimize double counting of actions within subscopes, such that:

$\bigcap_i S_i = \{\}$.

S can be defined broadly but should be limited to a specific set of entities, actions and outcomes to limit complexity. The homogeneity of activities and outcomes being evaluated will lead to simpler valuation, as the measurement, evaluation and reward functions may be hard to define fairly or consistently over broadly varying parameter spaces.

Practically speaking: Desired objectives O can (and should be) multivariate to capture various aspects of what "good objectives" can be. Diversifying the definition of valuable action increases resilience against bad actors gaming the system a la Goodhart's law[2] (e.g. ecosystem growth can be measured by # projects, satisfaction of builders, # projects raising money).

WidgetMakers example: In this example the scope is "Widgetmaker community", which can be structured broadly as:

- Entities: community members
- Indicators: contributions to the growth of the WidgetMaker community and the number of widgets it creates
- Interval: yearly

Measurement function (m)

The measurement function takes a scope, S, as input, and outputs a set of ordered pairs describing indicators within the scope and the entities that contributed to those indicators. Namely,

$$m(S_i) = \{(indicator_1, entity_1), (indicator_2, entity_2), \ldots\}$$
$$where\ indicator_i\ can\ be\ attributable\ to\ one\ or\ more\ entities$$

The indicators for each interval should be the direct result of activities/outcomes within the scope of that interval (potentially building off of activities within

[2] Goodhart's law is often stated as "when a measure becomes a target, it ceases to be a good measure"

previous intervals as well). These features can include qualitative measurements (e.g. value of community nominated projects), quantitative measurements (e.g. number of active IPFS nodes), or both.

Practically speaking: This step is highly non-trivial, requiring systems to objectively capture the state of S_i (entities, indicators contributing value during interval *i*). It's hard to perfectly specify a single comprehensive indicator of impact, so a combination of intermediate activities / metrics / outcomes are typically required. We cannot capture all aspects of a system, but we can design more constrained systems (e.g. blockchains) that limit the set of inputs / outputs that dictate a system's activity. In more open systems a subset of the most informative indicators can be selected

Widgetmakers example: We have taken "entities" to be valuable community members. We want to reward members who 1) create new widgets (outcome) 2) support the community in getting better at making widgets (action) 3) spread the word about the widgets (action). To measure this we choose a representative subset of possible measurements:

Objective	Indicator
Create new widgets	# widgets created
Support community	# technical reviews
Spread word about widgets	# posts with >50 reactions

Table 1: *Mapping Widgetmaker IE objective to Indicator*

To measure these contributions the ecosystem builds a system to pull these measurements and attribute them to the entity who created them, so we arrive at:

$$m\ (WidgetMakers\ last\ year)\ =\ \{(3\ widgets\ created,\ member_1),\ ...\ (9\ reviews,\ member_{10})\}$$

Evaluation function (*e*)

The evaluation function combines the outputs of the measurement function and one or more evaluators to ascribe some impact to the various outputs. This converts the set of outputs into a measure of value and attributes it to the entities that created it.

$$e(\{ (indicator,\ entity)_j \}, \{evaluator\}) \ = \ \{(entity,\ evalscore)_j\}$$

The output is a vector of entities and their respective evaluation score(s).

The evaluators can be individual humans, organizations, or objective functions. Naturally, objective functions can only operate on quantitative indicators, while humans and organizations can operate on either quantitative or qualitative indicators. Each evaluator should be expected to have their own value metrics; the value metrics of humans are implicit (and the choice of evaluator is a choice of metric by proxy) while the value metric of functions are explicitly coded by system architects. In the case of more than one evaluator, a mechanism for aggregating multiple value metrics is needed.

Practically speaking: We need to convert measurements from our scoped system S generated by m into a set of value metrics, and attribute the value generated by the contributing entities. This evaluation function can be a combination of programmatic and human-in-the-loop methods to maximize accuracy:

- Programmatic (quantitative) functions examples include:
  - Weighted product of input measurements
  - Contribution graphs
- Human-in-the-loop (expert) functions examples include:
  - Quadratic Voting
  - Manual sizing
  - Quadratic Funding

WidgetMakers example: We choose to convert measured indicators per entity into a usable single score by setting a weighted product of indicators (an objective evaluator function). In this case we presume attribution is possible via the measurement system (i.e. we can measure in a way that we can link causally to an entity) which is then verified by the evaluation function. We introduce initial human judgment in setting the weights that dictate the impact evaluation score:

$$e \ = \ w_1 \ * \ (\#\,widgets\ created) \ + \ w_2 \ * \ (\#\,technical\ reviews) \\ + \ \ w_3 \ * \ (\#\,posts\ with \ > 50\ reactions)$$

$where\ \{w_1, w_2, w_3\} = \{10, 7, 5\}$ → set based on expert input

Reward function (r)

Once we have a completed evaluation, indicating a magnitude of value generated by each entity, we can convert the evaluation into a suitable reward in our reserve resource using a reward function.

$$Rewards\ R_{ij} = r_i(\{(entity,\ evalscore)_j\},\ reserve\ reward) = \{(entity\ ,\ reward)_j\}$$

Where R_{ij} is the reward amount allocated to the jth entity during the ith interval, generated by applying a reward function on evaluations. The reward function r_i can vary in different intervals (expanded below).

Reward functions can be designed to create different types of dynamics among the entities participating in each epoch, but ideally should be no worse than zero sum:

A) Negative sum

- If entity A gets 1 unit more rewards, the sum of the reward to all other entities decreases by more than 1
- Usage of this form of reward function should actively disincentivize participation, as it encourages intense competition

B) Zero-sum

- Increasing reward to entity by 1 takes equal reward away from other entities
- E.g. a fixed reward pool that is split among participants

C) Positive-sum

- Individual entities achieving incremental objectives are proportionally rewarded, independent of rewards to other entities
- E.g. reward pool X is split by entities {A,B}; if an entity A creates additional value Y then the total reward pool increases proportionally to X+(~Y), and A is distributed the full ~Y

- Assumes a possible (win, not lose) scenario with positive-sum game
- Does not proportionally reward systems with network effects

<u>D) Superlinear positive-sum</u>

- Achieving objectives cooperating with other entities increases the total rewards more than achieving objectives individually
- E.g. reward pool X is split by entities {A,B}; if A and B collaborate with C to create additional value Y, all >(~Y) rewards are evenly distributed as reward pool increases
- Incentivizes cooperation and coordination among entities, proportionally rewards systems with network effects

<u>Reward assets</u>

To ensure rewards are available for each interval, a reserve of assets needs to be available and managed for the duration of the IE. These assets can take many forms, each with their own benefits / drawbacks:

- Project Token (e.g. in blockchain case)
  - Introduces market mechanics to the valuation / reward dynamic
  - High regulatory and setup cost
  - In simple case, removes need for treasury management (depending on token release dynamics)
- Treasury (DAO, foundation, ecosystem, etc…)
  - Evaluation of value must be linked to an external reward asset
  - Requires holding and maintaining a pool of assets to support rewards
- Social (signaling, recognition)
  - Can be itemized (POAP, NFT) or social recognition (e.g. community announcement)
  - May require scarcity to be seen as particularly valuable

<u>Reward Function over multiple IE Rounds</u>

It's important to distinguish between the reward function of a single instance of an impact evaluator instance r_i vs. repeated application of Impact Evaluator instances $r = \{r_i, \dots, r_N\}$. Designing and communicating the periodic nature of future rewards is needed to drive Impact Evaluator effectiveness - the expectation of a reward for achieving an objective incentivizes speculative work towards that objective. An Impact Evaluator can also run for a single round (e.g. as part of one-off retrofunding) but will not benefit from the incentives of repeated application.

In its simplest form, the reward function per epoch is $r_i = r$ i.e. the same reward function is applied each round. Reward functions over multiple rounds have a few main design choices:

- E.g. Interval (epoch) of reward release
  - Time-based (e.g. every 6 months)
  - Milestone-based (e.g. when ecosystem KPI reaches next target)
- E.g. Reward function per epoch
  - Consistent per interval $r_i = r_{i+1} = r$
  - Scale with value created each epoch $r_i \sim \sum_j \{(evalscore_j)\}_i$
  - Scale based on completion of milestones on S

Widgetmakers example: We will define a simple case using a fixed pool of assets per interval (zero sum game)

- Split a prize pool of $100k every year based on normalized evaluation score per entity
- Reward asset pool is the $1M USD allocation, which is managed to last at least 10 years

Summary

In summary, the overall flow of an IE is as follows:

$m(S_i) = \{(indicator, entity)_j\}$

$e(\{(indicator, entity)\}, \{evaluator\}) = \{(entity_i, evalscore_i)\}$

$Reward\ R_i = r_i(\{(entity_i, evalscore_i)\}, reserve\ reward) = \{(entity_i, reward_i)\}$

Which can be summarized

$\{(entity\ , reward)\}_i = r(\,e(\,m(S_i), \{evaluator\}), reserve\ reward)$

At any point, any quantity may be summed over i to determine the total quantity across all intervals and summed over j to determine the total across all participating entities

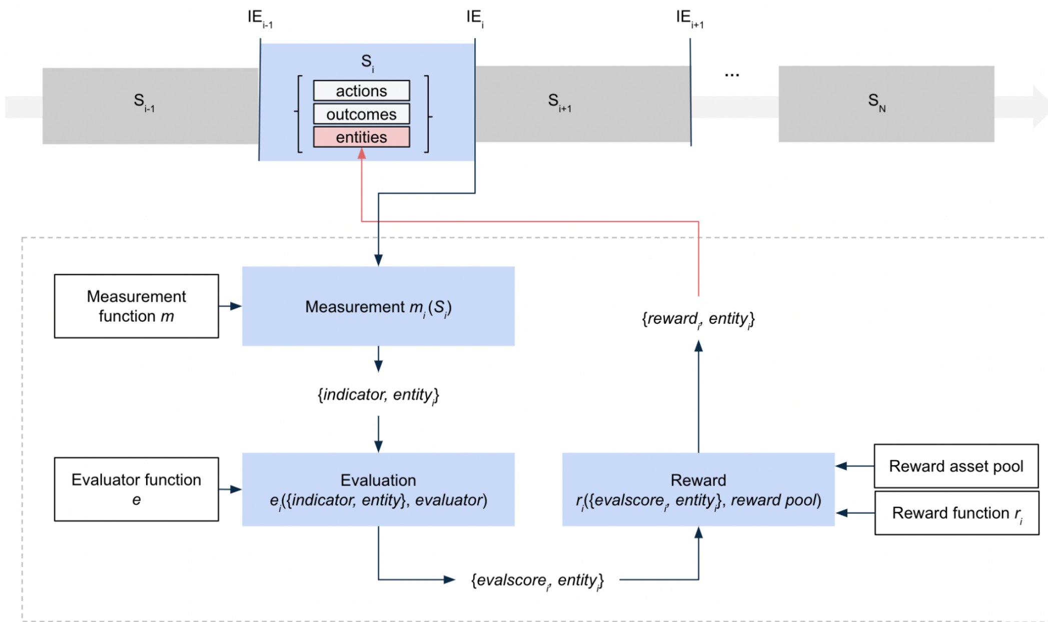

Fig 2: *Impact Evaluator component flow: functions (white boxes) combine with data input/outputs (brackets {}) to run computations (blue box), resulting in distribution of rewards to entities*

3. Impact Evaluators in practice

In this section we translate theory to practice by observing emerging paradigms for IEs, walking through considerations for operationalizing an IE, and discussing case studies which have informed our early perspective. So far our perspective has been shaped by scenarios where institutions or ecosystems have desired to 1) deploy capital in an effective manner or 2) create a scalable incentive mechanism for growth and maintainership, which will likely introduce a bias to our "practical considerations" below.

Emerging paradigms

Through our experiments, we've seen a few patterns forming with IE applications. We will introduce here then explore a few key considerations and examples below:

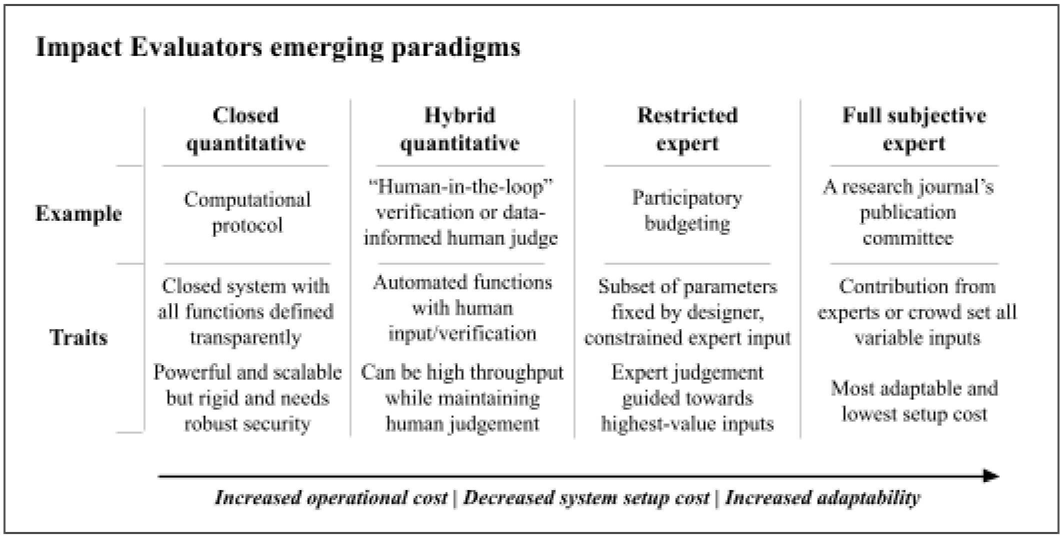

Figure 3: *Identifies four emerging paradigms within IEs, placing them on a continuum comprised of operational costs, system setup costs, and design flexibility.*

General operational considerations

IEs can be constructed in a variety of ways, enabling different degrees of freedom within the system to match the solution required.

Human (subjective) vs. automation (quantitative) tradeoffs

IE designs can range from computational protocols to voting mechanisms leveraging subject matter experts, depending on the solution desired. This comes with tradeoffs between:

1. *Up-front setup cost* to design and deploy the IE
2. O*ngoing operational cost* of experts and other human administration
3. *Adaptability* of an IE to adjust to unforeseen circumstances and effects

Quantitative IEs have high scalability by automating constituent functions, trading off high setup cost (design and tooling) and limited flexibility. Building an objective measurement function (*m*) is difficult and typically only tractable where robust measurement infrastructure already exists.

- Robust quantification can require reduction of *S* to ensure the integrity of data inputs
- With no administrative oversight, an IE becomes more vulnerable to gaming and unanticipated higher-order effects
- This function should be designed with resilience to bad actors and accuracy in representing desired outcomes, as opposed to choosing easily measurable metrics, as easily measurable metrics are subject to being gamed

Subjective IEs can accommodate more complex *S*, as expert evaluation has high adaptability. This results in lower startup costs but higher ongoing operational costs (expert in the loop).

- Expert bandwidth is the biggest constraint, as experts are typically highly sought after
- Efficient input mechanisms (e.g. quadratic voting on condensed project data) should be used to leverage limited expert time at the potential increase of signal / noise in inputs

Overhead costs of IEs are expensive (high fixed cost for system development and administration from experts), so a sufficient amount of capital should be deployed or number of rounds administered in case of quantitative IEs to offset fixed cost.

Fixed vs. evolving structure

IEs work on the entities' expectation that valuable work (as defined by scope *S*) towards objectives *O* will be rewarded in the future. Entities' level of trust in an IE will drive behavior, creating pressure for clear signal on reward distributions. A practical consideration of this problem is to ensure messaging for the rounds of recurring Impact Evaluators aims to build trust and legitimacy in the system's perception. This manifests in a tradeoff between fixed, transparent IEs vs. a need for iteration and flexibility for the IE designer.

A transparent IE (public *m*, *e*, *r* functions) is the most powerful in that it gives highest certainty for entities to work against the projected reward. However it requires a carefully crafted set of functions that cannot be changed, which opens up the risk of gaming and unprojected higher-order effects.

- Suggest deploying transparent IEs in more closed, mature systems with trusted input measurements and known value of target objectives to avoid miscalibrated rewards
- Transparency and a fixed structure introduces a "cost of forgery", where a bad actor can calculate the economic cost to undermine the system (e.g. a 51% attack in the case of Bitcoin)

An alternative approach when there is a higher uncertainty environment is to begin with a more flexible system and calibrate *m, e, r* over time. For example, in areas lacking metrics / measurements to support quantitative evaluations, lean more heavily on expert assessment while tracking the underlying quantitative factors that tie to these assessments. Over time, statistical methods can be used to link expert "tagging" (used as a source of truth) to the underlying feature set and gradually mix in a quantitative set of weights to the evaluation function while maintaining objective oversight. Suggest:

- Keep reward function *r* stable and adjusting *m, e* as calibration increases based on observed behavior
- Dictate clear reward profile AND levels of certainty (e.g. we will target $100k reward pool and calibrate +/- 20% based on XYZ)

Finally, IEs are noisy and can take time to achieve desired effects so IE effectiveness should be viewed over multiple rounds. Expect recurring IEs to over incentivize early participants as confidence is grown, but lead to better calibration as confidence stabilizes.

Reward function design

We will start by recalling the major needs that IEs address from Section 1:

- Evaluate value creation
- Reward high-potential/high-uncertainty work

To remain a viable long-term system, the *resources required* to incentivize desired objectives should be less than the evaluated *value of those objectives*. It is important to identify if the IE objectives are a growth scenario (e.g. each successive round or increase in core KPI(s) creates more value relative to last), or a stable scenario (maintaining existing value but not increasing). We won't explore negative-sum incentives, but they can be used in the case of extreme competition (e.g. weeding out poor participants)

In all cases the designer should estimate the average required incentive per entity expected to drive behavior, which can be up to the estimated value generated (to create a regenerative ROI-positive IE). Incentives do not only need to be monetary; the value of "signaling" is important in itself (e.g. scoring valuable contributions, community recognition). "Average incentive" is a useful simplification - the expected value of a reward also requires estimating the uncertainty of any one entity achieving impact towards the IE objectives. For example, in many cases a power law dynamic exists in value created per entity, and a reward should be structured against this (e.g. by having only the very top contributions rewarded heavily). Practically, it's good to have a sense check on lower / upper bounds of potential rewards per entity to ensure that the proposed IE is tractable:

$$min.\ required\ incentive\ <\ est.\ reward\ per\ entity\ <\ value\ of\ objective\ O$$

<u>Lower bound:</u> minimum est. reward for entities to work towards objective *O*

- E.g. per period, will take a developer an estimated 10 hrs @ $80/hr = $800 to contribute to objective that grants reward, total reward must be >$800

<u>Upper bound:</u> est. value of completed objective *O*

- E.g. bringing 100 new projects online is estimated value of $1M, so upper end of incentive is $10k / project
- This is hard to project as it implies a discounted value of all effects over time, but good as a sense check

Sizing the reward function then naturally comes out of the estimated incentives needed to achieve desired objectives *O*, but will be bounded by the availability of a sufficient reward pool. The *funding source* of the IE reward pool drives constraints on the reward function - where do the resources come from to sufficiently reward and drive incentives? As of writing this paper, the primary sources of IE reward pools have been 1) institutional funding commitment 2) a blockchain allocating a portion of future resource (token) releases or 3) a DAO or other community treasury.

When the reserve pool asset(s) value is fixed and uncorrelated to the success of the IE (e.g. fiat) then the reward function design becomes a budgeting and forecasting exercise. The simplest reward function in this scenario is a fixed-sum amount over a known number of rounds. A consistent reward round over round generates competitive dynamics for the fixed reward pool, which can discourage new participants. If the reward is perceived as attractive enough it can still generate superlinear output (e.g. prize competitions like X-prize).

"Positive sum" dynamics create an incentive for growth (new participants creating value increases the reward pool vs. competing for a fixed reward pool) and can be designed to encourage existing participants to bring in new participants. With a fixed budget the reward can be difficult to forecast (hard to project exponential trends) but some mildly superlinear dynamics can be useful if a fixed asset pool is large enough to accommodate a given number of rounds and projected growth. One practical case is when an increasing number of teams generate active work towards an objective, the prize per team is calibrated to increase:

- E.g. increasing a prize pool per round based on milestones of "valuable participants" → $100k if 10 teams ($1k / team), $120k if 15 teams ($800 / team)
- E.g. superlinear growth: $100k if 10 active participants ($1k / team), $225k if 15 active teams ($1.5k / team) → incentivizes existing teams to recruit new teams

In an ideal world, an IE generates increasing impact that causes value accretion in the reward pool, which would naturally increase the reward distributed by the IE. This implies the need for either a conditional future increase of the reward pool by a funder upon completion of objectives or a correlated value-accretive asset (e.g. a token) reward pool.

Finally, *loss factor* is a reality - a certain portion of IE rewards are likely to be given to inefficient or bad actors when a mechanism is in early stages and should be accounted for in the overall value consideration. Our rule of thumb is <10% bad behavior is reasonable

Case studies

Case study A: Bitcoin as a closed quantitative Impact Evaluator

Context: The block reward dynamics of the Bitcoin network (Satoshi, 2008) can be framed as an IE (Benet, 2022b). Fundamentally, the network rewards miners for their computational contribution to the creation of new blocks through the proof-of-work consensus protocol. Mapping this explicitly to IE functions:

Scope *S*

- The closed system of the Bitcoin network, which aims to create a "purely peer-to-peer version of electronic cash [that] would allow online payment to be sent directly from one party to another without going through a financial institution." Time frame is every 10 mins

Measurement *m*

- The hash rate contributed per miner per block time

Evaluation *e*

- The miner's work is computationally assessed by calculating the probability of contribution to the mining of a new block based on the miner's hash rate

Reward *r*

- At every block time, the block reward is split proportionally according to the token bonding curve. Token bonding curve is a discretized exponentially decreasing emission. Reward pool is zero-sum at first glance, but positive sum in that the value of each reward (BTC) increases with additional participants in the ecosystem

Features

- Low operational cost
- Low adaptability
- High system setup cost

This expectation is embedded that miners are proportionally rewarded for computational contribution. The functions (*m, e, r*) are transparent, so participants can make reliable predictions about their reward schedules when coupled with the bitcoin price. This de-risking has allowed other funding mechanisms to form around this core mechanism (e.g. BTC loans, financing for mining equipment).

Key takeaways:

- Fixed reward curve with high clarity and evaluation criteria can create powerful incentives, and form secondary funding mechanisms
- Restricting systems to simplified inputs / outputs is key for quantitative-based evaluation systems
- Reward assets in value-accreting tokens can create powerful alignment and possibility for exponential upside by contributing entities

Case Study B: Expert-based Impact Evaluators (RetroPGF on IPFS)

Context: This experiment was run in 2022 by Protocol Labs based on RetroPGF (Optimism, 2021), to reward high impact projects that contributed to the adoption of IPFS over the previous 12 months. Various restrictions emerged during the design process:

- Limited startup time (<2 months to launch)
- Limited existing / robust measurements for IPFS at an entity-level granularity
- Highly context-dependent and broadly scoped evaluation area

For flexibility and rapid deployment, a quadratic voting based expert system was deployed to facilitate retroactive funding with the following specs:

- Voting takes place through a proprietary tool with quadratic voting UI and a pre-filtered list of projects to allocate votes against
- Target 10-20 KYC'ed expert evaluators (50/50 internal & external to PL)
- Source contributions from expert + community recommendations and pre-filter for quality
- Experts vote on all projects (must recuse if conflicts) in a blinded fashion (no visibility into other expert votes)
- Fixed reward pool (zero-sum) for ease of operations / budget

This solutions features:

- High operational cost

- High adaptability
- Low setup cost

Structure

Scope *S*

- Actions → projects contributing to the growth of IPFS Implementations
- Entities → individual contributors and orgs responsible for the projects
- Interval → 12 month historical time period

Measurement *m*

- {Actions, Entities} Sourced via community generated list of projects (twitter, google forms)
- Curation step by a SME to trim to ~30 high-quality projects

Evaluation *e*

- Expert quadratic voting via platform requiring voter verification
- Group of 15 KYC'ed evaluators (6 internal experts, 9 community members)
- Experts chosen from select group to reduce sybil efforts

Reward r_i

- Fixed pool of FIL (pegged to $100k USD at contest start)
- Pool allocated according to normalized entity scores from QV Evaluation
- Recurring evaluation rounds every 6 months with same scope, but moving to more quantitative measurement and evaluation function

Key takeaways:

- Target no more than 1hr of input per Expert or risk drop-off in completion
- 15-25 experts and 20-40 projects are good rules of thumb when all experts vote on all projects
- Selecting Experts introduces bias, target groups of diverse high-context individuals (in this case an even split of internal and external)
- The value of "signaling" is much higher than we estimated
- Outcome primarily to experiment with Impact Evaluation and create signal for important projects on IPFS

Case Study C: Hybrid Quantitative IE (DocsDocs challenge)

Context: This was an early, time-constrained experiment with the idea of creating a KPI-based, automated IE in response to early experiments with purely subjective IEs. The objective was to create high quality documentation for projects within the Protocol Labs ecosystem. This was to be accomplished by rewarding the creation of docs and a decentralized evaluation of effectiveness.

Design:

- Impact Evaluator was split into 1) a core docs challenge that rewarded upon verified completion of documentation (volume) and 2) a score based on user feedback (quality)
- Teams were required to submit proof of completion of a list of documentation, which required a low-touch review from experts
- A survey tool was created to distribute a customized survey link and embed in team pages. This tool attributed survey feedback to a team and distributed rewards to teams

This solution was designed to feature:

- Low operational cost
- Med adaptability
- Med system setup cost

Structure

Scope *S*

- Actions → Launch of a set of new documentation
- Entities → Projects building within the PL Ecosystem
- Interval → 3 months

Measurement *m*

- {Actions, Entities}
- Sign-up form for DocsDocs challenge
- NPS tool widget installed in documentation

Evaluation *e*

- NPS score of documentation users
- Submission of links to verified documentation

Reward r_i

- Positive-sum dynamics based on number of participants and average quality score

Core docs challenge

# teams	Prize Pool	Cumulative reward	Reward / team
10	$10k + 0.5K FIL	-	$1k + 0.05k FIL
15	$10k + 0.5K FIL	$20k + 1k FIL	$1.3k + 0.067k FIL
20	$10k + 0.5K FIL	$30k + 1.5k FIL	$1.5k + 0.075k FIL
25	$10k + 0.5K FIL	$40k + 2k FIL	$1.6k + 0.08k FIL

User feedback challenge

Team NPS Score	Prize / Team
NPS > 50	$1500 + 60 FIL
NPS > 20	$1000 + 40 FIL

*Teams only qualified if >100 user feedback submissions

Key learnings:

- Setup cost to create tooling for open systems is difficult and high noise
- Where possible, work inputs should be “computably verifiable”
- Ensure incentives are aligned for all expert participants (in this case, rewards for user feedback on documentation via survey)
- Transaction costs for funding limit some IE’s from rewarding long-tail smaller contributions (rewards <$50 in this case were not worth processing)
- Reward calibration is criticalAssumption of “market dynamics” of a liquid pool of voters is a failure mode

4. Interoperability with other funding mechanisms

IEs are a powerful generalizable funding mechanism but not a "one-size fits all" approach to funding public goods and high uncertainty projects. We have seen IEs most effective when deployed with other incentives, explicitly or in a manner / scale such that other mechanisms can form around them (e.g. secondary lending markets for bitcoin miners). A full definition of impact markets and interoperable funding systems is beyond the scope of this paper, but IEs are an important component and we will touch on a few high value examples.

Prospective funding programs and cash flow

IEs retrospectively reward outcomes to align incentives for high-impact/high-uncertainty work, however operating cash flow is still needed to achieve these objectives (e.g. headcount, equipment) and is not provided by an IE. This is particularly relevant during early IE rounds when rewards have not been realized or distributed.

We have seen that IEs are most effective when combined with prospective funding mechanisms like grants to address the gap in cash flow and more generally, support funding at all development stages / risk profiles. As IEs repeat over time at scale, additional prospective funding mechanisms can form (e.g. funding miners of a blockchain against projected token rewards).

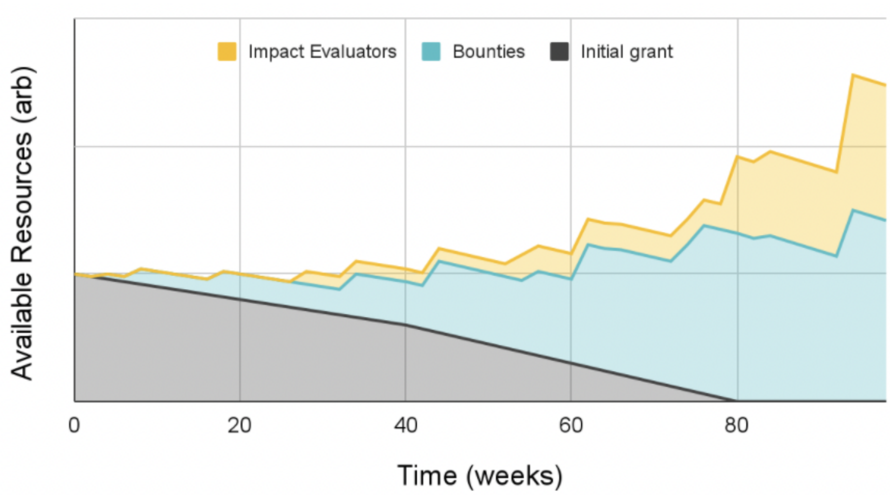

Figure 4: *Target project funding via multiple composable mechanisms over time*

Figure 4 illustrates how over the lifetime of a project the reliance on initial grants – that were necessary to start the project – decreases and retrospective funding mechanisms, such as bounties and Impact Evaluators create a sustainable financing model for the project (Miyazono, 2021).

Composability with other Impact Evaluators

A single impact evaluator is unlikely to generate the optimal incentives over a sufficiently large and complex scope. However impact evaluators can be composable in that one IE can pass $\{e, r\}$ functions as input to the measurement function m of another IE. Linking multiple Impact Evaluators together can create more precise incentive structures and lower setup cost by leveraging existing IE infrastructure (Benet, 2022a).

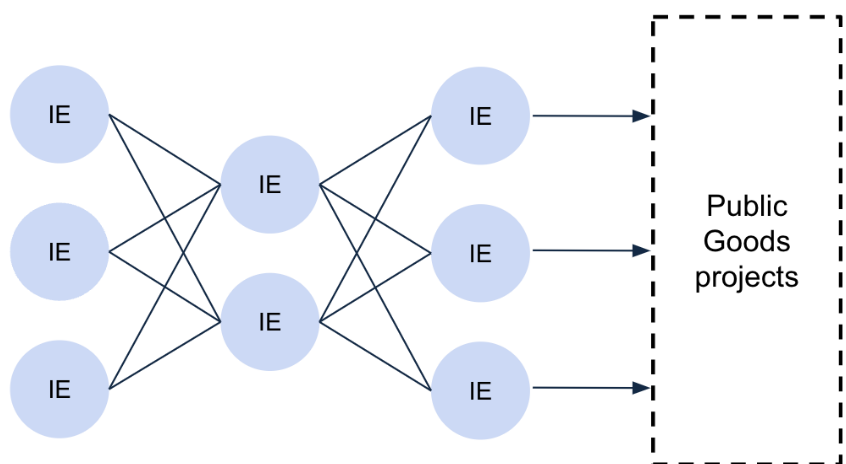

Figure 5: *Illustrating composability of Impact evaluators*

Hypercerts and interoperable funding systems

Some of the biggest challenges of IE operations are the discovery of relevant projects for their scope and the standardization of project data through the measurement (m) function. Once discovered, attributing impact (and rewards) to all contributing parties in a project is a burden that falls on the impact evaluator. Additionally, it is difficult to track funding received over time by an IE and other mechanisms, as well as the impact created by those projects across the scope of multiple mechanisms.

Enter the hypercert ledger, an interoperable data layer for impact-funding mechanisms. Hypercerts create a standard structure for projects to claim impact, make these claims identifiable, traceable and transferable, and link to evaluations of impact created (Network Goods, 2023). This not only introduces efficiencies for IE's directly, but reduces transaction costs between funding mechanisms and can enable impact market dynamics to form.

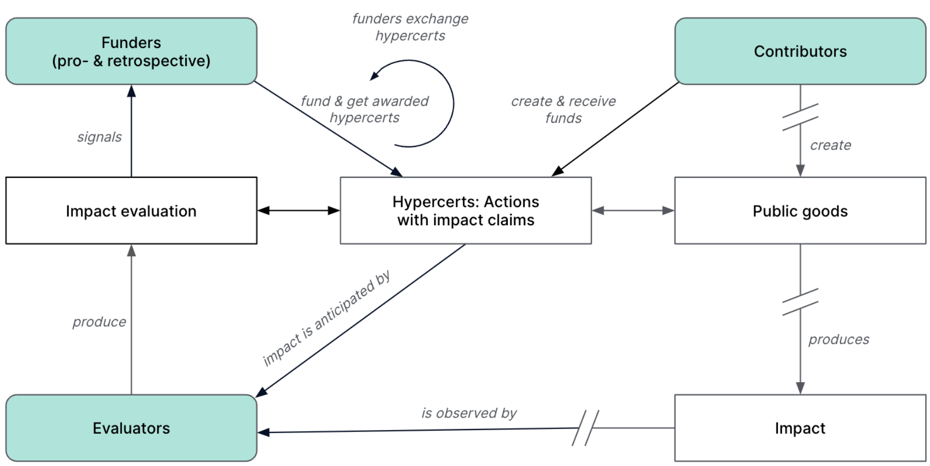

Figure 6: *Possible dynamics between participants in an impact funding system with hypercerts. Reprinted from Brammer (2022).*

Figure 6 illustrates how hypercerts facilitate impact funding systems, connecting the contributors of public goods to relevant funders and evaluating the value created. The right-hand side (contributors, public goods, impact) are measured by the measurement (m) function of an IE through a hypercert. The evaluators and impact evaluation maps to the evaluation function (e) and funders to reward function (r).

Through the IE function {*m,e,r*} we can see where Hypercerts provide value directly to IEs:

IE function	Value of Hypercerts
measurement (*m*)	**Medium:** consolidates indicator data (activities, outcomes) and contributors (entities) into a standardized format for input to IE
evaluation (*e*)	**Low:** *provides an open evaluation system to submit evaluations and optionally makes these publicly discoverable. No direct impact on the evaluation function*
reward (*r*)	**High:** efficient allocation of reward to contributors + tracking rewards over time

Table 3: *Hypercerts add value to all functions of an Impact Evaluator*

Credible and reputable Impact Evaluators create signals towards valuable projects which plays a critical role to create buy pressure for hypercerts in impact markets. This is what funders themselves need to signal to their stakeholders – such as customers for companies, the public for governments or philanthropists for foundations. Ultimately these stakeholders can now evaluate the effectiveness and efficiency of the funding decisions. Funders are enabled and incentivized to use their funds more wisely.

5. Conclusion

Impact Evaluators are an evolving concept that has spawned out of conversations in the web3 public goods / regenerative economics space. This concept is still being defined in relation to classical economics / science and requires further definition in relation to characteristics of blockchains and DAOs (which also can suffer from unclear definition). However we think the key innovation in Impact Evaluators is an incentive system with repeated evaluation and reward and the power of that system to generate large-scale value in high uncertainty public goods.

This paper is an early attempt to bring more structure to the discussion around this mechanism and share the knowledge from early experiments and learning. We hope that this will serve as an accelerant for those deploying or considering IE-like structures, and provide a common language to define and discuss these structures. This concept will undoubtedly evolve and further definition is required for many concepts that were introduced at a high level that were beyond the scope of this paper such as:

- Incentive structures for experts in subjective IEs to create incentive-aligned input
- A deep discussion on design of reward functions, particularly the interplay with tokens that accrete value from IE success and the ties to blockchains
- Formal structure on the ROI of Impact Evaluators and where they are more effectively deployed
- The connections to a broader interoperable funding system

We do note there are fields (e.g. economics, complex systems) that may overlap with concepts presented here, but a full comparison is beyond the scope of this paper. Finally we'd like to acknowledge that this work built on the thinking and contributions of many people including Juan Benet, Evan Miyazono, and members of the Network Goods team at Protocol Labs.

References

Several resources informed the thinking in this doc and are interesting sources to continue exploration:

Benet, Juan (2022a). Intro to Impact Evaluators.
Video, https://youtu.be/TdDHWv00Z4E, retrieved 21st December 2022

Benet, Juan (2022b). Impact Evaluator Design
Video, https://youtu.be/1soPQ31ZHkQ, retrieved 21st December 2022

Brammer, Holke (2022). Hypercerts: A new primitive for public goods funding, Blogpost, https://protocol.ai/blog/hypercert-new-primitive/, retrieved 19th December 2022.

Miyazono, Evan (2021). Impact Evaluators
Video, https://youtu.be/dpLtrugjfMc, retrieved 21st December 2022

Nakamoto, Satoshi (2008). Bitcoin: a Peer-to-Peer Electronic Cash System

Whitepaper, https://bitcoin.org/bitcoin.pdf, retrieved 21st December 2022

Network Goods (2023). Hypercerts: A new primitive for impact funding systems (whitepaper draft v0), [manuscript in preparation for publication]

Optimism (2021). Retroactive Public Goods Funding
Blogpost,
https://medium.com/ethereum-optimism/retroactive-public-goods-funding-33c9b7d00f0c, retrieved 21st December 2022

Protocol Labs (2022). A Public Goods experiment on Filecoin: retroactively funding impact with Quadratic Voting,
https://research.protocol.ai/blog/2022/a-public-goods-experiment-on-filecoin-retroactively-funding-impact-with-quadratic-voting/, retrieved 21st December 2022