星云开发者激励协议紫皮书

星云研究院

2018 年 6 月
版本号:1.0.1

目录

1 概要

2014 年以太坊的出现，标志着区块链行业进入 2.0 时代 [?]，其中支持图灵完备的智能合约的引入使得区块链应用开发者高效快地开发上层应用变为可能。具体而言，一个分布式应用（DApp）可以理解为为了实现特定功能的一系列智能合约的集合。正如广为人知的 AppStore 上的应用一样，DApp 可以包括游戏、博彩、社交等多种类型。相比于传统的 App 应用，DApp 具有如下特性：

- 应用数据分布式存储在区块链上所有节点，而不依赖于任何中心化的服务器，数据公开透明、不可篡改；
- 应用逻辑完全由智能合约代码决定，绝大多数智能合约会公布其源代码，从而保证可信度

下面这个切入点太直白了，试试这个：区块链做为典型的社区驱动的项目，对于区块链平台而言，维护社区中各方的利益，是不可推卸的责任和义务。然而，目前，大多数公链平台漠视、甚至损坏开发者的利益，这对于整个区块链行业及生态的长远健康发展，是极为不利的。因此，本文提出开发者激励协议。出发点应该是我们要维护社区的利益，而不是为了让星云繁荣，提出了这个东西。所以后续的措辞可能都需要改一下 对于区块链平台而言，DApp 应用数量和质量直观反映了其社区繁荣程度，也影响着平台发展潜力。迄今为止注明时间，根据 DAppradar 统计以太坊上已有超过 632 款 DApp[1]，相比之下目前大部分其他基础公链的 DApp 发行数量都远不及以太坊。值得注意的是，星云链一直致力于发展 DApp 生态并为开发者提供友好的开发平台。经过历时两个多月的激励计划不要提激励计划，目前星云链上已有超过 6871 个 DApp[2] 这个统计和对比是不严谨的。

伴随着 DApp 爆炸式增长，暴露出的问题是 DApp 的质量参差不齐。因此，如何对海量的 DApp 进行评价以提供更好的用户体验，以及如何促进激励开发者开发新的高质量 DApp 成为每条公链必须解决的问题。

所谓激励是指通过特定的方法让用户产生行为动机。对应地，星云开发者激励协议（DIP）是指以鼓励开发者开发优秀 DApp 为目的，以一套具有抗作弊性的 DApp 排名算法为核心，以对优秀 DApp 开发者给与 NAS 奖励为手段的一系列开源的激励机制的集合。简单而言，优秀的 DApp 将在我们的排名算法中获得较高排名进而获

[1] https://DAppradar.com/DApps.
[2] https://incentive.nebulas.io/summary.html.

得较高的奖励（在星云链上以官方代币 NAS 形式发放），以此来激励开发者。我们期望，DIP 能使每一个理性的诚实 DApp 开发者自发的设计优秀的 DApp，正如日常生活常见的评奖活动一样。同时，每一个理性的非诚实的开发者无法通过各种作弊手段来骗取 NAS 奖励。

我们认为开发者激励协议需要满足如下基本性质：

- 公开性：链上的 DApp 激励协议与传统的评奖方式最大的不同在于，所有评分的机制必须是完全公开的，且其中任何统计，计算，评选的过程都是全程可见的。这样就杜绝了传统中心化评奖暗箱操作的可能。同时也不会出现票数统计出错等情况。最后，根据评选结果分配奖励的过程也会保证被执行，奖励分配正如链上交易一样可被追溯。

- 有效性：这也是任何评选机制所要满足的基本性质。我们期望 DApp 评分能够真实反映用户的评价，即排名高的 DApp 是活跃用户所喜欢的且经常被调用的，而评分低的 DApp 是用户鲜有问津的。

- 抗作弊：对于任何排名算法，都需要解决各类作弊问题。对于 DIP 而言，主要存在两类作弊问题。1. 女巫攻击：区块链技术的一个重大特点就是一个用户建立新的节点地址代价是很小的。所以一个用户有可能建立多个由他控制的地址，并将他们伪装成多个正常用户来参与评选。一个好的激励协议应当保证每个用户无法通过女巫攻击带来巨大额外收益。2. 收买：由于我们衡量 DApp 好坏的主要指标是活跃用户调用的次数，一个 DApp 开发者有可能收买大量用户让他们调用自己的 DApp 以提高自己的排名从而获得更多奖励。这种作弊方式原则上无法杜绝，但我们期望激励协议能够让此类收买需要付出的代价变得很高以减少其出现的概率。

事实上，正如目前区块链中存在不可能三角（The Imposible Triangle）一样[3]，设计一个能够满足上述特性的激励协议存在很大挑战，同时我们需要保证一定的高效性与可扩展性。基于已有的星云指数（Nebulas Rank，NR）[?], 我们提出的激励机制能够在一定程度上保证上述性质。

为了实现公平性，我们把用户调用 DApp 的次数作为影响评分的主要标准。正如各大排行网站出现的点击榜一样，这样能够保证好的 DApp 受到了更多关注，并能促进整个系统的活跃性。

为了抵抗女巫攻击，我们应用了星云指数（NR）本身具有的性质。我们让具有更高 NR 值的用户拥有更多的投票权。由于 NR 的设计保证了伪造高 NR 用户的困难性从而有效防止了用户这方面的女巫攻击。同时，我们最终奖励函数的凹函数性质能保证开发者将他的 DApp 拆分成多个小 DApp 不会提升他的收益。

[3]https://news.8btc.com/the-impossible-triangle-of-blockchain.

为了防止收买，我们给与将票投给多个 DApp 的用户更高的投票权。因为被收买的用户往往只将票投给一个人，导致效用较低，这样就增加了收买的代价。

本紫皮书的结构如下：TBA

2 背景

本章主要介绍区块链激励机制和传统投票系统相关背景。鉴于目前尚无基于用户行为的 DApp 评价体系，我们进一步地介绍了相关设定。

2.1 区块链激励机制

比特币作为区块链的太初应用，践行了其作为“一个去中心化电子现金系统”的初衷。比特币的产生不依赖于任何机构，而是根据特定算法，依靠大量计算产生，保证了比特币网络分布式记账系统的一致性。尽管比特币并不支持智能合约特性，但其本身已经具有完善的激励机制。比特币的去中心化一方面是依赖于技术手段，另一部分则是通过巧妙的激励设置来实现，后者已成为广义上共识机制的重要组成部分。

比特币对新挖出区块的矿工的奖励本质上就是一种激励机制。它包括两个方面，对矿工的固定奖励（每 4 年减半，目前为 12.5BTC）以及区块打包所有交易的交易费（transaction fee）。值得一提的是，有关研究指出当固定奖励变得足够小时，光靠交易费无法维持比特币共识算法的稳定性，因为矿工将拥有更多的手段进行进攻同时获利 [?]。这说明适当的通货膨胀是对整个生态是有利的，也为星云每年增发适量的 NAS 用于生态开发提供了依据。

作为区块链 2.0 的代表，以太坊突破性地提供了图灵完备的智能合约，从而大幅拓展了应用场景。智能合约本质上是一个被代码控制的账户（smart contract account），而基于智能合约可以实现功能复杂的应用则称之为去中心化应用（decentralized application，DApp）。从技术架构来看，现有的大部分 DApp 通常以链上智能合约作为后端 SPA，同时采用了常用的前端技术与之交互，因而 DApp 形态即可以是传统 PC 客户端也可以是移动 App 或者 Web 形式。

目前以太坊上已经拥有一定数量的 DApp，涵盖了游戏、博彩、众筹、借贷等众多类型，其中以 2017 年底的以太猫（CryptoKitties）和 2018 年中的 Fomo3D[?] 最为出名，两者一度引发了以太坊网络交易拥塞。实际上，大量 DApp 正如同前两者一样，本质上都是充斥着氪金交易或者资金盘的商人游戏，而真正具有应用场景的杀手级 DApp 目前仍未出现 [?]。

本质上来说，这是由于价值尺度的缺失所导致，在缺少客观的评价标准下，开发

者亦没有开发优秀 DApp 的动机。一般情况下，开发者除了兴趣支撑外，主要获利手段为在智能合约中写入用户调用需要支付费用，或者 DApp 本身具有获利属性。

因此，针对开发者的激励机制将有望打破当前困境从而彰显重要性。在星云链开展的激励计划中，总共产生了超过 6781 个 DApp，并且大量优秀开发团队得以走向前台并获得高额投资 [?]。其他类似公链也随之效仿推出了短期的基于中心化管理的激励活动，此类激励活动以对社区进行宣传为主要目的，在这类活动中官方主观评价占据主导因素，并且缺乏长期持续性。

2.2 投票制度相关

因为在我们的 DApp 排名算法中起主要作用的是用户和 DApp 的交互，故其本质上可以看做一个用户给 DApp 投票的过程。关于投票系统各个领域已有大量的相关工作。其中最著名的结果是所谓的 Arrow 定理 [?]。其指出不存在任何一个排名算法能够同时满足非独裁性，帕累托有效性（Pareto Efficiency，即排名结果符合大多数人的利益）以及无关侯选者独立性（Independent of irrelevant alternatives，即两个候选者的排名相对关系不会受第三者影响）。这说明任何排名算法都不可能面面俱到。我们的排名算法将更多的侧重于重要程度较高的以及广为人知的属性。

在现实生活中也有大量需要用到排名算法的场景。其中一个典型和我们类似的例子为亚马逊以及淘宝平台中买家对卖家（商户）的评分。好评率较高的商户将被推荐系统排在靠前的位置从而获得较高的关注度。特别的，这类电商平台存在着和女巫攻击类似的问题，即刷单问题：商户可以以各种手段雇佣大量买家账号为其给五星好评。就目前而言，即便是此类中心化平台方刷单的手段大部分为通过机器学习手段判断真实用户和虚假用户 [?, ?, ?]。然而实际表明此类方法效果并不理想。[?] 指出甚至人工识别都不能有效判别此类账户。[?] 从机制设计的角度给出了一个消除商户刷单动机的算法，虽然和我们的模型不同但具有一定的借鉴意义。

其他场景下的排名算法：

[?] 介绍了网络社区给帖子排名的算法，结合了用户的投票数以及随时间衰减的过程。[?] 介绍了 Reddit 上帖子的牌名算法，引入了考虑了用户可以投反对票的情形。[?] 介绍了 Reddit 关于评论的排名算法，将置信区间考虑了进去。IMDB[?] 上对电影的排名引入了贝叶斯平均的思想，可以拉近不同电影之间投票人数的差异。

在我们的设计中，得益于星云指数 [?] 的高抗作弊性，反而能够更加清晰的判别真实用户与虚假用户。故我们设计的重点在于将用户的 NR 值通过交互行为转移到 DApp 的评分上来。

2.3 DApp 调用相关

在星云的 DApp 开发生态中，每个用户具有一个 NR 值，他可以调用（一个或多个）DApp，每次调用用户会消耗一定的 gas 费用用于执行智能合约。gas 费用最终将支付给矿工。同时，根据智能合约性质的不同，用户有可能会直接支付一定数量的 nas 给智能合约地址，最终由 DApp 开发者获得。星云团队（多少钱?）会支付最高 xx nas 用于奖励优秀 DApp 开发者，其中参与评选活跃用户越多（根据 NR 来判断）奖励总额越高。

这里我们需要说明的是，用户每次调用所花费的 gas 费用以及向合约地址支付的费用不会被排名算法考虑进去，即用户支付更多的钱无法提高其投票效用。前者是因为在星云系统中 gas 费用平均只有 10^{-8}nas 这个级别，完全可以忽略不计。

而不考虑后者的原因在于有效防作弊手段的缺失。咋看之下，用户愿意支付一笔钱给 DApp 确实能提高该 DApp 的被认同度，但实际情况下，这笔钱的最终流向存在以下三种可能。

1. 这笔钱最终归 DApp 开发者所有。这种情况可以认为用户自愿付出这些钱给 DApp，值得提高该 DApp 排名。但此时 DApp 开发者相当于已经从中获取了调用费用，再提高其排名的意义降低。

2.DApp 开发者承诺所有为其投入的钱最终会返还到用户手里。这本质上是一种作弊手段，而据此提高 DApp 排名将会助长此种作弊手段。

3.DApp 本身是赌博类游戏，与用户之间存在大量资金交互。这本身是一种诚实的应用，但此时用户与 DApp 的资金交互主要在于用户希望通过赌博赚钱。所以用户乐于支持 DApp 的钱可能为 0，不应据此提高 DApp 排名。

介于目前我们无法判断用户和智能合约地址的资金交互属于上述哪一种情况，且上述任何一种情况都存在不介入排名的理由，故我们最终的算法也将独立于用户和 DApp 间的资金交互。

3 模型

我们用 $\mathcal{A} = \{a_1, a_2, ..., a_m\}$ 表示所有 (m 个) 用户的集合。并用 $NR_1, ..., NR_m$ 表示这些用户对应的 NR 值。我们用 $\mathcal{D} = \{d_1, ..., d_n\}$ 表示所有 n 个 dapp 的集合。e_{ij} 表示用户 a_i 调用 dapp d_j 的次数。

用 $score_1, score_2, ..., score_n$ 表示我们给 n 个 dapp 的排名分。M 表示星云团队

用于激励计划的 nas 总额。函数

$$u_i(score_1, score_2, ..., score_n), i = 1, 2, ...n$$

表示第 i 个 dapp 最终获得的奖励。

关于时间的说明

4 开发者激励协议

4.1 从调用次数到投票贡献值

对于任何一个用户 a_i，我们将 NR_i 认为是该用户能投票的总效用值，即可以当作他手中握有的选票总张数。这意味着 NR 值高的用户可以获得更高的投票权。根据章节2.3的说明我们暂时不考虑用户调用 dapp 过程中的资金交互。在后面的章节中我们会讨论基于完善的资金流向检测机制下如何引入资金交互来扩展该模型。

设 NR_{ij} 为 a_i 对 d_j 的分贡献值，其定义为

$$NR_{ij} = \frac{e_{ij}}{e_{i0} + \sum_{j=1}^{n} e_{ij}}$$

即对 d_j 调用次数在总调用次数上的占比。这里 e_{i0} 表示 a_i 不属于上述任何一个 dapp 的调用次数。用户 a_i 可以任意调整 e_{i0} 以及 e_{ij} 的数值。

易见，

$$\sum_{j=1}^{n} NR_{ij} \leq NR_i$$

值得一提的是，本小章我们需要达到的目的仅仅是让用户可以任意分配自己的 NR 值用于投票（即任意选择分贡献值）。有些 dapp 可能采取强行增加调用次数的手段（比如规定必须调用两次才生效），但因为用户调用次数和 NR 分配现状都是可见的，用户仍然可以通过调整调用次数来达到自己期望达到的 NR 值分配方式。并且即使我们不是根据次数线性分配 NR 而是采用其他方式，聪明的用户总能找到其他投票方案以实现（期望达到的 NR 值分配方式）。

引入 e_{i0} 的意义在于，为了保证用户的个人理性（individual rational，即参加此次活动不会损失利益），我们不强制用户投出所有的选票，用户可以选择性的行使部

分投票权或完全弃权，通过适量增加 e_{i0} 的值。[4]这适用于用户觉得质量上乘的 dapp 太少的情况。

4.2 从分贡献值到排名分

给定所有的分贡献值 $NR_{ij}, i = 1, 2, ..., m, j = 1, 2, ..., n$，定义 dapp d_j 的排名分为

$$score_j = \sum_{i=1}^{m} \sqrt{NR_{ij}}$$

易见，当用户 a_i 只投给一个 dapp 时，他的投票分贡献值之和为 $\sqrt{NR_i}$。而当他将票分散给不同的 dapp 时，根据根号函数的反叠加性，他投票的总效用会提高，这意味着他接触了更多的 dapp，而这也是我们的系统所鼓励的。下一章会证明这样构造排名分的方法具有很好的性质。类似的思想可用见于 V 神的论文 [?] 里面用到的二阶投票算法，虽然两者的模型完全不同。

当排名分 $score_i$ 给出之后，dapp 排名也将据此在链上展示。排名分高的 dapp 将被放在更显著的位置，也会受到更多的关注。值得一提的是我们假设用户只关心心目中的 dapp 的排名，而不会关心 dapp 开发者具体分配到多少奖励。这在我们后面的用户行为分析章节中会详细讨论。

4.3 从排名分到最终奖励

给定所有 dapp 的排名分 $score_j, i = 1, 2, ..., n$，定义 dapp d_j 开发者的奖励为

$$u_i = \frac{score_j^2}{\sum_{k=1}^{n} score_j^2} \times \lambda M$$

其中 M 是星云团队用于发放奖励的最大值，λ 表示参与系数，即我们希望参与评选的用户越多给的奖励总额越大，具体定义为

$$\lambda = \min\{\frac{NR_p}{NR_s \times \alpha} \times \min\{\frac{NR_p^2 \times \beta}{Var_p}, 1\}, 1\}$$

[4] e_{i0} 的实现可以通过官方设立一个类似“垃圾桶”的虚拟 dapp，不含任何实际效用。用户可以调用“垃圾桶”任意次数。

其中

$$NR_p = \sum_{i=1}^{m}(NR_i - NR_{i0}), \quad NR_{i0} = \frac{e_{i0}NR_i}{e_{i0} + \sum_{j=1}^{n} e_{ij}}$$

为参与投票的用户有效分贡献之和。NR_s 为社区所有用户 NR 的总和。Var_p 为参与投票用户有效分贡献的方差，其最大值为 $\frac{(m-1)^2}{m^2}NR_p^2$。$\alpha, \beta < 1$ 为可调的参数。引入参与系数目的是让参与活动的用户总 NR 达到某个阈值（社区总 NR 的 α 倍），以及让参与评选用户 NR 值的方差限定在某个范围内，防止出现少量高 NR 值用户带大量虚假账户的情况。两者可互补，即参与投票用户 NR 总值够高时可以不考虑方差的影响。

5 性质分析

这章我们主要分析我们的排名算法所剧本的各种防作弊性质。主要包括防收买，防恶意分拆 dapp，防女巫攻击等等。在实际运作中，我们可能根据实际情况对具体公式会有所调整，但这些性质仍能够保证大致满足。

5.1 抗收买

所谓收买是指 dapp 开发者通过各种手段获得用户的全部投票权。这里我们假设所有正常用户都是利益最大化的。我们认为用户关心的是他心目中 dapp 在排行榜上所处的排名，而不关心 dapp 开发者最终会获得多少奖金。亦即，每个用户乐于最大限度的提升自己心目中优秀 dapp 的排名分（score 值）。我们的二阶排名算法保证了下面这个特征：

特征 1. 对于一个利益最大化的用户，一般而言，他会将手中的票投给多个不同的 dapp。

我们用如下模型来具体说明：

不失一般性，假设用户 a_i 所有 dapp 的价值权重为 $b_{i1}, b_{i2}, ..., b_{in}$(可理解为用户的打分)，则该用户最终分配的分贡献值满足

$$\frac{b_{i1}^2}{NR_{i1}} = \frac{b_{i2}^2}{NR_{i2}} = \cdots = \frac{b_{in}^2}{NR_{in}}$$

具体证明见附录。

若使用更常见的线性排名分算法（即 *score* 为所有的分贡献值之和，而不是根号之和），理性用户则只会将所有票投给自己最喜爱的 dapp。相比之下，我们的算法更能促进用户和 dapp 之间的交互。造成这样的原因在于根号函数的特性，即用户把票投给多个 dapp 相当于他的总票数变多了。故用户会倾向于投给多个 dapp 的同时保证自己最喜欢 dapp 的领先性，即上面的比例等式。

实际生活中，某些常见投票方式为限制用户为单个目标投票的最大票数，从而强制让用户投给多个目标。而我们的算法则是通过激励从本质上达到同样的目的，且在数学表达更为简洁，更适合作为激励协议的内容。

下面这个推论体现了我们的算法抗收买的特性:

推论 1. 被收买的用户对 dapp 排名分的总贡献远远小于正常的用户。

显而易见，已经被收买的用户 a_i 最多给收买其的 dapp 开发者贡献 $\sqrt{NR_i}$。而一个正常用户，假设他认为好的 dapp 有 K 个，在其对这些 dapp 的价值权重是分布较为均匀的情况下，他给所有 dapp 带来的总排名分提升大约在 $O(\sqrt{K})$ 这个级别[5]，即他起到的作用是被收买用户的 $O(\sqrt{K})$ 倍。这样就一定程度上提高了开发者收买用户的代价。

5.2 抗恶意分拆

（增加开发者开发多个正常 dapp 的讨论?）

所谓恶意分拆是指 dapp 开发者将自己的 dapp 强行分拆成两个或多个低质量的 dapp 以获得所有分拆的 dapp 的总奖励。我们认为 dapp 开发者关心的是最终获得的总奖金。同时也有一定的高排名带来隐性收益。

我们的最终奖励的凹函数性能保证下面的特征

特征 2. 在所有投票者全是正常用户的情况下，开发者进行 dapp 分拆不会提升他的收益。

这里我们假设对于一个正常用户，若拆分之前对该 dapp 的价值权重为 c，对两个拆分之后的 dapp 的价值权重分别为 a 和 b，则 $c \geq a + b$。这个可以理解为 dapp 拆分之后两者的质量将大幅下降且缺乏联动性，导致两者的质量和比原 dapp 还要差。该特征的具体证明见附录。

[5] K 反应的是该用户给出具有区分度的投票的 dapp 数目，通常是大于 1 的，只要该用户对 dapp 价值权重分布不是太极端，即只喜欢某个特定的 dapp 而对其他 dapp 打分都趋近于 0。

开发者还可采取的作弊手段为同时进行 dapp 分拆和收买，例如，先分拆成 K 个 dapp，然后让被收买的用户均匀的将票投在自己拆分的 K 个 dapp 上以实现效用最大化。我们有下面的推论：

推论 2. 即使引入被收买者的情况下，开发者进行 dapp 分拆不会提升他的收益。

具体证明见附录。

值得注意的是，开发者将 dapp 进行拆分同时也会降低在排行榜上的排名，从而减少排名带来的隐形收益。综上所述，我们的算法能从本质上防止 dapp 分拆的进攻手段。

5.3 抗女巫攻击

所谓女巫攻击是指一个原本高 NR 的用户将自己的财产转移到多个新账户里，然后多个账户作为被收买的新用户统一给某个 dapp 投票。

解决这个问题的主要手段是利用 NR 本身的抗作弊性质。假设进行女巫攻击的用户原 NR 值为 NR_{a+b}, 进行财产拆分之后两个新用户的 NR 值分别为 NR_a 与 NR_b。由于我们只需要考虑该用户给同一个 dapp 投票，故我们只需比较 $\sqrt{NR_{a+b}}$ 与 $\sqrt{NR_a}+\sqrt{NR_b}$ 的大小。若能满足

$$\sqrt{NR_{a+b}} > \sqrt{NR_a}+\sqrt{NR_b} \tag{1}$$

则以此类推可得到将 NR 值拆分成多个用户投票效用也不会增加。

我们可通过下面三种方案来实现这个目标：

- 通过调整 NR 本身的参数使得在一定范围内满足 (1)。虽然在 NR_a, NR_b 均趋于无穷时有 $NR_{a+b} = NR_a + NR_b$，与 (1) 矛盾，但因为用于女巫攻击的新用户 NR 值都较低，故我们只需在 NR_a, NR_b 时都很低时 (1) 成立即可，而这是 NR 通过调参所能达到的。
- 通过对所有参与投票的用户的 NR 值进行预处理，一个简单的方案为将所有 NR 值直接平方。那么 (1) 可根据黄皮书 [?] 里 Wilbur 函数的性质 $f(x_1+x_2) > f(x_1)+f(x_2)$ 直接得到。值得一提的是，将 NR 进行平方的预处理仅仅用在 DIP 中，而对 NR 的其他应用均不作处理，进而也不会损害 NR 本身的性质。
- 通过只取 NR 排名前 L 或 $L\%$ 的用户。这样拆分成多个小 NR 用户可能无法达到排名要求从而增加女巫攻击的代价。而这也是白皮书 [?] 里所采取的方法。

我们认为，通过上述三种方案的结合使用能够有效防止女巫攻击。

6 DIP 的实现

7 扩展

1. 当对调用合约是价格的流向有所监控的时候，可以考虑提高钱的权重。
2. 可以考虑引入 dapp 之间的调用

附录 A　证明

A.1　特征 1 证明

证明. 不失一般性，假设用户 a_i 对所有 dapp 的价值权重为 $b_{i1}, b_{i2}, ..., b_{in}$。这些均为定值。假设用户 a_i 对所有 dapp 的分贡献值分别为 $NR_{i1}, ..., NR_{in}$，这些为可调整的变量。

用户 a_i 的优化目标为他所提供的总排名分，定义为

$$w_i = \sum_{j=1}^{n} b_{ij}\sqrt{NR_{ij}}$$

根据柯西不等式可得

$$w_i = \sum_{j=1}^{n} b_{ij}\sqrt{NR_{ij}} \leq (\sum_{j=1}^{n} b_{ij}^2)(\sum_{j=1}^{n} NR_{ij}) \leq (\sum_{j=1}^{n} b_{ij}^2)NR_i$$

上式最右边为定值。等号成立当且仅当

$$\frac{b_{i1}^2}{NR_{i1}} = \frac{b_{i2}^2}{NR_{i2}} = \cdots = \frac{b_{in}^2}{NR_{in}}$$

故命题得证。

□

A.2　特征 2 证明

证明. 不失一般性，假设 d_1 开发者将其 dapp 拆分成 2 个 dapp。对于任何正常用户 a_i，拆分之前他对所有 dapp 的价值权重分别为 $b_{i1}, b_{i2}, ..., b_{in}$。拆分之后，假设 a_i 对于拆分出的 2 个 dapp 价值权重为 b'_{i1}, b'_{i2}。根据我们的假设有 $b_{i1} \geq b'_{i1} + b'_{i2}$

我们接下来计算拆分之前的分贡献值。计 $H_i = \sum_{j=2}^{n} b_{ij}^2$，根据特征 1 的结论及合分比定理有

$$\frac{NR_{i1}}{b_{i1}^2} = \frac{\sum_{j=1}^{n} NR_{ij}}{\sum_{j=1}^{n} b_{ij}^2} = \frac{NR_i}{b_{i1}^2 + H_i}$$

故

$$NR_{i1} = \frac{b_{i1}^2 NR_i}{b_{i1}^2 + H_i}$$

类似的可得到拆分之后 a_i 对拆分的第 t 个 dapp 的分贡献值为（定义为 $NR'_{it}, t = 1, 2$）

$$NR'_{it} = \frac{b_{it}'^2 NR_i}{b_{i1}'^2 + b_{i2}'^2 + H_i}$$

注意到 $b_{i1}^2 \geq (b\text{`}_{i1} + b\text{'}_{i2})^2 > b_{i1}'^2 + b_{i2}'^2$，可得

$$NR_{i1} > NR'_{i1} + NR'_{i2}$$

定义 $score'_1, score'_2$ 分别为 d_1 拆分之后的两个 dapp 的排名分，根据定义，有

$$score'_1 = \sum_{i=1}^{m} \sqrt{NR'_{i1}}, \quad score'_2 = \sum_{i=1}^{m} \sqrt{NR'_{i2}}, \quad score_1 = \sum_{i=1}^{m} \sqrt{NR_{i1}}$$

定义 u'_1 为拆分之后 d_1 开发者从所有 dapp 中获得的奖励，则

$$u'_1 = \frac{score_1'^2 + score_2'^2}{score_1'^2 + score_2'^2 + \sum_{j=2}^{n} score_j^2}, \quad u_1 = \frac{score_1^2}{score_1^2 + \sum_{j=2}^{n} score_j^2}$$

注意到在固定 $score_2, ..., score_n$ 的情况下，

$$u_1 \geq u'_1 \Leftrightarrow score_1^2 \geq score_1'^2 + score_2'^2$$

所以，为了比较 d_1 开发者在拆分前后的收益，我们只需要比较下面两个量

$$score_1^2 = (\sum_{i=1}^{m} \sqrt{NR_{i1}})^2, \quad score_1'^2 + score_2'^2 = (\sum_{i=1}^{m} \sqrt{NR'_{i1}})^2 + (\sum_{i=1}^{m} \sqrt{NR'_{i2}})^2$$

事实上，$score_1^2 \geq score_1'^2 + score_2'^2$ 可由最短路性质得出。如图1，构造一个网格，其长和宽均被分成了 m 段，其第 i 段长度分别为 $\sqrt{NR'_{i1}}$ 和 $\sqrt{NR'_{i2}}$。

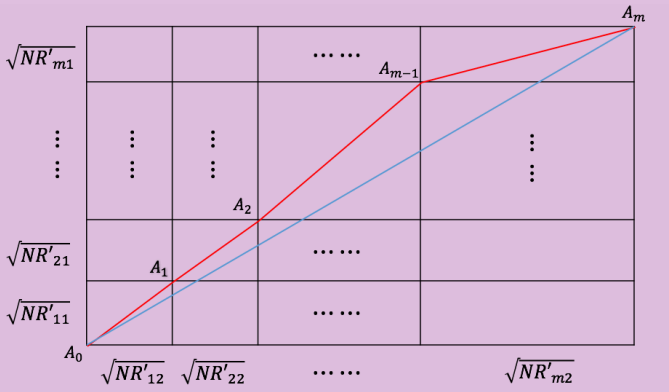

图 1: 最短路证明

$score_1'^2 + score_2'^2 = A_0A_m^2$，即恰好等于图中蓝线的长度的平方。而

$$score_1^2 = (\sum_{i=1}^{m} \sqrt{NR_{i1}})^2 > (\sum_{i=1}^{m} \sqrt{NR'_{i1} + NR'_{i2}})^2 = (\sum_{i=1}^{m} A_{i-1}A_i)^2$$

即所有红线长度之和的平方。根据两点之间线段最短可得 $score_1^2 > score_1'^2 + score_2'^2$。

对于拆分成 $k > 2$ 个 dapp 的情形，只需转化成逐次拆分然后每次应用 $k = 2$ 时的结论即可。

故命题得证。 □

A.3 推论 1 证明

证明. 对于一个被 d_1 开发者收买的用户，在 d_1 进行拆分之前，可以将该用户等价于一个价值权重向量为 $(1, 0, 0, ..., 0)$ 的正常用户。而拆分成 k 个 dapp 之后，假设被收买用户对这 k 个 dapp 的分贡献值为 $NR_{t1}, ..., NR_{tk}$，其和为定值。根据特征 1 的证明中柯西不等式取等号的条件，可将该用户等价于一个价值权重向量为 $(\sqrt{NR_{t1}}/C, \sqrt{NR_{t2}}/C, ..., \sqrt{NR_{tk}}/C, 0, 0, ..., 0)$ 的正常用户。其中 $C = \sum_{j=1}^{k} \sqrt{NR_{tj}}$。即对拆分出的 dapp 按某种比例分配权重，所有其他 dapp 权重为 0[6]。此时因为

$$\sum_{j=1}^{k} \sqrt{NR_{tj}}/C = 1$$

[6]注意将所有价值权重同时扩大若干倍对结论没有影响，因为最终用户分贡献值总和只与权重占据的比例有关。

可规约为特征 2 的情况，即变成满足假设的正常用户的情形。故命题得证。 □