[

Screenshot 2024-04-25 at 10.49.52

1622×1606 128 KB

](https://ethresear.ch/uploads/default/original/3X/1/d/1d5c2ab8b7b15232b55c806bcfbb744f02dab255.jpeg)

By @Julian, @barnabe and @soispoke

Validators are the most decentralized set of participants at any level of the Ethereum protocol and infrastructure. This is a design goal of Ethereum because only then can the protocol leverage this decentralization to obtain protocol resilience:

- As consensus service participants, a decentralized set of validators ensures resilience against correlated failures, either accidental (e.g., due to bugs taking offline a certain share of the validator set) or malicious (e.g., a share of the validator set producing a safety fault).
- As block producers, a decentralized set of validators creates resilience against extractive cartels, which may be looking for undue rents, e.g., via censoring or re-ordering.

This post investigates the second role of validators and their duties as block producers. First, we explore how the protocol can leverage the decentralized validator set to uphold a concept we call chain neutrality. Then, we discuss how preconfirmations and PEPC may crowd out an in-protocol inclusion list from upholding chain neutrality, thereby introducing a property of inclusion lists that is the main topic of this post: uncrowdable inclusion lists. Finally, we compare unconditional inclusion lists with an instance of PEPC and investigate who should receive tips associated with transactions that are executed via the inclusion list.

# Towards recovering chain neutrality

In the past, validators (and miners before them) engaged passively

both as builders

, collecting transactions and sequencing them in blocks using simple heuristics, and as proposers

, signing the block and gossiping it over the network.

Maximal Extractable Value (MEV), the value which a validator may extract in their duties as block producer, changed the paradigm. Validators turned into passive proposers engaging with a distinct set of active builders. Active builders do not simply collect transactions and mindlessly pack them into blocks; instead, they actively optimize for the value they can extract from the block's contents. Meanwhile, validators, as passive proposers, simply listen for offers from builders who wish to build on their behalf.

Unfortunately, as a numerically much smaller set with its own profit-maximizing goals, there is no reason for builders' objectives and preferences to be aligned with validators regarding the inclusion or censorship of certain transactions or even with the preferences of network stakeholders broadly. More generally, the following principle may be put forward:

Chain neutrality:

Any pending, fee-paying transaction ought to be included if it is available and if there is room to include it on-chain.

Chain neutrality is meant to encompass principles larger than censorship resistance alone. Under censorship resistance, as we conceive of it, there is an active and targeted effort to prevent the access of some user to on-chain inclusion. However, with today's dense supply networks of intermediated relationships, some user interactions may fall by the wayside for accidental reasons simply because their transaction does not participate in increasing the welfare of the supply network intermediaries. An example is builders excluding transactions not to censor them per se, but to decrease the amount of time it would take a block to propagate over the network. In our opinion, users must always be able to "get on-chain" as long as they satisfy the two requirements spelled out in the definition of chain neutrality.

Validators as passive block producers would simply fill up their blocks from the mempool, ensuring allocation of blockspace to users who indeed satisfy the requirements of chain neutrality. Yet, with today's systems, proposers have the choice of either remaining passive ("local building") or locking themselves out entirely from participating in block production (with MEV-Boost). To preserve chain neutrality, we are then looking for ways to recover the diversity that validators embody by allowing them to provide input into block production, even when they decide to delegate building.

Inclusion lists (ILs) are a proposed method to recover validator participation in block creation. The validator produces an IL that the builder must satisfy

according to the IL model's specifications (e.g., the builder must include all transactions from the IL in their block or produce a block that cannot contain more transactions from the IL).

The first observation is that validators have not been forced into outsourcing their block building to active builders and have done so to maximize their own returns. It is then unclear why a validator would care to produce an IL binding the builders for their slot, forcing them to include transactions that the builders may wish to steer clear of. For this reason, the model of forward

inclusion lists was proposed. Forward inclusion lists are built by the validator-proposer of slot n

and constrain the block produced by the validator-proposer of slot n

- 1. From the perspective of network stakeholders, constraints on validators to respect and preserve chain neutrality, even when imposed externally by distinct validators

, appear to be legitimate.

But how can we judge the success of inclusion lists in bolstering chain neutrality beyond other potential use cases for ILs? Naively, we could consider the quantity and quality of transactions in the IL. Are the transactions reported in the IL indeed susceptible to censorship, for example, or is the IL used for other goals? This is an important question since the inclusion list is a scarce protocol product, yet other use cases lie in wait!

- Preconfirmations:

Preconfirmations are agreements between a proposer and a user regarding the conditions for inclusion of a user's transaction, e.g., within some time interval or acting upon some specific state. Preconfirmations offered by the validator-proposer hence require the validator to have some input into the block-building process. ILs look like a good place for the proposer to register their commitments, which may mean that the IL might be used for preconfirmations instead of for its purpose of ensuring chain neutrality.

- Partial block building:

Allowing multiple builders to build different parts of the block may have value. ILs could offer a path to implementing a rough protocol for such partial building commitments, using the same pattern as mev-boost (think "IL-boost" with an IL builder feeding the data to the validator-proposer). More generally, the value proposition of PEPC is to leverage the proposer's ability to make commitments regarding the organization of block building, including commitments about partial contents. ILs allow for a limited class of commitments on building a block, so is there again a conflict between ILs as neutral input surfacing and ILs as coordination gadgets toward partial block building.

Here lies the crux of the problem. Other use cases that potentially unlock a lot of value for the IL proposer can use the commitment power that the IL proposer receives from the protocol. Preconfirmations and certain PEPC-esque proposer commitments could be realized via inclusion lists. If a proposer makes more profit by constructing their IL for these use cases than for others, they likely will. When ILs are used for cases other than chain neutrality, we say that the chain neutrality use case is crowded out

. This post suggests a new design property for inclusion lists: uncrowdability

. We examine how a protocol can ensure that a protocol product becomes uncrowdable, and we apply this reasoning to the currently proposed inclusion list models.

## Uncrowdable Inclusion Lists

We informally define uncrowdable inclusion lists as inclusion lists that create more value for the inclusion list proposer when used for the purpose the protocol intended them to than when used for other purposes. Making ILs uncrowdable should be a protocol design goal. Similarly to work done on PBS, we delineate the design space of inclusion lists into the allocation rule and the market structure.

The market structure is the collection of rights and obligations that are assigned to the inclusion list. Who gets to make the inclusion list? To which block(s) does the inclusion list apply? How is the inclusion list enforced? We have seen a number of different proposed IL designs that each use a different market structure. Forward inclusion lists allow the beacon proposer of block n

to create the IL that applies to block n + 1

. Conditional ILs are enforced only if there is space for the transactions from the IL to be included in the block, whereas unconditional ILs are enforced regardless of the number of transactions in the main block body. Similarly, a cumulative, non-expiring inclusion list is also a different protocol product.

The allocation rule of inclusion lists refers to the space of commitments that the IL proposer may implement. The ideal allocation would be including in the list a set of transactions that are "censored." IL proposers may implement other allocation rules, such as outsourcing IL production to third parties, e.g., via the aforementioned hypothetical IL-boost. The protocol can create a market structure such that the allocation rule that an IL proposer chooses also achieves the goal of

upholding chain neutrality.

The protocol can use the market structure to incentivize the IL proposer to choose an allocation rule that upholds chain neutrality. Different market structures elicit different allocation rules used by IL proposers.

Arguably, conditional inclusion lists could be closer to being uncrowdable than unconditional ones because they offer fewer guarantees about the inclusion of transactions. Unconditional ILs could more easily be used for other use cases. Therefore, using the market structure of conditional ILs may incentivize the IL proposer to choose an allocation rule that upholds chain neutrality, thereby making ILs more uncrowdable.

The protocol could also make ILs uncrowdable by choosing a market structure that restricts the set of possible allocation rules. COMIS is such a mechanism. It roughly states that an inclusion list is constructed out of the inclusion sets that each member of a large committee makes. If a transaction is included in at least a certain fraction of the inclusion sets, it must be included in the final inclusion list. Such a mechanism clearly leads to a market structure that is more opinionated about its purpose, and it would be more costly for an IL proposer to use the IL for any other purpose than for chain neutrality.

There may be a more general argument to be made here. In upholding chain neutrality, the protocol makes an opinionated choice in favor of censored parties, even though this may lead to a welfare gap. A market-based solution, where the allocation rule is unconstrained, generally aims to find the welfare-maximizing solution, meaning that the MEV of one specific block is maximized. Therefore, if a protocol wants chain neutrality, maybe it must constrain the set of possible allocation rules. Potentially, this is analogous to a protocol that needs to specify an allocation rule for block construction if it wants to enforce a first-come, first-served block allocation rule.

Given this welfare gap, one might also wonder whether it is bad for the protocol-specified purpose to be crowded out by other use cases. However, whether this welfare gap is desirable is a governance question and not one that should be answered by markets. Moreover, other use cases, such as preconfirmations, do not require protocol products and may be satisfied by out-of-protocol proposer commitments such as PEPC-Boost or EigenLayer-based systems. Chain neutrality is hard to improve with out-of-protocol systems.

## Comparison between Inclusion Lists, PEPC, and Multiple Concurrent Block Producers

We've touched upon the possibility of some use cases crowding out others, given the design of the inclusion list market structure. Use cases embody the demand for certain features or outcomes that market participants desire to achieve. If the demand cannot be satisfied directly by some product yet can be satisfied indirectly by some other product, then even if the indirect product is not ex-ante

designed to accommodate this demand, some use cases will crowd out others. We must now understand where the demand for use cases such as preconfirmations or more general proposer commitments comes from and why inclusion lists allow for indirect expression of these use cases.

Broadly, we think of inclusion lists, preconfirmations, proposer commitments, and other phenomena as instances of a larger will to participate in block co-creation

. As an atomic unit, a block represents a big

amount of space to allocate for the expression of user preferences, via transactions. A Hayekian argument may be advanced that no single party has the knowledge necessary to make a good block and thus must rely on decentralized knowledge of a larger number of market participants. Preventing the expression of this decentralized will to include creates pent-up demand, which will seek to realize itself in any way that it can.

As an example, unconditional ILs reserve a certain amount of blockspace in block n+1

for the inclusion list, which is proposed and built by the proposer of block n

. This setup is very similar to an instance of PEPC with two partial builders. In fact, if the protocol could take delivery of both blocks separately instead of requiring one builder to bundle the inclusion list with the regular block, we would not need the "forward" property of the IL, and we would obtain exactly two parallel builders, although some transactions in the unconditional IL may already be executed in the main block body if it were appended to the end of block n+1

.

If a single funnel, such as inclusion lists, cannot adequately realize every type of demand from the market, can we look for differentiated "pipes" adapted to each type of use case that may occur? Barnabé suggests the idea of a "mixed IL": an IL built up from a)

a spot (applying to the block of the IL proposer) unconditional IL and b)

a forward conditional IL. This would allow good quality preconfirmations to be issued in the spot unconditional IL, whereas the forward conditional IL could still be used for chain neutrality.

Alternatively, we could also design a spot IL that upholds chain neutrality. By using a spot unconditional IL with COMIS, the role of the block proposer in the construction of the IL could be removed and replaced by a committee. This is a change in market structure similar to execution tickets since we question whether the beacon proposer of a slot should receive the rights to propose the inclusion list, instead of actors directly interacting with the protocol.

Furthermore, unconditional ILs that apply to slot n

but are made by another party than the beacon proposer of slot n

result in something that looks very similar tomultiple concurrent block producers. The differences and similarities between these different forms of block co-creation are currently under-explored.

## Inclusion List Tips

The current inclusion list spec says that the tips associated with the transactions in the IL are rewarded to the proposer of the block in which these transactions are executed, not to the proposer that included the transactions in the IL. In this part of the post, we argue that rewarding the tips to the IL proposer instead of the block proposer 1) creates a scheme in which tips can conditionally pay for censorship resistance, 2) may be inevitable, and 3) increases the cost of censorship. Note that if a credible mechanism to reward IL proposers existed, the protocol could reward IL proposers in different ways; it could use issuance, for example.

First, by giving tips from transactions that are executed via the IL, we can reward the IL proposer for providing a valuable service of chain neutrality. Here, we would consider a transaction executed via the IL

if the transaction is in the IL but not in the main block body of any earlier block. This is good for the proposer as it closes the welfare gap between using the IL for censorship resistance and for other purposes: the IL is more uncrowdable. Moreover, it is also good for the user because it allows them to pay for this service, conditional on the transaction not being included in any other way. This may be more difficult to do out-of-protocol.

While conditional tips for censorship resistance may be more difficult out-of-protocol, it is trivial for the inclusion list proposer to demand that any transaction that wants to be included in the IL bribe the proposer and then allow these transactions to be included in the list with 0 tips.

Finally, the cost of censorship, defined in "Censorship Resistance in On-Chain Auctions" by Elijah Fox, Mallesh Pai, and Max Resnick as the minimum cost an adversary would have to pay to censor a transaction, doubles when rewarding the IL proposer with the tip instead of the block proposer. This is because an adversary would not only need to bribe the block proposer to exclude the transaction, but it would also need to bribe the inclusion list proposer by the amount of the tip instead of by any arbitrarily small amount.

## Conclusion

To summarize, this post argues that the forward property of inclusion lists is insufficient to guarantee that inclusion lists will be used to uphold chain neutrality. We introduce the concept of uncrowdable

inclusion lists, which may explain why a certain inclusion list design will or will not achieve the protocol-specified goal of chain neutrality. Uncrowdable inclusion lists create more value for the list proposer when used for the protocol-desired goal than for any other goal. The protocol can be designed for uncrowdable inclusion lists by creating a market structure that induces a desirable allocation rule. Finally, we argue that it is beneficial for the tips associated with transactions in the inclusion list to be rewarded to the inclusion list proposer instead of to the block proposer.