I think there's an interesting discussion to be had about the various strategies that plasma chain clients should take when dealing with things like challenges or exits. I haven't seen a ton

of discussion about this yet - a great thread was started here in the context of withdrawals, but I'm very curious to see how client developers are already handling this.

Motivation

To first motivate why you should care about this topic, let's look at a naive plasma implementation. A client developer might choose to have a piece of software that always watches the plasma/root chains and automatically challenges if something goes wrong. This sounds pretty good - as soon as someone tries to start an invalid exit, someone will respond with a challenge.

Unfortunately, this strategy of immediately challenging becomes problematic when more than a few clients are actively watching/challenging. Only one person can actually win the challenge, but all of the clients are submitting challenges at the same time! One of these challenges will go through, the rest will throw and burn a bunch of gas.

There's also similar discussion necessary about exactly how long to wait before bailing on a plasma chain that isn't publishing the contents of a block. It's not unlikely that a plasma chain operator will eventually submit a block and then somehow crash/go offline unexpectedly, even for a few hours. What strategy should clients use to decide when to exit? It seems like there's inherently a social component here too - if a trustworthy operator publicly claims the downtime is accidental, I might be inclined to believe it. @MihailoBjelic started a thread about this in a post here.

Coming up with strategies

We obviously need a better strategy for this - it generally seems like interesting design space. In theory we could allow clients to select their own strategy, but unfortunately my guess is that most clients will simply follow the default strategy (also it's a UX nightmare). So we'll probably have to come up with a solid strategy to ship along with clients.

I'll note here that this might be a totally solved problem, ideally there's some provably optimal strategy that statistically burns the least gas possible. I'm sure we could do some simulations to get initial results. I've heard suggestions that this could be similar to traffic control problems, which already have some game-theoretic thinking behind them. This seems like a pretty simple game and I'd be surprised if someone hasn't explored it in a different context.

One simple strategy might be to have clients wait a random amount of time before challenging, based on their best estimate to the number of challenging clients in the network. This probably works pretty effectively at first, but gets worse and worse as the number of clients increases. It's also worth noting that not all clients are required to challenge, so the total number of clients may not be an accurate indicator. I'm also guessing that we should weight the % chance that a client challenges at any point in time more heavily as the number of invalid exits increases and as an invalid exit gets closer and closer to finalizing.

All of this is purely conjecture and I haven't spent nearly enough time thinking about it in front of a whiteboard, but it seems important that client developers be aware of this problem!

Some questions

1. How are clients currently handling this?
2. Has anyone explored different strategies already?

Useful things to do

1. A simulation of various strategies for clients who want to exit from a plasma chain.
2. A simulation of various strategies for clients who want to challenge an invalid exit.
3. Seeing if there's more game theoretic research out there that could help us.

I'm going to spend some time working on (2) here and try to optimize on total gas wasted. Help/suggestions would be much appreciated!