关于 Mixin 免费 BTC 活动可持续性的数学分析

版本 1.0, November 24, 2019

俞一峻
计算及通讯系
开放大学, 米尔顿凯恩斯, 英国
y.yu@open.ac.uk

I. 引言

Mixin Networks 近期推出一项名为 “Lucky BTC Bot” 的推广活动[1]，通过这个机器人（bot），任何 Minxin 网络的用户都可以每天免费领取 $0.000001 \times n$ 个比特币（BTC），n 为连续领取的天数。如果某天中断了领取，那么可以再从 0.000001 个比特币重新开始。

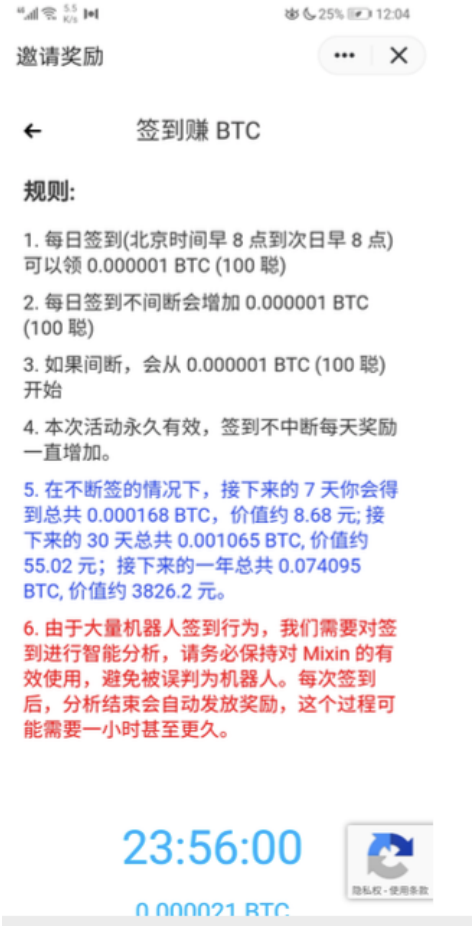

Fig. 1. Mixin 免费 BTC 领取规则

按照 Mixin CEO Cedric Fung（冯晓东）在视频直播中给出的解释，这个推广活动会一直持续地执行下去。而在活动中会适时推出一些免费的任务，需要完成以后才能继续领取免费的比特币。

一个明显的问题是，活动的每一个参与者（据说头一天就有十万人之多）都能够实实在在地获得规定份额的比特币。那么，由于所有的附加任务都是免费的，作为活动的主办方，不可避免地要支出这些比特币，最坏情况下[2]进行 n 天的活动，每人需要支出 $0.000001 \times n(n+1)/2$ 比特币。以活动参与人数 10 万计，主办方需要支出 $0.1n(n+1)/2$ 个比特币。这是什么概念？如果活动进行 1 年，$n = 365$, 一共要准备 6,679.5 个比特币，按照每个比特币目前价值 6 万元人民币计算，也就是 $400,770,000$, 即 4 亿元的费用。更进一步，到了第二年结束的时候，这个数字会飞涨到 26,681.5 个比特币，按照市价为 1,600,890,000, 即 16 亿元人民币，到了第三年结束的时候是 36 亿元人民币..., 到了第七年结束的时候是惊人的 196 亿元人民币。这还是建立在仅有 10 万人坚持不懈到底的假设下。

好吧，你会以为不会有 10 万人坚持到七年以后吧。也许是的，但是这个数字哪怕除以 10 也是不容小看的了。那么，冯晓东凭什么能够承诺活动无限期呢？另外，可以挖矿出来的比特币总数是 2 千 1 百万 [2]，这显然是有限的数量，这个活动在进行到第 18 年的时候，需要支出的比特币总数就会超过所有现在和将来可能发行的比特币总和！

践行群里流行的一种解释是，通过这个活动，参与者必须实名认证，并且每天签到，这就保证了大量的 “日活”，按照互联网公司的发展路子，这些每日活跃人数的一部分最终能够转化为公司的业务收入。

姑且不论这个解释是否合理，单单看维持每个日活的代价，我认为就不能自圆其说。至少在开始没有广告和额外收入的前提下，活动主办方的支出是实打实的，而每个用户也是仅仅以获得 BTC 为目标参与活动，因此暂时不能成为广告的对象。

那么问题来了，到底这个活动有什么特别之处，让它能够做出 “永远进行下去” 的承诺呢？

本文将从风险收益评估的数学分析模型出发，通过基于概率的马尔可夫链（Markov Chain）行为模型估算，得出背后的原理。当然，不可避免的，我们需要适当引入一些假设以便分析计算。

[1] https://bitcointalk.org/index.php?topic=5198497.0

[2] 这是从支出的角度看，如果是从用户收益的角度看这是最好的情况！

A. 对免费任务的感性认识

带着这个问题，作为用户我参与了这个免费派发 BTC 的活动。在活动开始的第 1 天，我被要求实名认证，这样避免让未经实名认证的假用户分走宝贵的 BTC；第 6 天，我被要求做一个任务，要求尝试把获得的 BTC 在 Mixin 平台上交易为人民币，向我证明得到的 BTC 是真实有效的；第 13 天，我被要求做一个 "查水表" 的任务，把之前得到的 0.000091 个 BTC 存放在 Mixin 钱包里面，证明我确实能够 hold 住好不容易得到的 BTC。正如冯晓东所说，这三个任务都是免费的，而且自然而然我能够做到的。至于后面的任务还没有分配，按照冯晓东的说明，所有未来的任务都应该是类似的免费任务。

II. 风险收益分析的概率模型

这些免费任务不是自动发生的，而是需要用户真实地参与。并且，直观地说，这些任务背后的逻辑似乎跟前面的悖论有一定的联系。下面，我们就用统计概率的模型来分析一下。

A. 背景知识（你可以跳过不看）

[定义 1] 一个非确定自动机（马尔可夫链）可以定义为一个有向图结构 $(S, s_0, \Sigma, \delta, \pi)$，其中:

- S 为状态点的集合;
- s_0 为 S 的初始状态子集;
- Σ 为输入符号的集合;
- $\delta : S \times \Sigma \to S$ 为状态迁移边的集合;
- $\pi : \delta \to [0,1]$ 为状态迁移的概率。对每一个状态而言，所有转出的迁移边的概率之和为 0 或者 1: $\sum_{s',i|(s,i,s')\in\delta} \pi(s,i,s') = \{0,1\}$。当和为 0 时，该状态称为吸收态或终止态。

为了评估风险收益，需要对这个自动机做一项拓展 $(S, s_0, \Sigma, \delta, \pi, \mathcal{I})$：

- $\mathcal{I} : S \to [0, \infty)$ 为状态影响函数。

[定义 2] 从初始状态 s_0 到某一个状态 s 的一个轨迹 $\langle s_0, s\rangle$ 定义为一个序列的 n 条状态迁移边 $(s_k, \_, s_{k+1}) \in \delta$，其中 $n > 0, k = 0, \ldots, n-1$, 且 $s_n = s$. 同一对状态 s_0 和 s 之间可以存在多条不同长度的轨迹。给定一条轨迹长度为 n 的 $\langle s_0, s_n\rangle$, 其抵达的可能性 $p(s)$ 可以定义为以下概率的乘积:

$$p(s) = \prod_{k=0}^{n-1} \pi(s_k, \_, s_{k+1}) \tag{1}$$

其对应的风险 $r^n(s)$ 可定义为状态影响程度 $\mathcal{I}(s)$ 和抵达该状态可能性 $p(s)$ 的乘积:

$$r^n(s) = \mathcal{I}(s) \times p(s). \tag{2}$$

把所有从初始状态 s_0 到达某状态 s 的轨迹一起考虑, 总的风险可以计算为

$$r^(s) = \sum_{n=1}^{\infty} \sum_{\langle s_0, s\rangle \in \delta^n} r^n(s). \tag{3}$$

[引理 1] 当 n 趋向无穷大时，状态 s 的抵达可能性和风险如果收敛，必为某一个可计算的数:

$$p(s) = \lim_{n\to\infty} p(s_n) \tag{4}$$

$$r^(s) = \lim_{n\to\infty} r^n(s) \tag{5}$$

即便在马尔可夫模型中的概率 π 和影响 $\mathcal{I}$ 函数定义中出现了变量，这两个函数仍然可以用代数方法求解 [3]，参见我在开源软件 https://github.com/yijunyu/demo-riskexplore 中的实现。

B. 建立数学模型

现在把我们的问题建模为如下离散时间的马尔可夫链 (DTMC)。这里我们使用牛津大学开发的概率模型检查工具 PRISM 中定义 DTMC 输入语言建模 [1]。

```
dtmc
const double p1;
const double p2;
const double p3;
const double c1;
const double c2;
const double c3;
const double c4;
module free_btc
  s: [0..3] init 0;
  []s=0->(1-p1):(s'=0)+p1:(s'=1);
  []s=1->(1-p2-p3):(s'=1)+p2:(s'=3)+p3:(s'=2);
  []s=2->(1-p1):(s'=0)+p1:(s'=1);
endmodule
rewards "value"
  s=0: c1;
  s=1: c2;
  s=2: c3;
  s=3: c4;
endrewards
```

首先我们从用户的角度建模马尔可夫链的数学模型.

图中共有 4 个状态和 8 条状态迁移边。这里，s_0 为初始状态，$c_0 = 0$；s_1 为领取第一个 c_2 即 100 satoshi 的状态；s_2 为一次领取 n 份 100 satoshi，即 $c_3 = 100n$ 之前的状态；s_3 为兑现全部累计 $c_4 = n(n+1)/2 * 100$ satoshi 的终止状态。

注意，这里面有一个循环，即 s_2 在领取完 BTC 后会进入 s_1（如果当天领取）或者 s_0 状态（如果没有当天领取）。另外，三个免费任务发生的概率分别为 p_1（当天领取）, p_2（验证 BTC 可以转化为人民币）和 p_3（验证 Mixin 钱包里面有所有已经积累的 BTC），其取值最坏情况下都为 1，正常情况下比 1 要小。

C. 代数分析

利用开源软件 https://github.com/yijunyu/demo-riskexplore [4]，我们计算出以上模型的代数解为

$$c_1(p_2 + p_3 - p_1p_2)/(p_1p_3) + c_2/p_3 + c_3p_2/p_3 + c_4$$

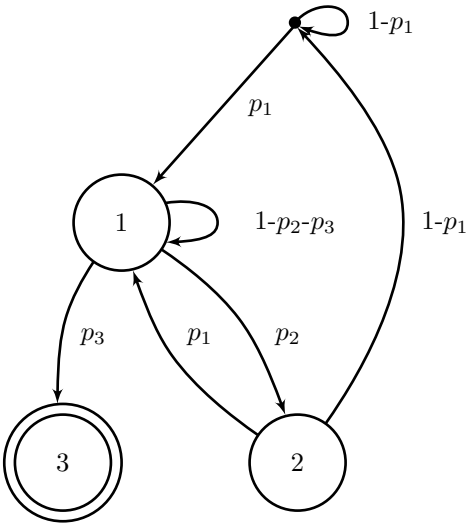

Fig. 2. 关于用户的马尔可夫行为模型

按照活动的设置，我们已知 $c_1 = 0, c_2 = 0.000001, c_3 = c_2 n, c_4 = c_2 n(n+1)/2$。可是这样带入一算，就发现，以上的收益函数为：

$$c_2((1 + np_2)/p_3 + n(n+1)/2)$$

随着 n 趋向无穷大，该函数不收敛，也趋向于无穷大！也就是说，对用户而言，收益无穷大，反之，对于活动主办方而言，损失也就无穷大！

哪里不对了呢？

原来这里有一个不容易觉察到的细节：在从 s_2 回到 s_1 的状态以后，每一次 “查水表” 任务之后，在 Mixin 账户中的余额必须大于或者等于所有的免费获得的 BTC。

也就是说，从 Mixin 网络总体当作一个账户来说，并没有支出 BTC。什么时候会支出呢？只有当用户执行 “BTC 到 RMB” 任务的时候，才发生 BTC 从 Mixin 网络的流失。

如果考虑从整个 Mixin 网络的角度分析，答案就不同了。同样的马尔可夫链模型，状态到达的影响函数不同了。假设初始有 1000 个 BTC，由 $m = 100000$ 用户分。$c_1 = 1000/m$。$c_2 = -0.000001$, 即任务 2 的效果是把这部分初始的比特币转换为人民币。$c_3 = 0.0000001$, 即通过查水表按照要求用户需要往 Mixin 账户里面充值以保证总额不少于免费获得的 BTC。$c_4 = 0$，这是因为 Mixin 网络并没有把账户里面的 BTC 转出。重新计算以上收益函数乘以用户数 m，得到

$$1000(p_2 + p_3 - p_1 p_2)/(p_1 p_3) - 0.1(1 - p_2)/p_3$$

你会发现，这个函数不仅收敛，反而还有可能扭亏为盈。

D. 讨论

这是镜像世界多好的例证啊！

做一个类比，这就像你去一家银行开了一个户头，存一小笔钱。然后每天银行从一个虚拟的账户上往这个户头上都存钱。只要你不把这个账户上的钱取走，那么从银行的角度来看，所有存款交易的总和还是平衡的，只不过这个虚拟的账户是 “借记” 而已。什么时候这样的银行会倒闭呢？只有当大量用户同时去 “挤兑” 的时候，才会倒闭。因此，只要银行有足够的机动资金应对个别用户的挤兑，银行就不会倒闭。相反，银行还可以用存款去借贷，以钱生钱。

这个类比跟我们的主要研究对象还是有所不同的。这体现在银行用户存款是来自于用户的，而 Mixin 网络给出的 BTC 并不是来自于用户的，因此，它更不容易让用户损失，只要这个用户不把辛苦得到的 BTC 去兑换为法币。但是，如果不幸兑换了，那么用户需要补足兑换的损失 (“查水表”)，这样才能够继续免费获得 BTC。

III. 结论

那些对定投不坚定的用户，不能长期坚持定投，连长期领取免费 BTC 的任务都完成不了的用户，对不起，你是无法最终获利的。

与其说这个服务像银行业务，不如说它更像是对定投者的一项保险业务：当你不能坚持到底的时候，虽然能够得到部分的法币，但是 BTC 账面上会清零，还得重新启程。

这里需要注意的是，Mixin 网络的存在本身就是有价值的，这个价值来源于大家对 BTC 和其它数字资产的信任。只要这个信任存在，就可以一直继续下去。如果像冯晓东说的那样，初始拿出了 1000 个 BTC 作为信用担保，而且用户不会随随便便把好不容易得到的 BTC 去变现，那么这个活动就可以永久持续下去！

IV. 注意事项

以上数学分析基于很多的假设甚至臆测，模型也比较抽象，结论仅供参考。

Appendix

安装和运行 Risk Explorer 的步骤 参见 README.md：

```
cd prism/prism && make && cd ../..
brew install R
brew install gawk
cd docker; ./b; cd ..
cd risk && ./b && cd ..
./r examples/MIXIN1
```

References

[1] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[2] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. https://bitcoin.org/bitcoin.pdf.

[3] Yijun Yu and Chun Liu. Little model in big data: An algebraic approach to analysing abstract software behaviours 小模型大数据: 一种分析软件行为的代数方法. *Journal of Software(*软件学报*)*, 28(6):1488–1497, 2017.

[4] Yijun Yu, Nobukazu Yoshioka, and Tetsuo Tamai. Assessing security and privacy behavioural risks for self-protection systems. In *Engineering Adaptive Software Systems - Communications of NII Shonan Meetings*, pages 135–147. Springer, 2019.