Really great and thorough post Barnabe! A lot to react to here, so I'll try to do this in chunks and start first maybe with higher-level comments & questions for you (along with everyone else following this topic).

Goals of PBS

This is more so for me but I find it helpful to start with an explicit outlining of what outcomes we're trying to achieve with PBS. In my mind, a successful PBS design should consider the broader goals below. And I haven't thought through enough as to whether or not there's some "impossibility theorem" or yet another form of a "trilemma" in here:

1. Maximizing value accrual for proposers
2. Minimizing the difference in returns on capital between sophisticated & unsophisticated proposers
3. Censorship resistance
4. Minimizing external, "off-chain" dependencies
5. Minimizing incremental protocol complexity

Summary of PEPC as you envision it

And here's my attempt at a tl;dr of your tl;dr

1. Use an EigenLayer-like system for enforcing certain rules and behaviors in the builder market
2. Open question of whether to use optimistic or validity proofs
3. The data availability problem rears its ugly head yet again

Some questions that popped into my mind

- Until we have a strong consensus on what outcomes we're trying to optimize for, it's difficult I think to react to specific proposals or ideas around mechanism design?
- What EIPs would this type of system depend

on?

- What EIPs would be nice to have

and make the implementation of PEPC far easier / more elegant? [Initial thought is that 4337 (AA) might be somewhat helpful here]

- Should something like EigenLayer be enshrined or not? Essentially the same question people have asked around other fairly dominant "side-car" pieces of software or middleware whether it be scaling solutions, liquid staking, and/or MEV relays
- What are some low effort / low cost ways we can simulate different types of mechanism designs here? Or "do it live" but in far lower stakes environments (testnets, sidechains, etc)?