STAKING REWARDS

THE GOOD, THE BAD AND THE UGLY

DEMYSTIFYING RSS

DISCLOSURES AND AFFILIATIONS

- I am not a mathematician (but studied algebra, -calculus and -statistics in the universities).
- Bear in mind, I could easily misinterpret how RSS works or intended to work.
- I am an SPO and ADA holder, therefore I might biased, though I am aware of it. ;)
- I have no any affiliates with any entities relating to Cardano Blockchain.
- I am not a member of any pool alliances (SPOCRA etc.). I am only member of the TG guild and the Official Cardano forum and TG channels.
- I think, my moral incentives are a little bit stronger (to a certain threshold:)) than my financial ones.

AGENDA

There a 3 main parts of the RSS:

1. Rewards Sources (Tx Fees, Monetary policy, decayed deposits), which are used for calculating the ***total epoch reward*** (WON'T BE DISCUSSED HERE).
2. Reward Sharing Scheme (RSS) for calculating (**from the above total epoch reward**) and distributing **pool reward**s. (for simplicity, the unclaimed rewards and some other which irrelevant to the topic I would like to explain, won't be discussed here)
3. Ranking, which is acting as an additional incentive for stake holders (operators and delegators) to have the system to reach the expected equilibrium for long term. (WON'T BE DISCUSSED HERE)

THE BASICS

- Formulas are scary, hard, confusing and probably have no meaning for the general ppl in our case to us the SPOs and to the delegators.
- My aim is to give some meanings to the formulas required to understand the fundamentals of the reward calculation and distribution, by trying to provide some ELI5 like explanations and simple visual representations of the formulas.
- The aim of having SPOs in the Cardano, is to have a sustainable and secure Cardano blockchain. I can see (end expect) a future when SPOs are no longer needed to achieve this.

THE GOOD

- What is the main function of the RSS reward calculation?

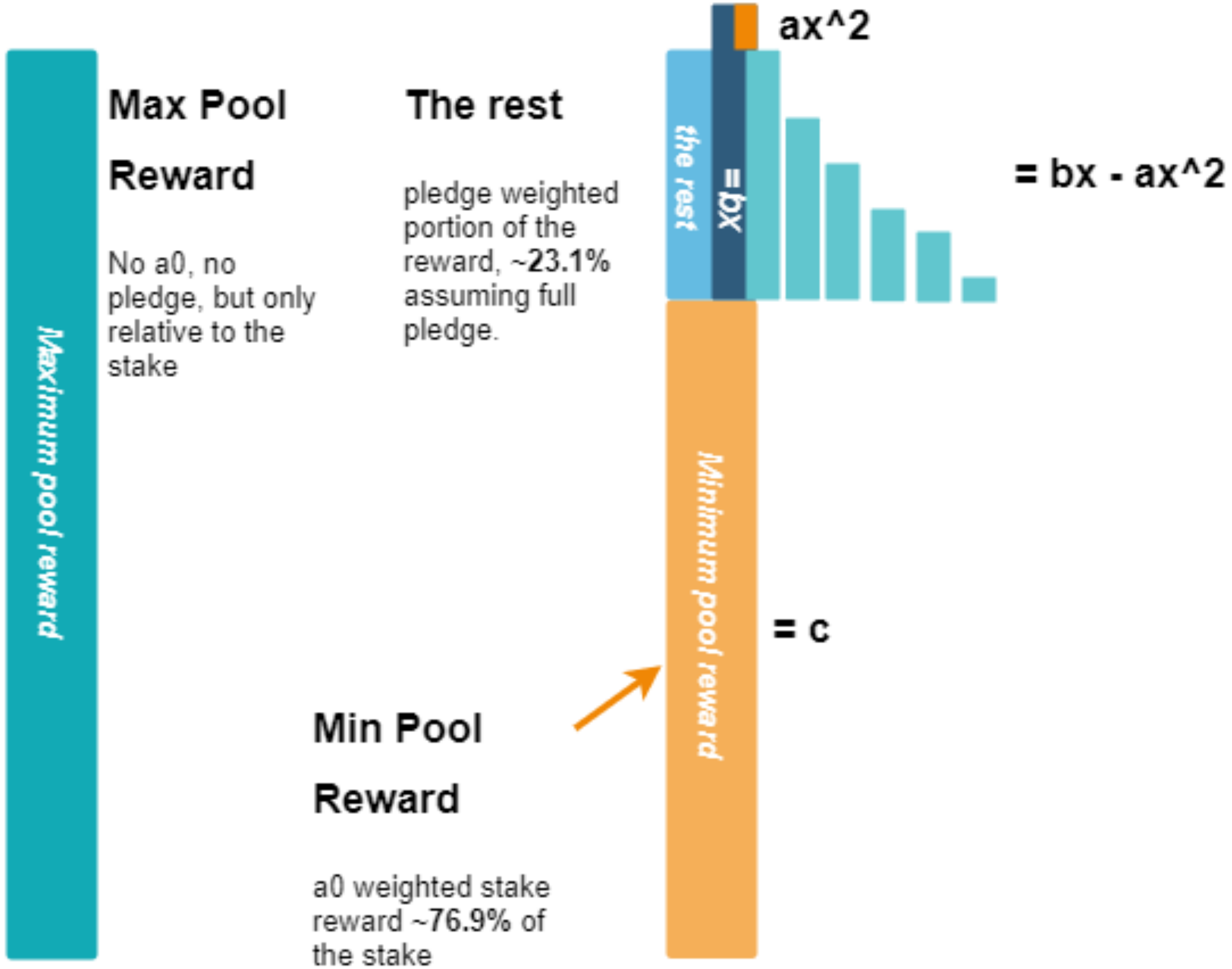
$$f(s,\sigma) := \frac{R}{1+a_0} \cdot \left(\sigma' + s' \cdot a_0 \cdot \frac{\sigma' - s' \frac{z_0 - \sigma'}{z_0}}{z_0} \right) .$$

THE GOOD

- What is the main function of the RSS reward calculation?

Simply, just calculate a

pool's epoch reward.



Max Pool Reward

No a0, no pledge, but only relative to the stake

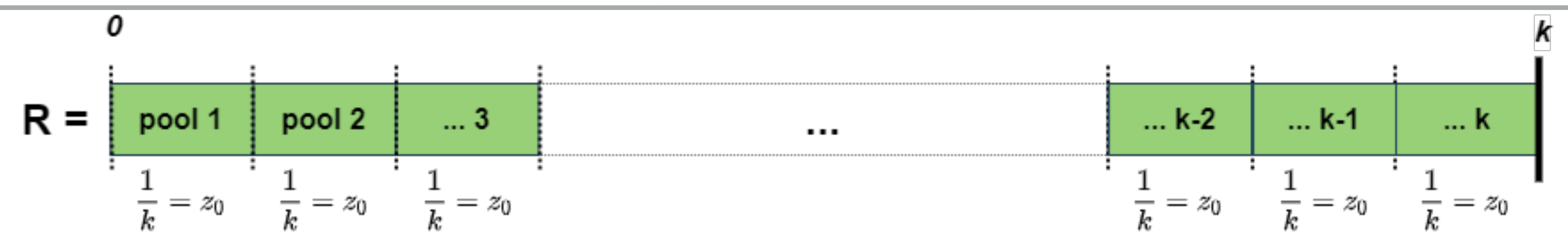
The rest

pledge weighted portion of the reward, ~23.1% assuming full pledge.

Min Pool Reward

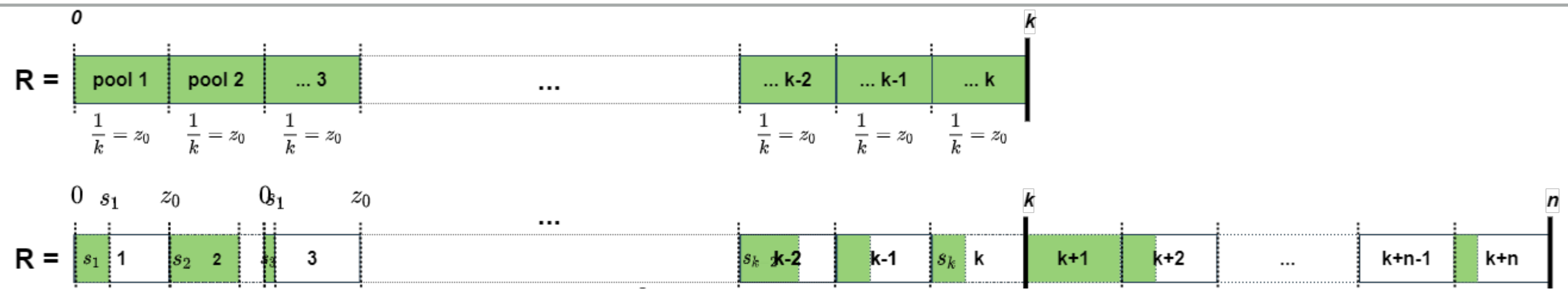
a0 weighted stake reward ~76.9% of the stake

THE GOOD



In ideal case, we would have ***k*** number of fully saturated fully pledged pool

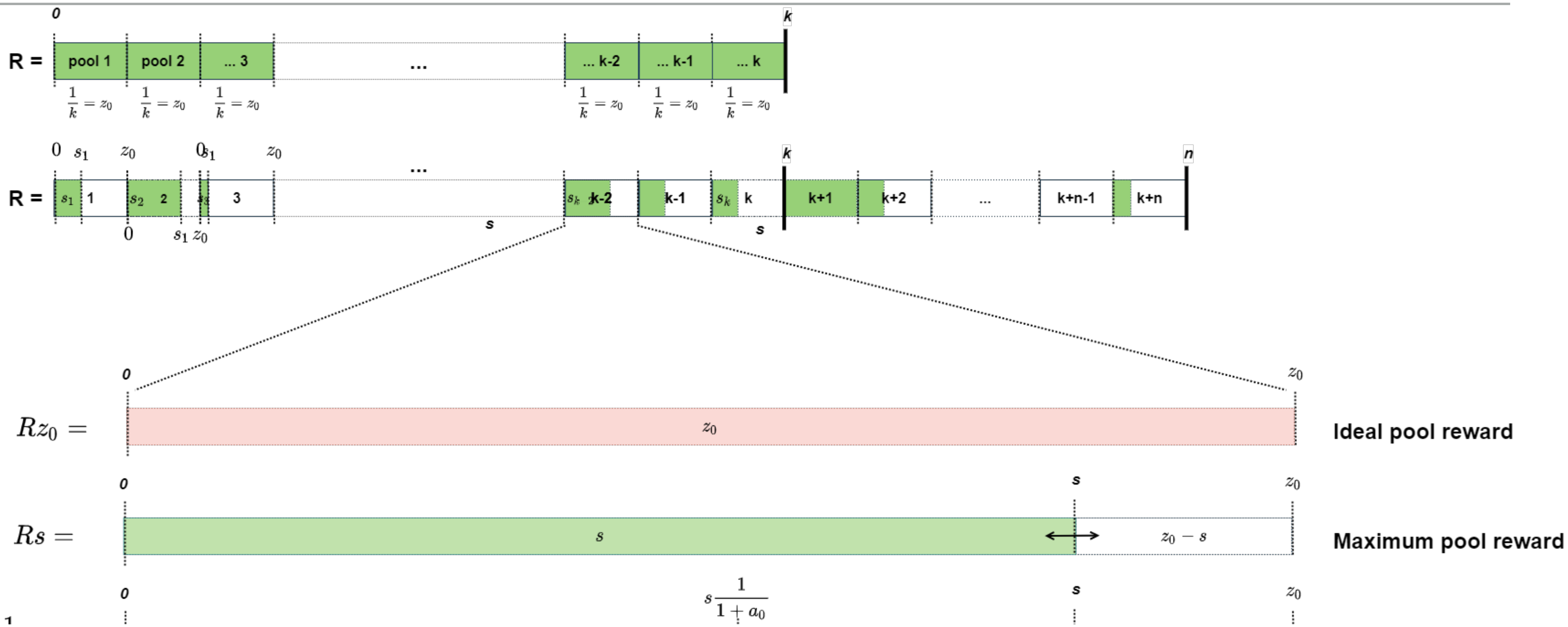
THE GOOD



In real life, we have ***k*** + ***n*** number of pools.

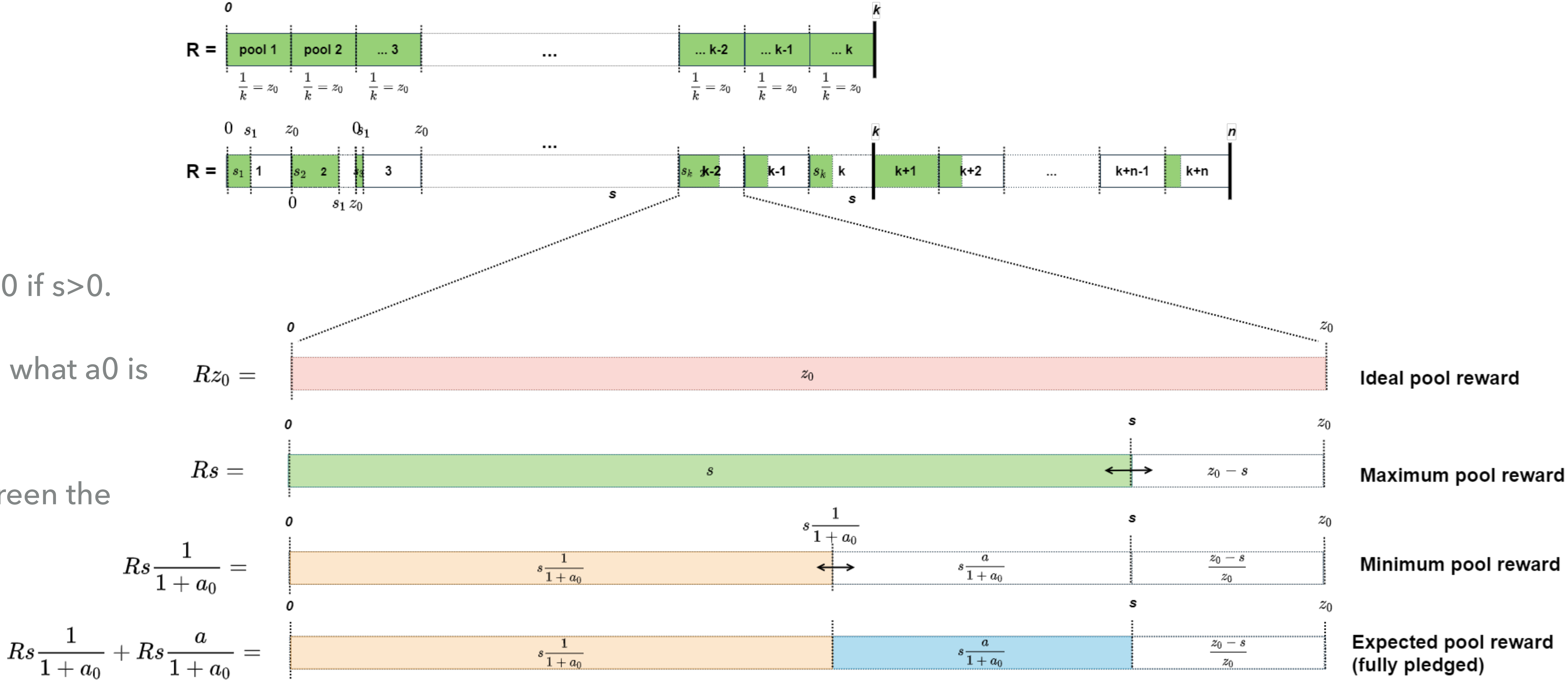
Note: The total delegated stakes cannot exceed the Total supply (currently ~31.9bn)

THE GOOD



Ideal and **maximum** pool reward

THE GOOD



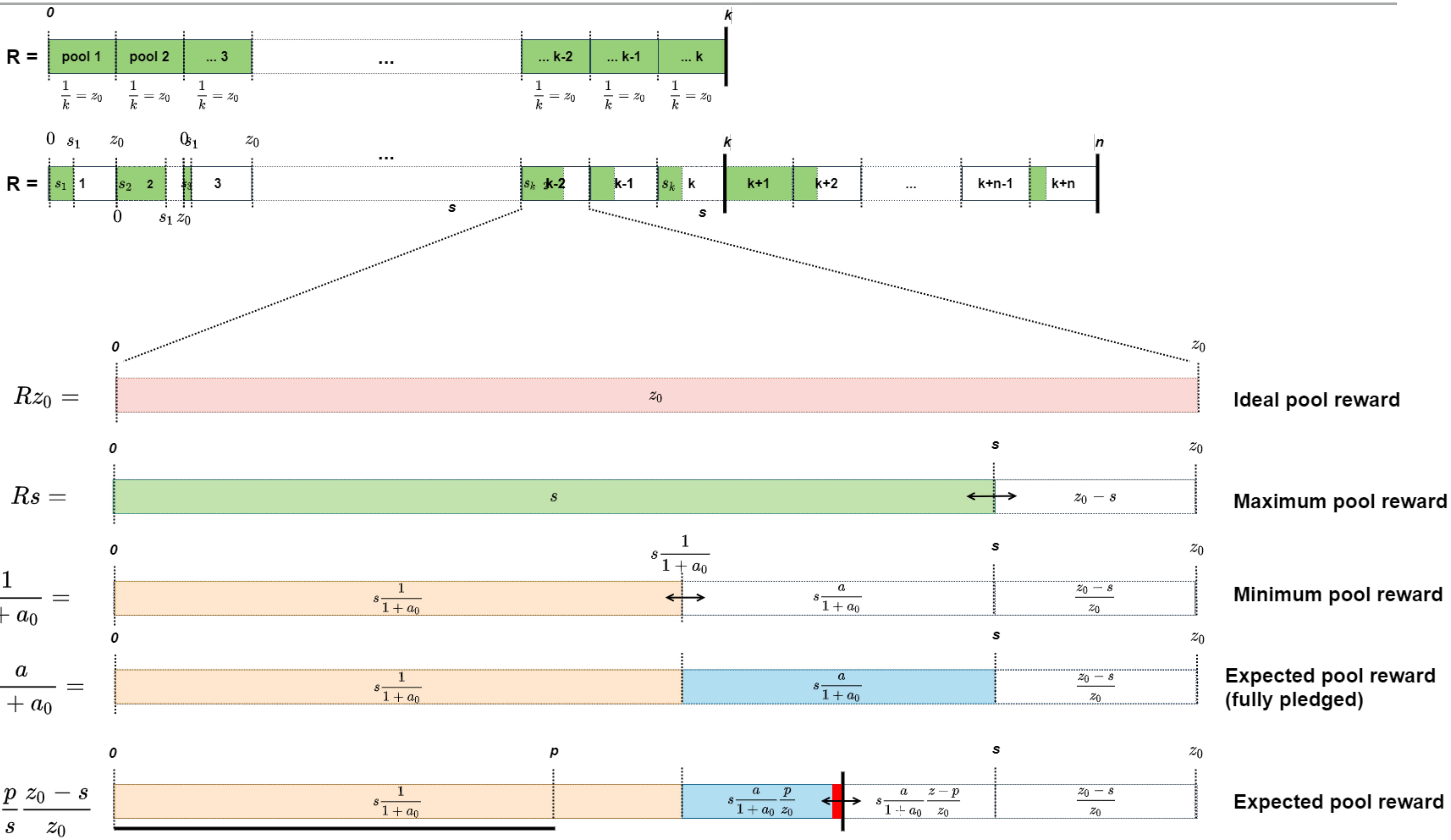
- Min stake cannot reach 0 if s>0.
- Orange is the base with what a0 is calculated.
- i.e. orange * (1+a0) = green the 100%

$$R\frac{1}{1+a_0}s = 0.769Rs$$

Minimum pool reward

THE GOOD

Expected reward, relative to the pledge. I.e. **min** + **pledge** weighted part



$$Rz_0 =$$

$$Rs =$$

$$Rs\frac{1}{1+a_0} =$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0} =$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\frac{p}{s}\frac{z_0 - s}{z_0}$$

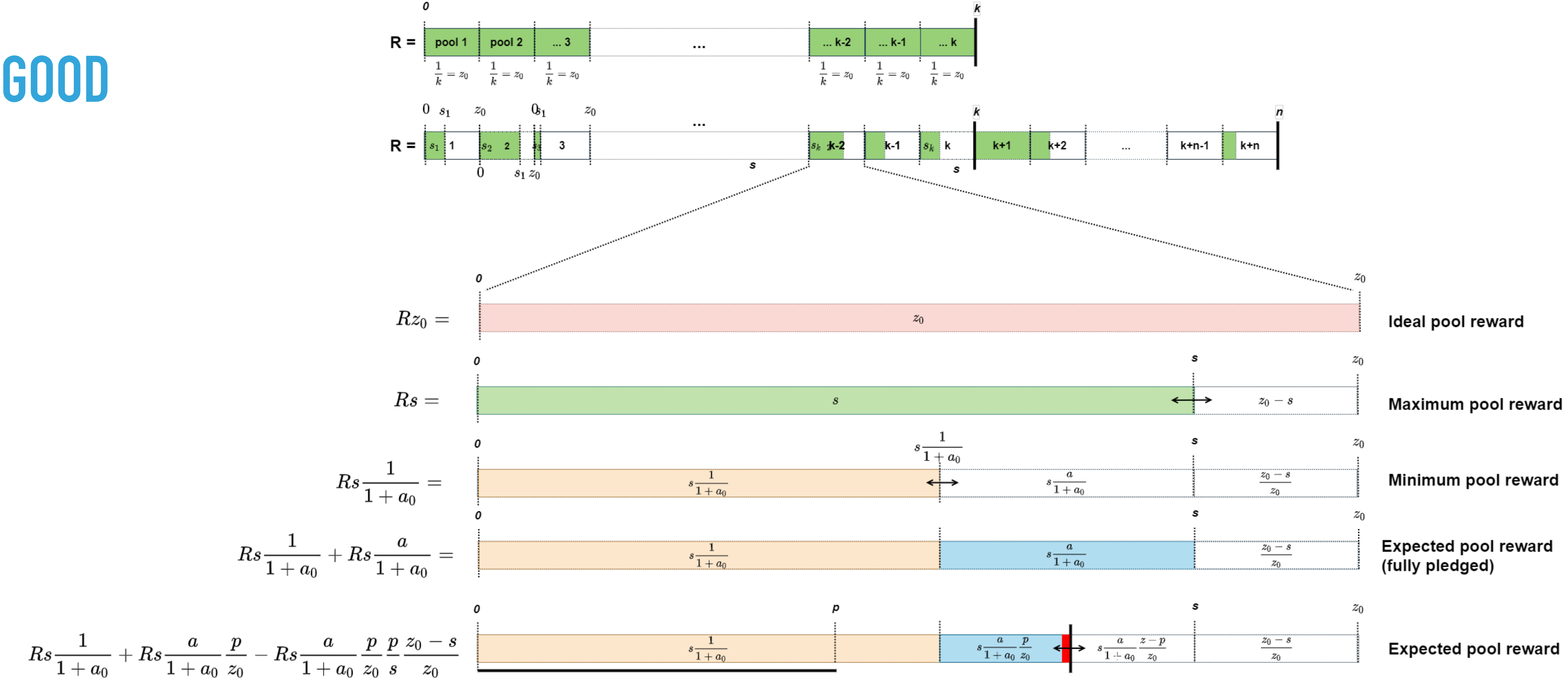
$$Rs\frac{1}{\ } - Rs\frac{a}{\ }\frac{p}{\ }\left(1 - \frac{p}{\ }\frac{z_0 - s}{\ }\right)$$

$1 \quad a \quad p \quad a \quad p^2 \quad z_0 - s$

But pledge is relative to **s** and not to that blue box, cos pledge can be **0** or max **s**

Note: The 0 pledge is not considered here for the sake of simplicity

THE GOOD



$$Rs\frac{1}{1+a_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\left(1-\frac{p}{s}\frac{z_0-s}{z_0}\right)$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - R\frac{a}{1+a_0}\frac{p^2}{z_0}\frac{z_0-s}{z_0}$$

$$R \Big(\quad p \quad p^2 \; z_0 - s \Big)$$

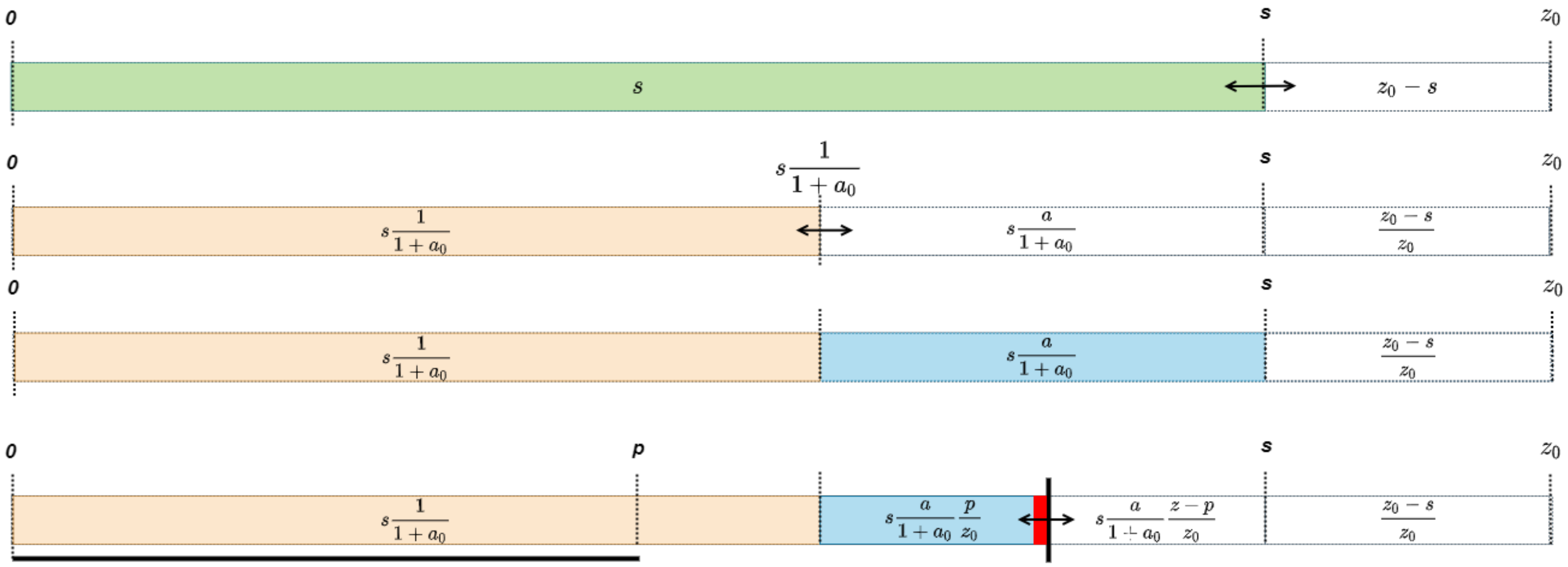
THE GOOD

$$Rs =$$

$$Rs\frac{1}{1+a_0} =$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0} =$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\frac{p}{s}\frac{z_0-s}{z_0}$$



Maximum pool reward

Minimum pool reward

Expected pool reward (fully pledged)

Expected pool reward

$$Rs\frac{1}{1+a_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\left(1-\frac{p}{s}\frac{z_0-s}{z_0}\right)$$

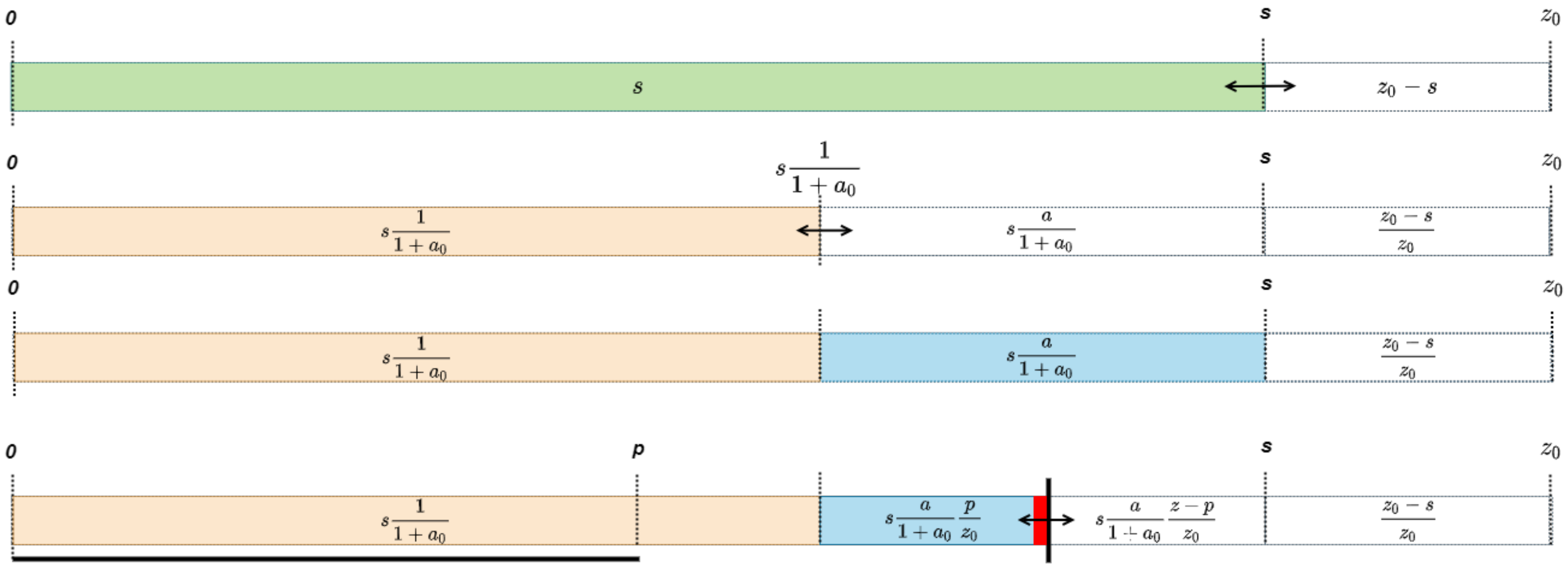
$$f(s,\sigma) := \frac{R}{1+a_0}\cdot\left(\sigma' + s'\cdot a_0\cdot\frac{\sigma' - s'\frac{z_0-\sigma'}{z_0}}{z_0}\right).$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - R\frac{a}{1+a_0}\frac{p^2}{z_0}\frac{z_0-s}{z_0}$$

$$\frac{R}{1+a_0}\left(s + sa\frac{p}{z_0} - a\frac{p^2}{z_0}\frac{z_0-s}{z_0}\right)$$

$$\frac{R}{1+a_0}\left(s + pa\frac{s-p\frac{z_0-s}{z_0}}{z_0}\right)$$

THE GOOD



$$Rs =$$

Maximum pool reward

$$Rs\frac{1}{1+a_0} =$$

Minimum pool reward

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0} =$$

Expected pool reward (fully pledged)

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\frac{p}{s}\frac{z_0-s}{z_0}$$

Expected pool reward

$$Rs\frac{1}{1+a_0} - Rs\frac{a}{1+a_0}\frac{p}{z_0}\left(1-\frac{p}{s}\frac{z_0-s}{z_0}\right)$$

$$Rs\frac{1}{1+a_0} + Rs\frac{a}{1+a_0}\frac{p}{z_0} - R\frac{a}{1+a_0}\frac{p^2}{z_0}\frac{z_0-s}{z_0}$$

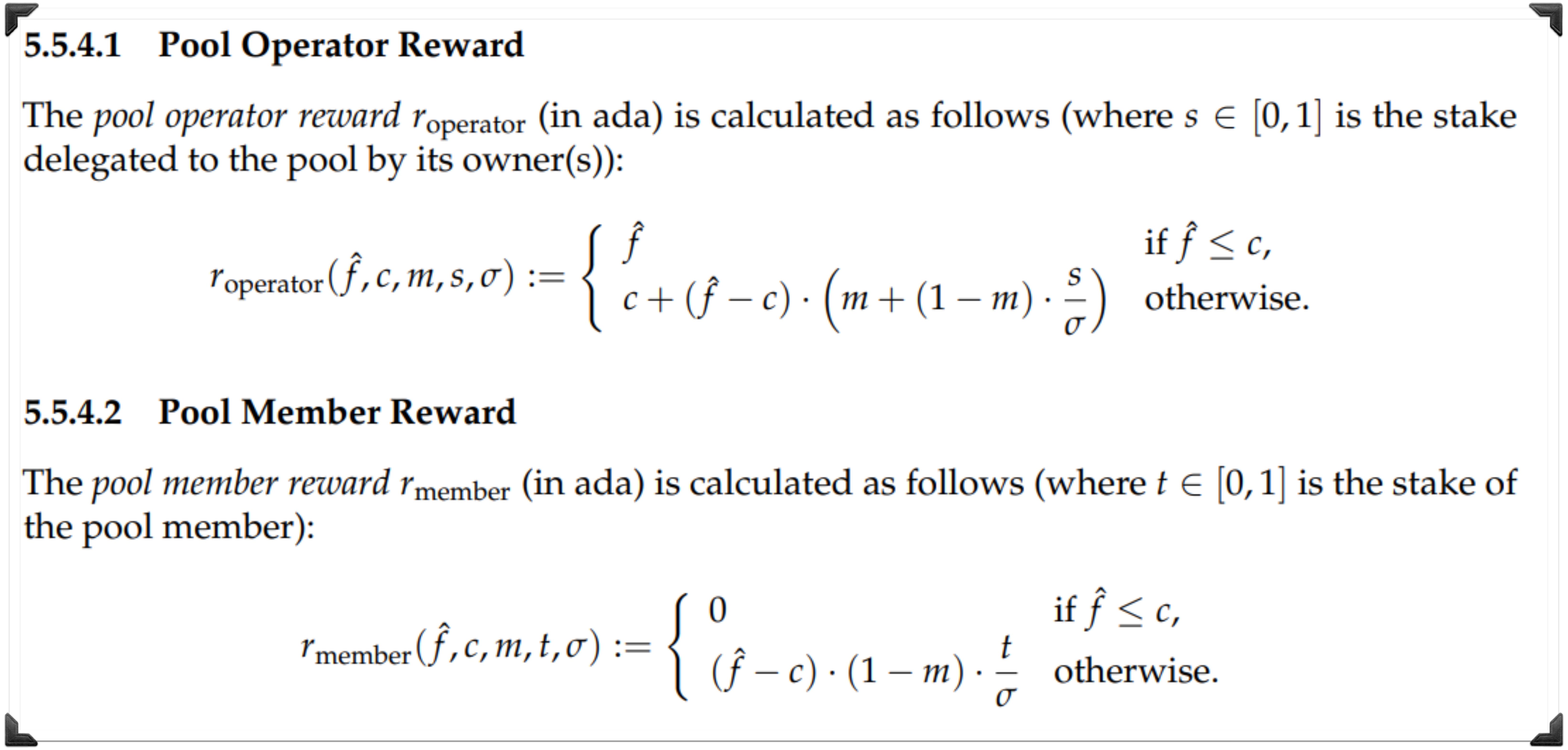
$$\frac{R}{1+a_0}\boxed{\left(s + sa\frac{p}{z_0} - a\frac{p^2}{z_0}\frac{z_0-s}{z_0}\right)}$$

$$\frac{R}{1+a_0}\left(s + pa\frac{s - p\frac{z_0-s}{z_0}}{z_0}\right)$$

$$f(s,\sigma) := \frac{R}{1+a_0}\cdot\left(\sigma' + s'\cdot a_0\cdot\frac{\sigma' - s'\frac{z_0-\sigma'}{z_0}}{z_0}\right).$$

THE BAD

- **What are the reward distribution functions?**
- What are their purpose?
- Introducing cost for the operator
- Introducing margin for the operator
- Split the rest between the operator and pool members



Pool Operator and Member Reward functions

5.5.4.1 Pool Operator Reward

The *pool operator reward* r_{operator} (in ada) is calculated as follows (where $s \in [0,1]$ is the stake delegated to the pool by its owner(s)):

$$r_{\text{operator}}(\hat{f}, c, m, s, \sigma) := \begin{cases} \hat{f} & \text{if } \hat{f} \leq c, \\ c + (\hat{f} - c) \cdot \left(m + (1-m) \cdot \frac{s}{\sigma}\right) & \text{otherwise.} \end{cases}$$

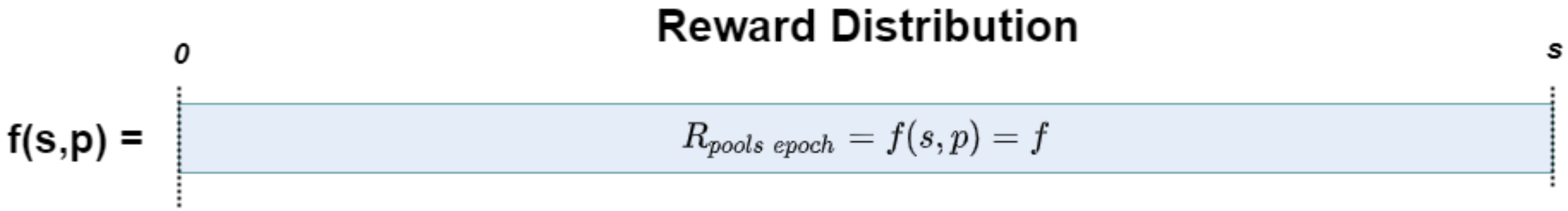
5.5.4.2 Pool Member Reward

The *pool member reward* r_{member} (in ada) is calculated as follows (where $t \in [0,1]$ is the stake of the pool member):

$$r_{\text{member}}(\hat{f}, c, m, t, \sigma) := \begin{cases} 0 & \text{if } \hat{f} \leq c, \\ (\hat{f} - c) \cdot (1-m) \cdot \frac{t}{\sigma} & \text{otherwise.} \end{cases}$$

THE BAD

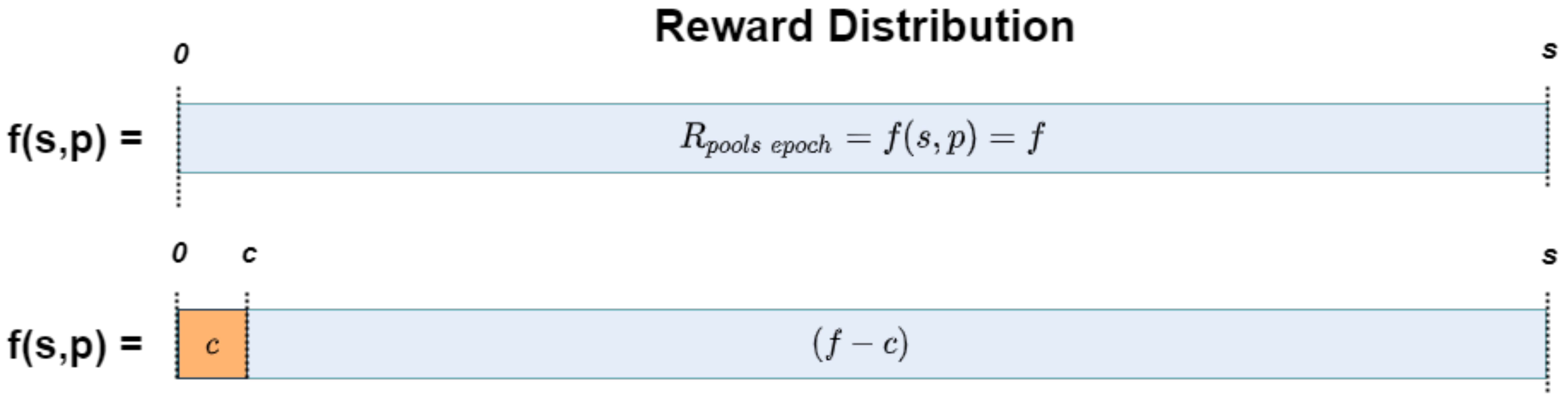
- What are the reward distribution functions?
- **What are their purpose?**
- Introducing cost for the operator
- Introducing margin for the operator
- Split the rest between the operator and pool members



Simply just split a pool's calculated epoch reward between the **operator** and **pool members**.

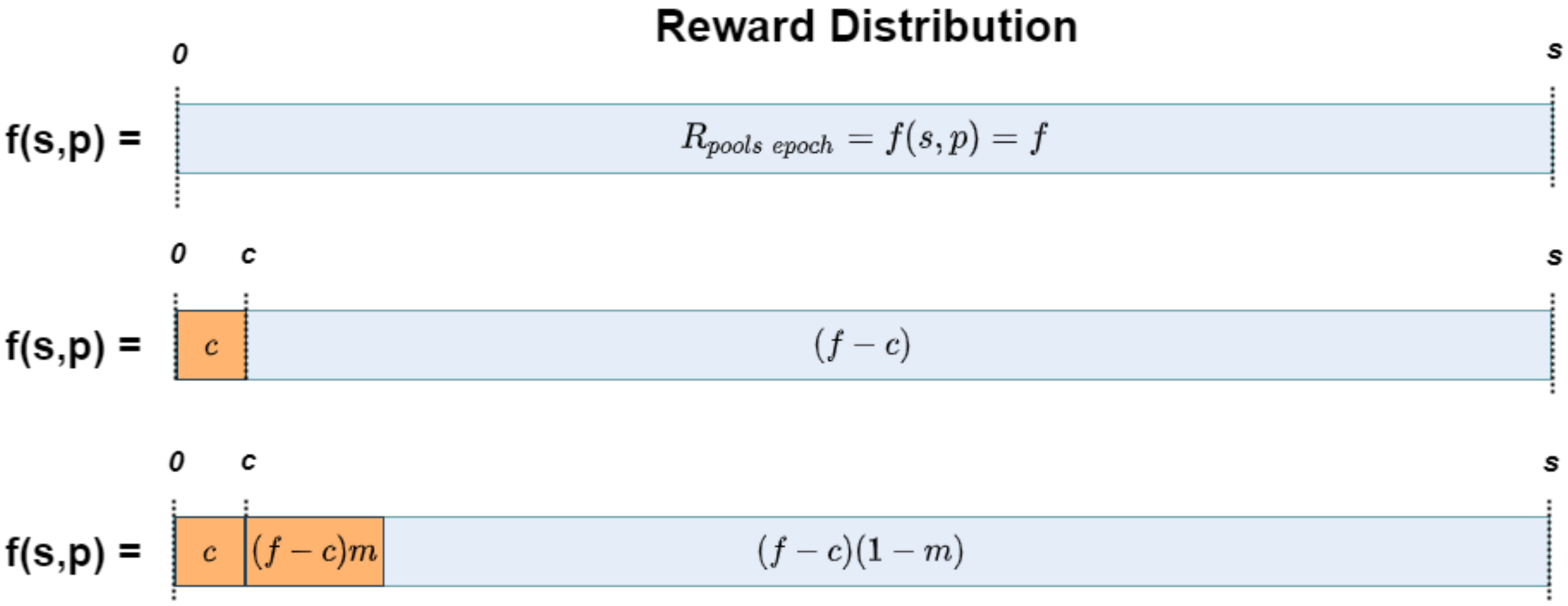
THE BAD

- What are the reward distribution functions?
- What are their purpose?
- **Introducing cost for the operator**
- Introducing margin for the operator
- Split the rest between the operator and pool members



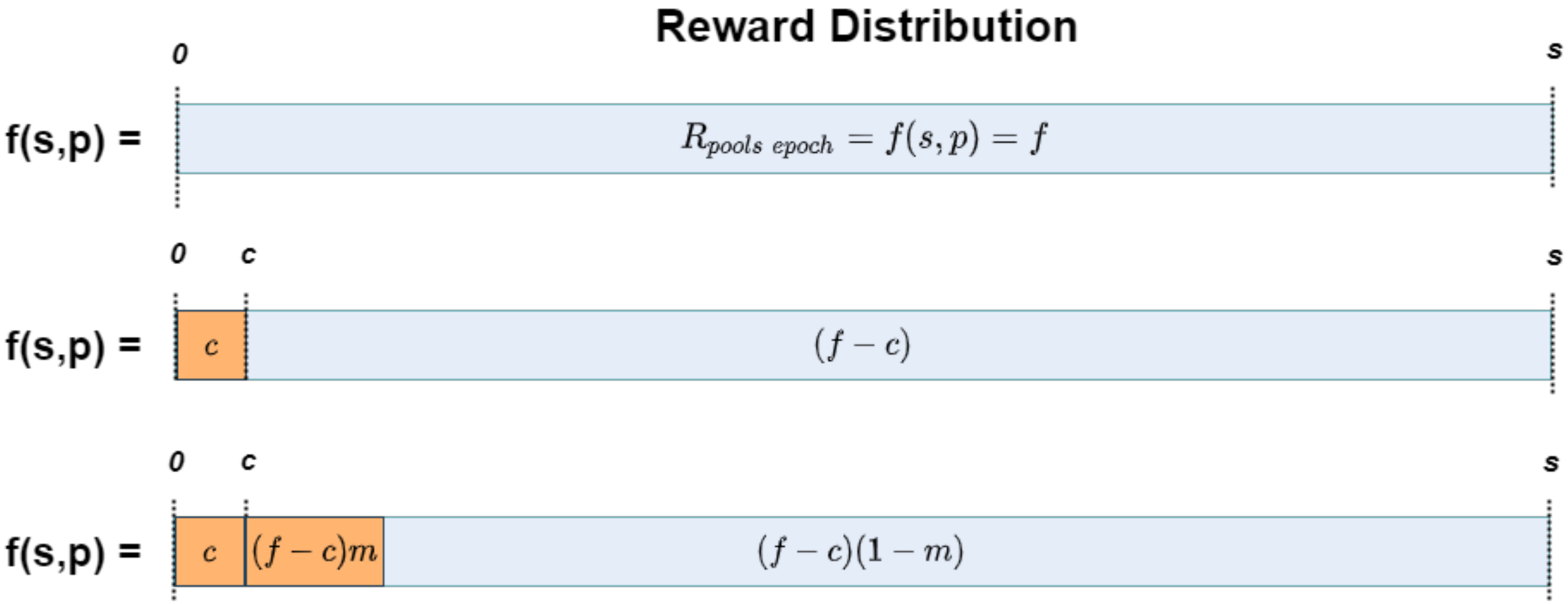
THE BAD

- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- **Introducing margin for the operator**
- Split the rest between the operator and pool members



THE BAD

- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- **Introducing margin for the operator**
- Split the rest between the operator and pool members



Reward Distribution

0 s

f(s,p) = $R_{pools\ epoch} = f(s,p) = f$

0 c s

f(s,p) = c | $(f-c)$

0 c s

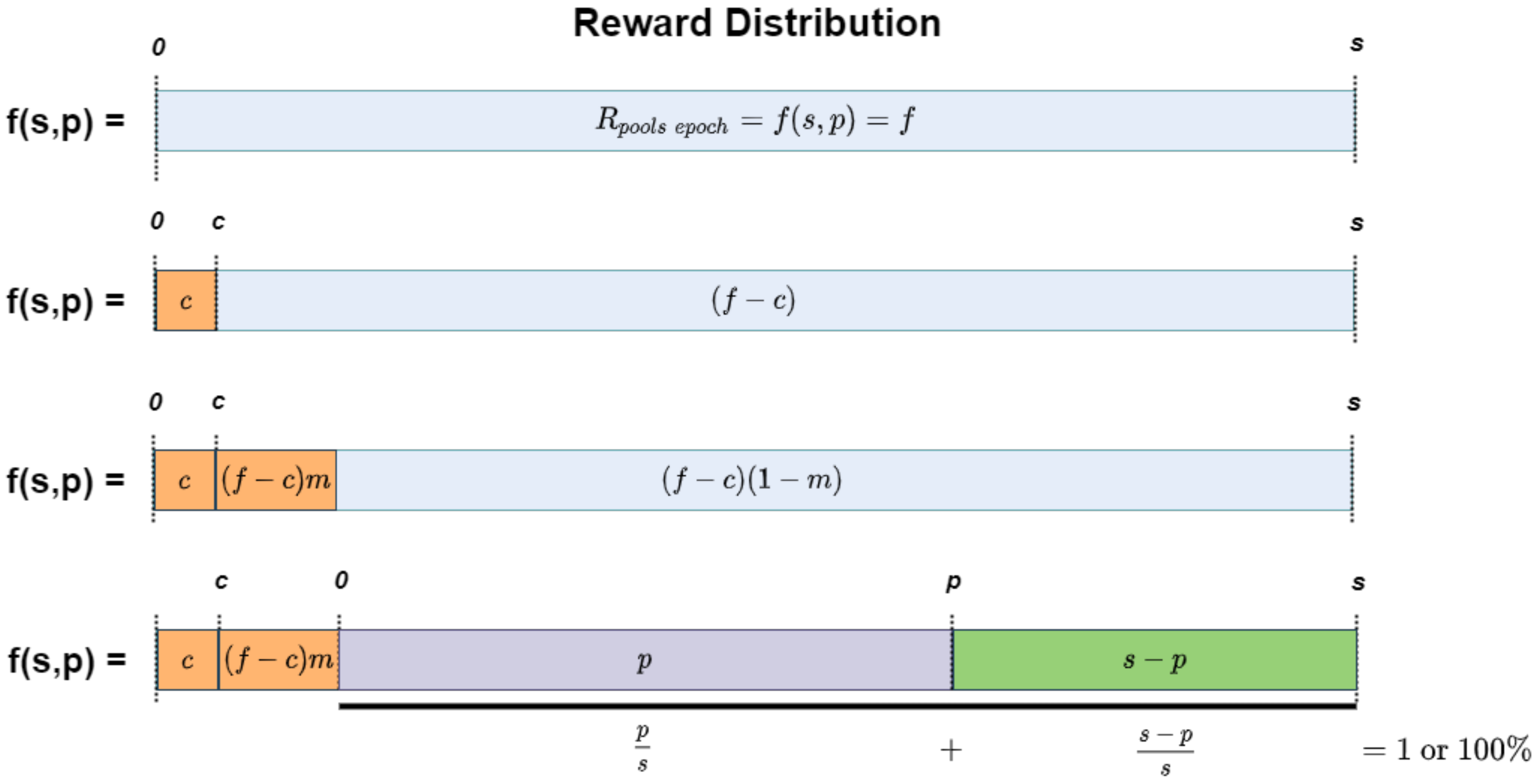
f(s,p) = c | $(f-c)m$ | $(f-c)(1-m)$

c+(f-c)m

(f-c) => (f-c)*1=(f-c)*(m+1-m) =>(f-c)m+(f-c)(1-m)

THE BAD

- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- Introducing margin for the operator
- **Split the rest between the operator and pool members**



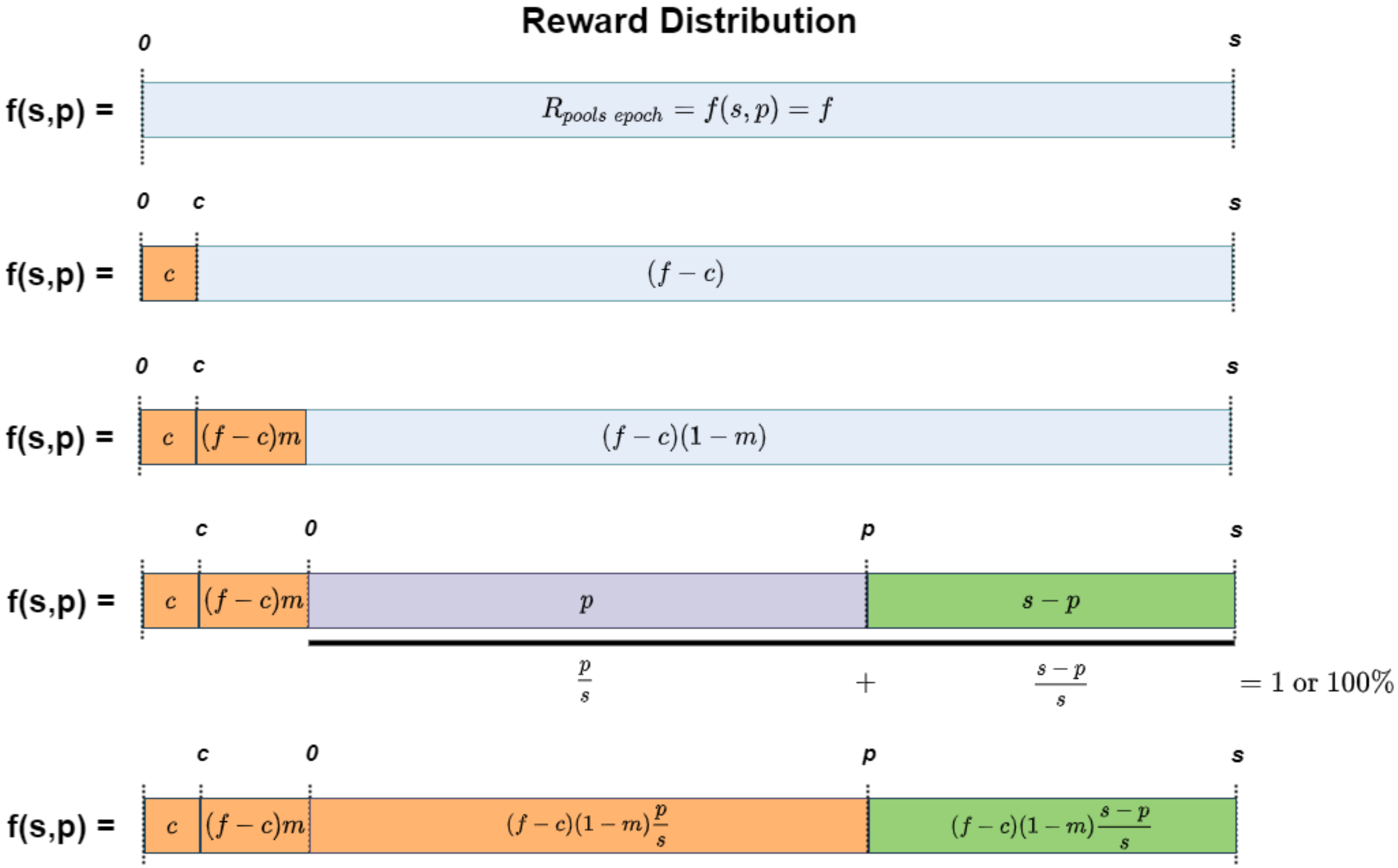
(s-p), what I use here for the sake of simplicity, means all members of that pool.

In the RSS, they use **t** for representing the individual member's reward, which is simply a portion of that whole **(s-p)**.

This, can be interpreted as a slider inside the green box, for a user's delegated stake.

THE BAD

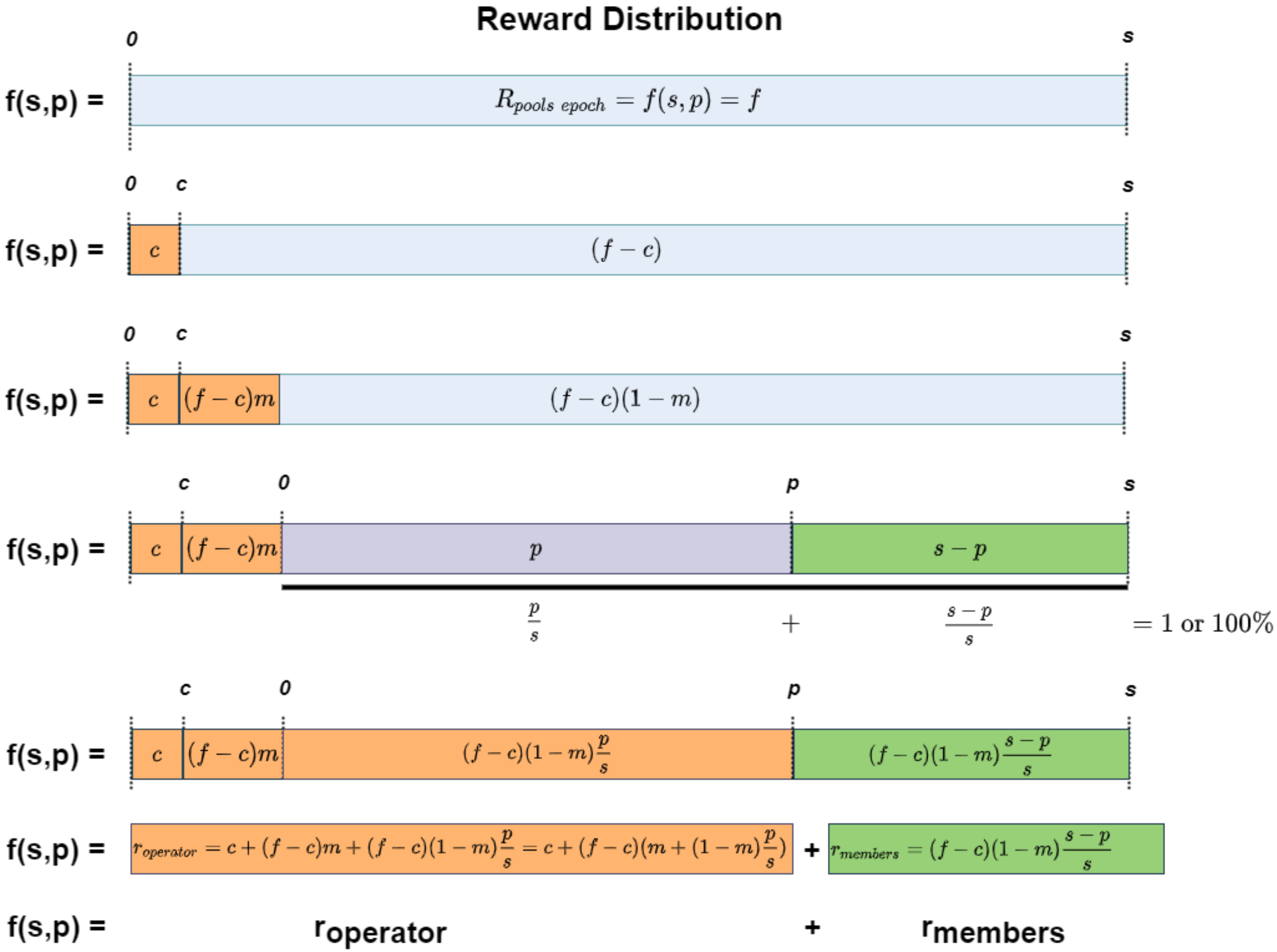
- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- Introducing margin for the operator
- **Split the rest between the operator and pool members**



Reward Distribution

THE BAD

- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- Introducing margin for the operator
- **Split the rest between the operator and pool members**

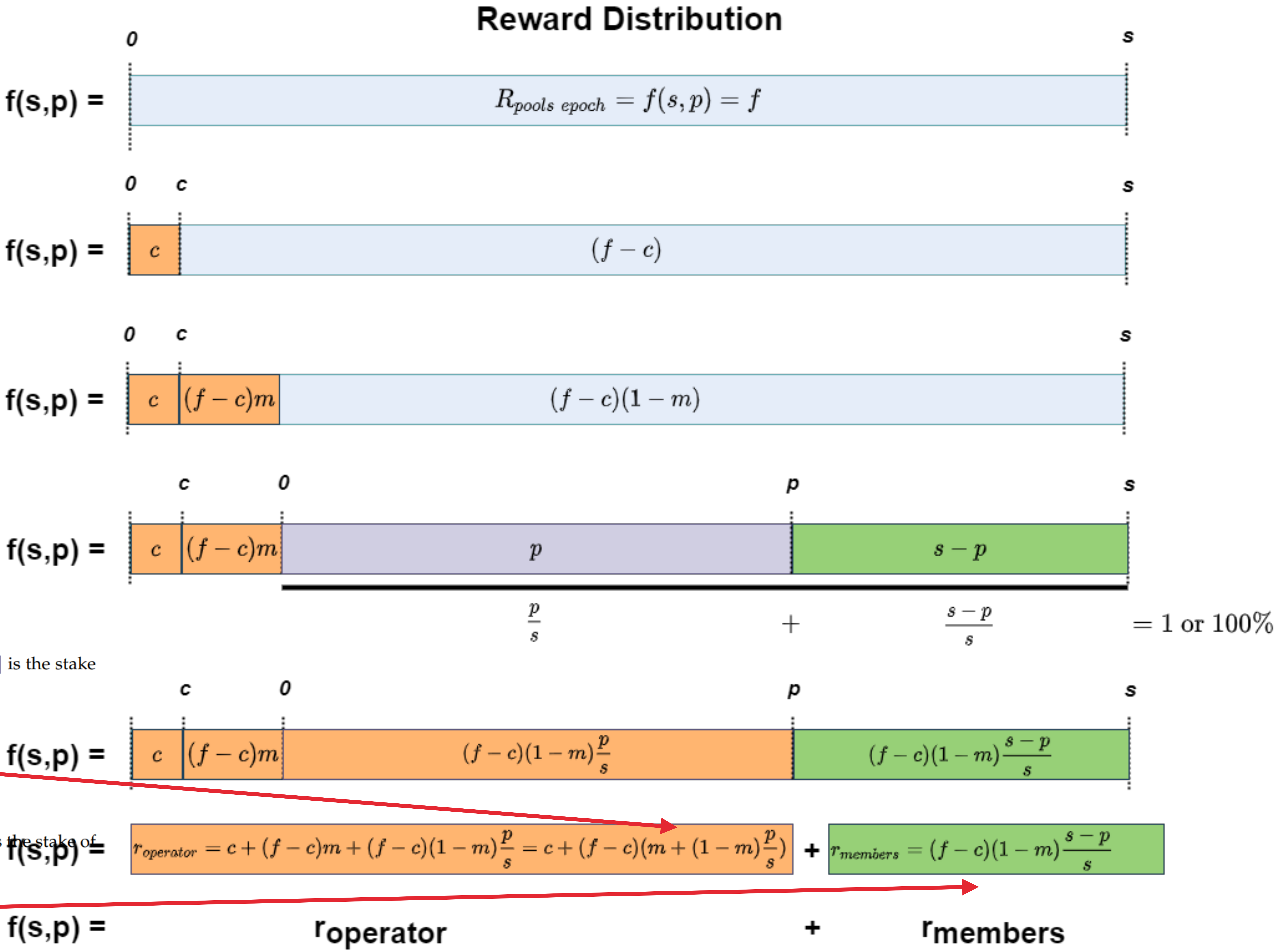


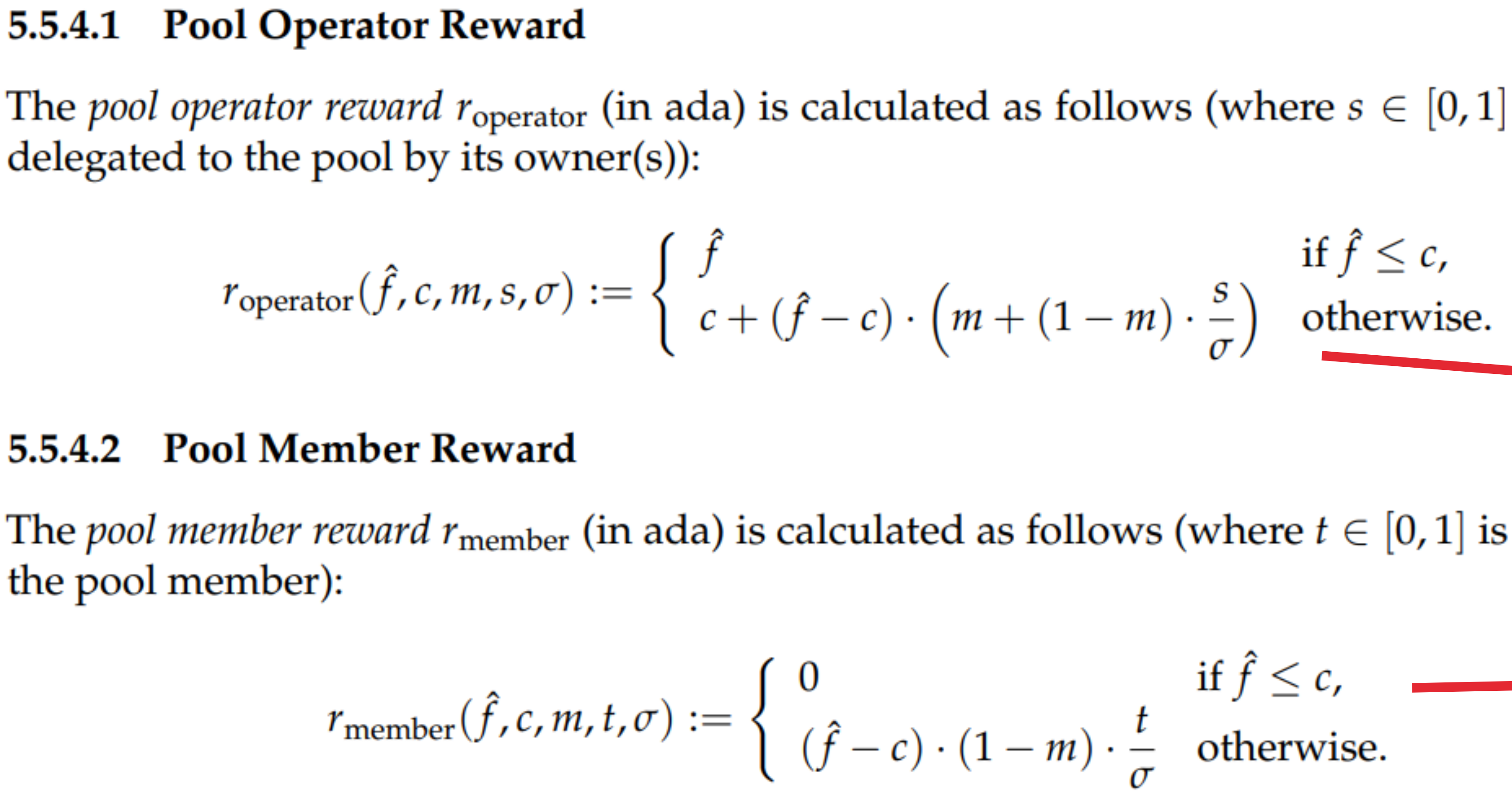
Reward Distribution

0
s
f(s,p) =
$R_{pools\ epoch} = f(s,p) = f$
0
c
s
f(s,p) =
c
$(f-c)$
0
c
s
f(s,p) =
c
$(f-c)m$
$(f-c)(1-m)$
c
0
p
s
f(s,p) =
c
$(f-c)m$
p
$s-p$
$\frac{p}{s} + \frac{s-p}{s} = 1 \text{ or } 100\%$
c
0
p
s
f(s,p) =
c
$(f-c)m$
$(f-c)(1-m)\frac{p}{s}$
$(f-c)(1-m)\frac{s-p}{s}$
f(s,p) =
$r_{operator} = c + (f-c)m + (f-c)(1-m)\frac{p}{s} = c + (f-c)(m + (1-m)\frac{p}{s})$
+
$r_{members} = (f-c)(1-m)\frac{s-p}{s}$
f(s,p) =
$r_{operator}$
+
$r_{members}$

THE BAD

- What are the reward distribution functions?
- What are their purpose?
- Introducing cost for the operator
- Introducing margin for the operator
- **Split the rest between the operator and pool members**





5.5.4.1 Pool Operator Reward

The *pool operator reward* r_{operator} (in ada) is calculated as follows (where $s \in [0,1]$ is the stake delegated to the pool by its owner(s)):

$$r_{\text{operator}}(\hat{f}, c, m, s, \sigma) := \begin{cases} \hat{f} & \text{if } \hat{f} \le c, \\ c + (\hat{f} - c) \cdot \left(m + (1-m) \cdot \frac{s}{\sigma}\right) & \text{otherwise.} \end{cases}$$

5.5.4.2 Pool Member Reward

The *pool member reward* r_{member} (in ada) is calculated as follows (where $t \in [0,1]$ is the stake of the pool member):

$$r_{\text{member}}(\hat{f}, c, m, t, \sigma) := \begin{cases} 0 & \text{if } \hat{f} \le c, \\ (\hat{f} - c) \cdot (1-m) \cdot \frac{t}{\sigma} & \text{otherwise.} \end{cases}$$

AND THE UGLY

What is the issue /w the RSS and/or RSS distribution functions?

- The Cardano protocol should offer Sybil protection to prevent:

1. having only very few nr. of pools that controls total delegated stakes.
2. generating as many pools as the adversaries can.
3. power accumulation of multiple pools owned by one entity.

We can have consensus that the 1st and 2nd points are achieved by

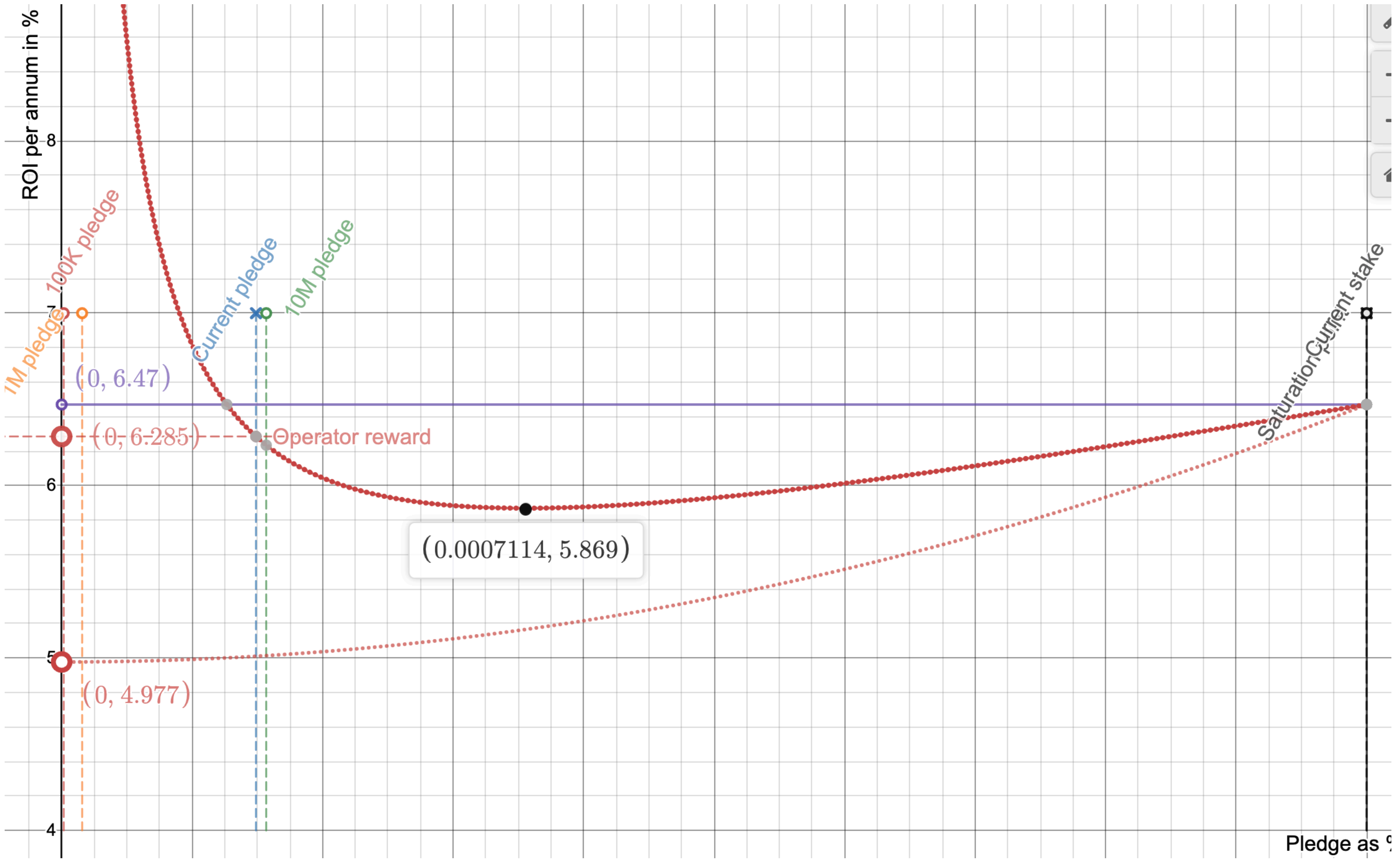
- RSS, the k and the a0 parameters (i.e. oversaturation) and

- the fact that Cardano blockchain is based on the Cardano's intrinsic value a.k.a **ADA** which has limited supply.
- deposit etc.

But, the 3rd one is tricky

AND THE UGLY

- **What is the issue /w the RSS and/or RSS distribution functions?**

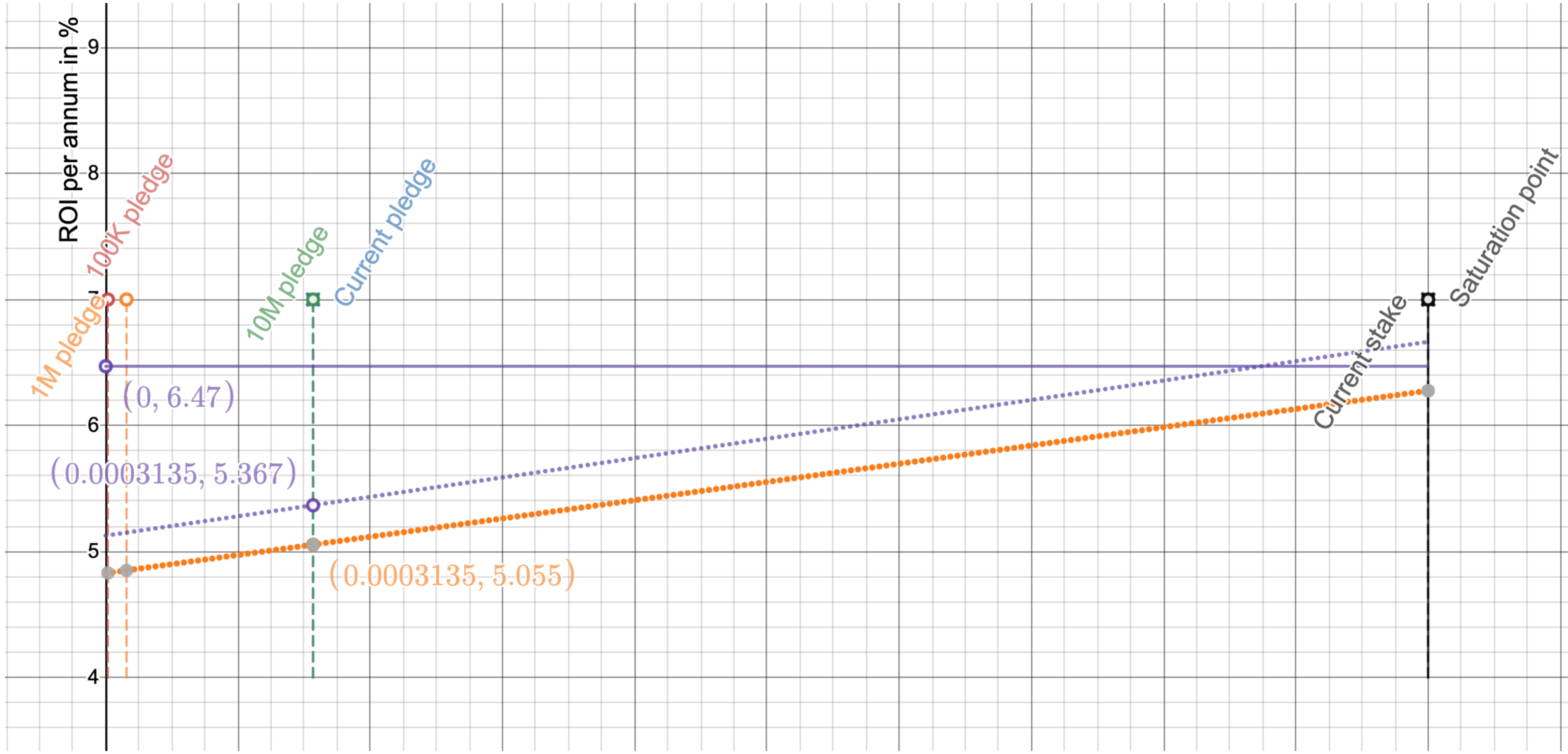


DISCUSSION

- **Is there any solution at all?**
- Simple distribution
- Shawn's CIP proposal
- Other proposals?

DISCUSSION

- Is there any solution at all?
- **Simple distribution**
- Shawn's CIP proposal
- Other proposals?



f(s,p)(1-m)

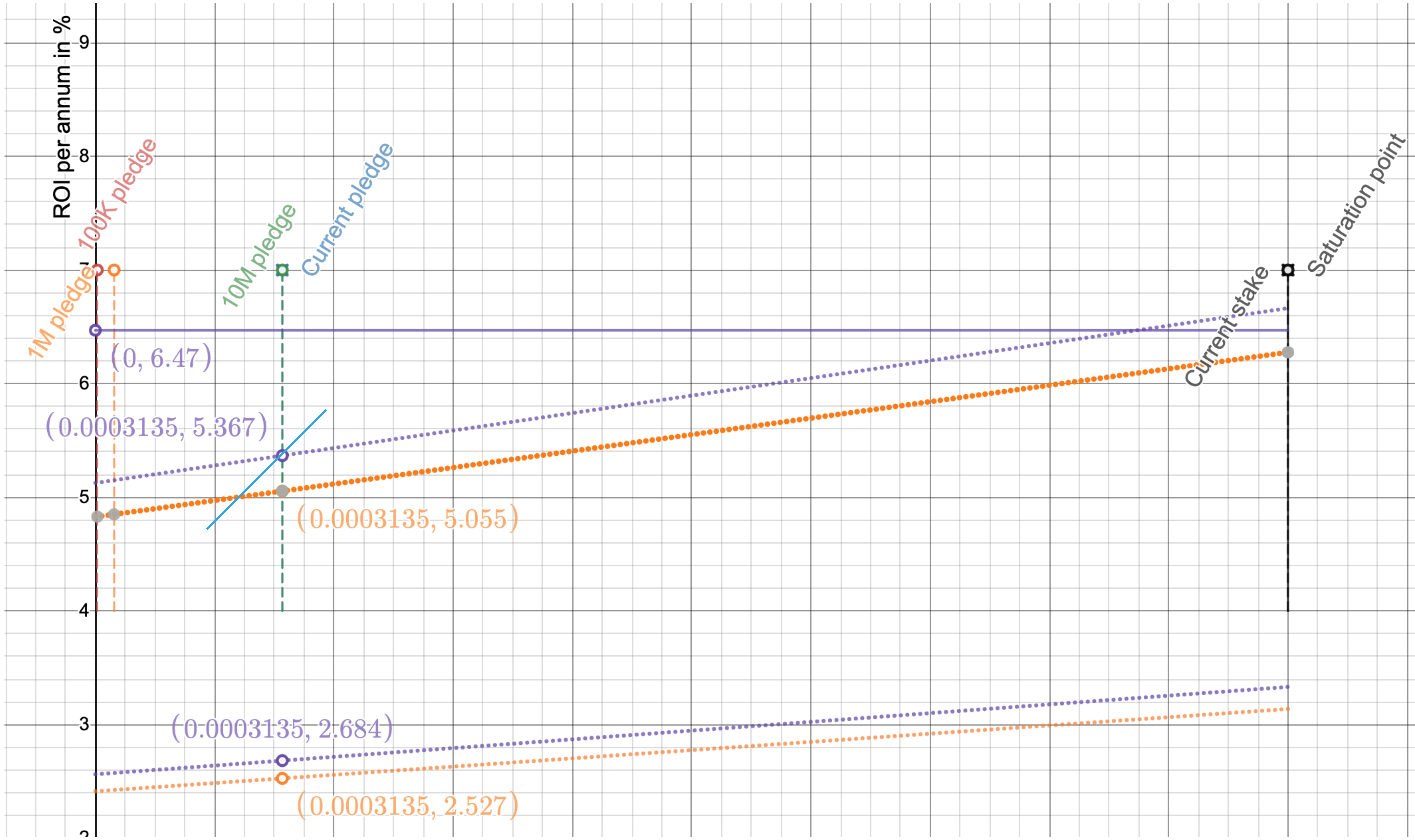
f(s,p)(1+m)

DISCUSSION

- Is there any solution at all?
- **Simple distribution**
- Shawn's CIP proposal
- Other proposals?

$$f(s,p)(1-m)$$

$$f(s,p)(1+m)$$

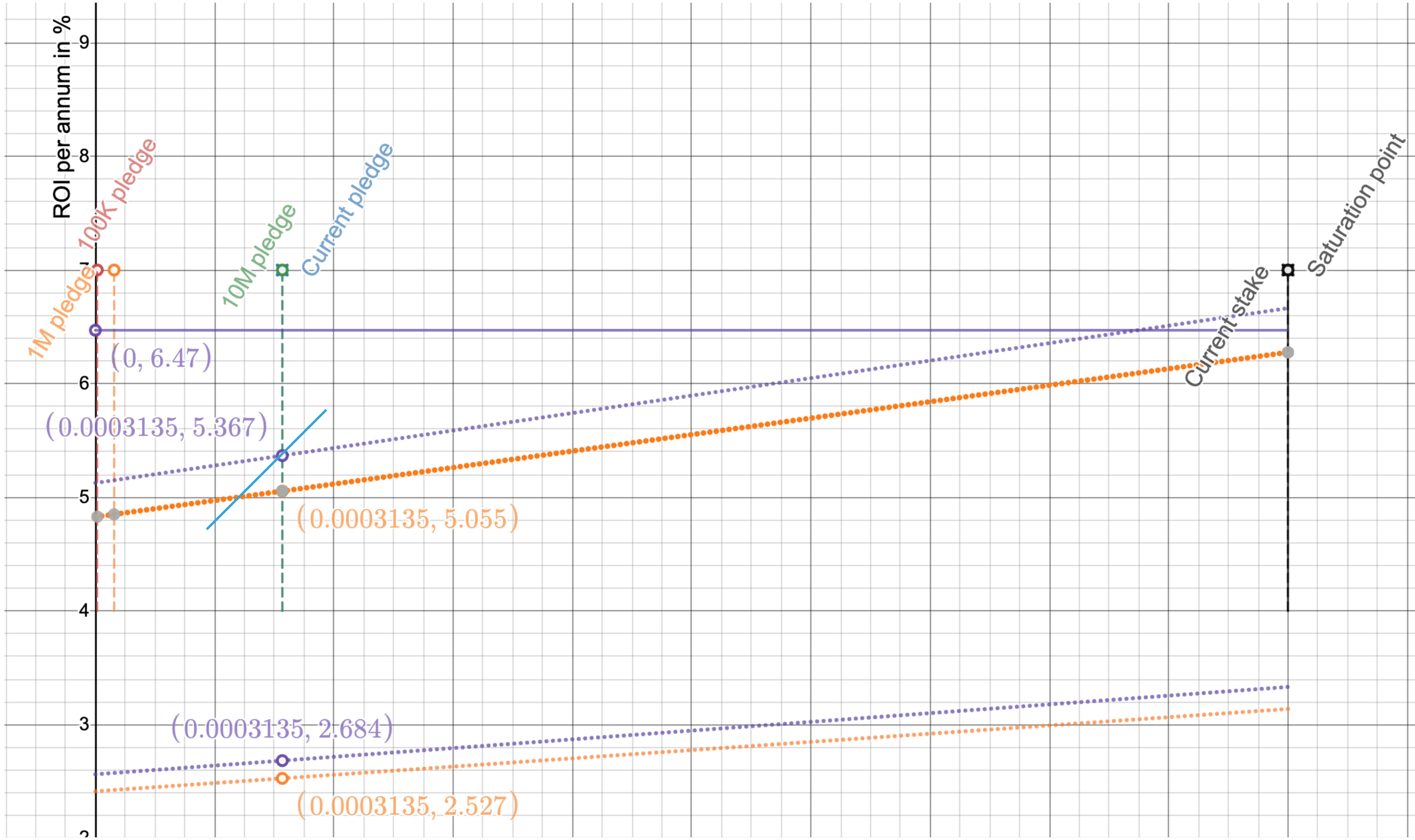


$$f(s,p) = f(s,p)(1-m) + f(s,p)(1+m)$$

$$= f(s,p)(1-m+1+m)=2f(s,p)$$

DISCUSSION

- Is there any solution at all?
- **Simple distribution**
- Shawn's CIP proposal
- Other proposals?



$$f(s,p)(1-m)$$

$$f(s,p)(1+m)$$

$$f(s,p) = f(s,p)(1-m) + f(s,p)(1+m)$$

$$= f(s,p)(1-m+1+m) = \cancel{2f(s,p)}$$

DISCUSSION

- Is there any solution at all?
- Simple distribution
- **Shawn's CIP proposal**
- Other proposals?

AND..... FINALLY

THANKS

REFERENCES

- The Reward Sharing Schemes for Stake Pools: https://arxiv.org/ftp/arxiv/papers/1807/1807.11218.pdf
- Design Specification for Delegation and Incentives in Cardano: https://hydra.iohk.io/job/Cardano/cardano-ledger-specs/delegationDesignSpec/latest-finished
- A Formal Specification of the Cardano Ledger: https://hydra.iohk.io/job/Cardano/cardano-ledger-specs/shelleyLedgerSpec/latest-finished/download/1