A opinionated critique of the "Canonical Transaction Ordering for Bitcoin" white paper, written by Joannes Vermorel et al.

By /u/awemany, with many key thoughts and insights from Tom Zander (and likely others I failed to mentioned here)

Preface

This is a critique of https://blog.vermorel.com/pdf/canonical-tx-ordering-2018-06-12.pdf, SHA256 46b31403b83a04c372eeae09878ffa9ec70887388b14e3b7df7316fe148f4f0d.

This is not meant to deride the authors, their contributions or their paper. This is simply an attempt to bluntly state, rather in the spirit of a peer review, the problems that the author sees in the assumptions and arguments of the aforementioned document. Hints of anger of the reviewer in this document should be taken mostly as directed to himself for not seeing the issues more clearly earlier.

In form, I will use here the "quote a section and argue about it" approach that is rather common on platforms like Reddit and unusual for a review of a paper. But it appears to me that this approach is quite apt to keep context intact and allow a reader to follow the arguments more easily as well as being able to cross reference the original to check for biased quoting and similar problems on the reviewers end.

Summary / TL;DR

In the view of this critic , there's a strong lack of arguments and facts and a starry-eyed inadvertent construction of red herrings that lead to false conclusions and false support regarding the removal of the transaction order. After back-and-forth consideration, he strongly favors at least a further delay on implementing this change and likes to support this view with the hopefully convincing arguments herein.

Detailed criticism

From the introduction:

> In the following, we refer to this rule as the *topological transaction ordering* rule (TTOR). The point of the present paper is to demonstrate the superiority of the *canonical transaction ordering* rule (CTOR).
>
> Unlike many other consensus rules, it's not clear that the TTOR was ever intended as such for Bitcoin. Most likely this rule is the unintended consequence of the early implementation of the Satoshi client itself. Indeed, a naïve implementation of the block validation consists of enumerating the transactions of the block while sequentially updating the UTXO dataset. Thus, transactions are required to be topologically ordered for this implementation to work.

This completely overlooks the key reason why blocks are ordered the way they are and is the root of most of the problems with the arguments that follow from here. They are clearly NOT ordered like this because of an *accident*, they are ordered like this because it is the *natural* order. It is the order by which transactions can be generated, if one would have an omniscient view. (Note: I assume without loss of generality from here on that double spends do not occur. They can in some cases, of course, but the current data structures are efficient at detecting them, given that one of Bitcoin's key reason of being invented was to solve the double-spending problem. Because they can be detected and because Bitcoin is designed to stop them from happening, they do not occur, except for – in the big picture – relative fringe cases like the often 0-conf security problem.)

If you think of Bitcoin as an incentivized timestamping system, then all data that comes into the system naturally follows what is named TTOR by the authors (and what is named **natural order** herein). Blocks serve as structured time stamps of this data and giving a unique order to what uncertainty remains in the system due to physical laws and its consequences (propagation delays etc.).

Furthermore, to specifically address the second part of the above quote, that is:

> Indeed, a naïve implementation of the block validation consists of enumerating the transactions of the block while sequentially updating the UTXO dataset. Thus, transactions are required to be topologically ordered for this implementation to work.

Updating the UTXO set can only happen by following the partial order of transaction dependence. This is completely independent of a parallel or a serial single core implementation, it always has to happen with honoring the partial order of transactions!

Very bluntly, the paper here conflates the desire for easier parallel validation with the need to have the partial transaction order available and from this conflation, and in turn arriving at invalid results.

This is a criticism that threads through several arguments in the paper as well, as burying this fact behind a new order of transactions in a block does not make this core necessity go away! The partial order is, just like, for example, the preconditions when running a Makefile, essential to be followed when updating the UTXO set. Which still did not prevent the implementers of `make` to add a `-j` option. (Yes, the critic is aware that he algorithms employed in make would likely fail if applied to Bitcoin in a high transaction rate regime)

Now, there is a problem here with saying "*the* natural order" vs. "*a* natural order". As it is a partial order, several orderings would fulfill the partial ordering criterion of children transactions following their parent transactions. But just because the natural order can not be observed in an unique way does not mean that the partial ordering it follows is worthless. I hereby use "the natural order" to mean the order transactions are put in blocks henceforth, while emphasizing the caveat that this is (usually) one order of many that would work.

Going on with the next two sentences:

> We propose the CTOR as an alternative where transactions, within a block, are ordered by increasing *transaction identifier*[2]. We argue that the CTOR is highly desirable for a series of reasons, notably:

With CTOR being the "canonical transaction ordering rule". I like to remark here on a more social level that naming it "canonical ordering" is IMO showing the positive bias the authors have for their construct but can have the unintended consequence of unduly transporting this bias on the "it

sounds good" level to the audience. This is why I will call the by-TXID ordering more neutrally **lexicographic order** henceforth.

Then:

- Block emission is more efficient
- Block propagation is more efficient
- Software implementations are simplified
- Proofs of transaction inclusion are improved
- Opt-in locality between participants becomes possible
- Potential attack vectors are mitigated

Most of the above points become quite weak or even moot when addressing the underlying assumptions, so that should be done in more detail:

> The TTOR implies a partial order[3], which means that to a large extent transactions can be rearranged within a block without violating the consensus.

Yes and it is this ability that let Tom Zander remark (https://bitco.in/forum/threads/gold-collapsing-bitcoin-up.16/page-1217#post-78681) that this can be used to make Graphene efficient with the current implementation. A miner can reorder under the relatively weak constraint of the natural (partial) order, as it is required for validation. This would greatly reduce the extra cost to propagate ordering, such as done in the current BU Graphene implementation and could be done without a change of the validation rules. Which addresses the "Block propagation is more efficient" above and, furthermore, one might want to point out here that a Graphene block still following the natural order has a key advantage for the receiving end that is dropped completely in this analysis: He gets the block in an order he can validate it in, without further processing or reconstruction necessary. In other words, with lexicographic ordering, there is going to be adapters in the code that move from a natural order to the lexicographic one (which is always at minimum an O(n log n) operation), followed by an adapter that goes the other way around and reconstruct a partial from the lexicographic order.

> Considering the dominant use case of Bitcoin which is *electronic cash*, it can be reasonably expected that the average topological depth of transactions within a given block is relatively low, with probably a median depth of 1. From this insight, it can be also expected that *on average* a massive amount of transaction permutations within a block is also valid according to the topological transaction ordering rule.
>
> Let's immediately point out that over the first decade of operations of Bitcoin, there is no clear economic upside that has been identified in taking advantage of this flexible ordering of transactions. No significant participant is known to the authors to take advantage of this "feature" to achieve any purpose of economic value to herself or to the ecosystem at large.

This missing feature is hidden in plain sight. The natural order of transactions is simply an order in which to validate them. Yes, the order can be changed from the natural order to any other one that fulfills the "transactions come before their descendants" partial ordering requirement. This does not

take remove the fact that it already is in this way and is taken for granted. Taken for granted so much that the authors seem to be blind to its benefits, which are rather simply the fulfillment of *requirements*.

> Also, transactions can only be successfully propagated within the peer-to-peer network if broadcasted in their topological order. The inner ordering - or lack of ordering - of the transactions within a block has no positive influence on the transaction propagation, only a negative one as detailed in the following.

As an observation that might itself raise criticism due to its assumption of intent, I still like to say here that the part that strikes on “lack of ordering” seems a bit revealing to me, as it seems to emphasize the perceived (and emotionally understandable) intent of the authors to make something neater and more orderly. But the lack of ordering can be addressed in other ways, by for example sorting blocks first by partial and then lexicographic order, as Tom Zander suggested for Graphene.

Also, there is again an obvious positive that the authors miss: The successful propagation by natural order comes along with a successful validation in natural order by the network. When transactions go into the mempool, they can be validated for all the required, transaction-specific rule adherence at that point in time already.
Later on, the authors talk about having a smooth and continuous use of computing and network resources being desirable for transaction processing(see below).
But this is also the easiest to achieve when the transactions that come into the network naturally follow a partial order as necessary for validation! Resorting them when blocks come in is busy work that isn’t necessary with the current approach.

> *In conclusion, Bitcoin would not lose any of its economic properties if the TTOR was removed from the consensus rules.*

This seems to be a quick assertion without an underlying analysis. The above certainly isn’t that. In detail it can well be argued that the change of the ordering will change the incentives for transaction processing in minor ways at least, and I hope I have demonstrated above why that could also be in a negative way. Granted, it would most likely not cause catastrophic damage, but there has been no arguments presented to support making the above statement bluntly like that.

On this larger section:

Asymptotic performance analysis

I have to say that it is likely correct in its math but mostly irrelevant. Because it completely misses the point because of a key assumption being very wrong.
As discussed above, the authors have talked about how the transaction order will only allow transactions to come in in natural order. But that is exactly what is needed to quickly and efficiently validate.
Because the transactions come in the desired order **already**, there is NO NEED to rearrange in topological order and the cited complexities are absolutely meaningless. To exemplary go down on a few points to show how much of a red herring has been constructed here:

> In the specific context of Bitcoin, it can be reasonably assumed that most transactions will only consume endpoints that are already buried in a previous block - in this case the transaction can appear anywhere in a block. Even if only a small fraction of the transactions - say 1% - happens to be constructed over 0-conf transactions, this small fraction will be troublesome to manage. State of the art algorithms only offer $O(\sqrt{m})$ as the best performance for incremental topological ordering. For a block of 1TB, assuming 1% of non-trivial transactions, we have at least 40 millions edges to consider, i.e. a constant factor of $\sqrt{40,000,000} \approx 6325$ per insertion. This constant factor is not even taking into account the performance loss which will incur as the calculation is distributed over multiple processors. In practice, we are contemplating an overhead of order 10,000 per insertion as the direct consequence of the TTOR.

But the transaction ordering IS ALREADY topological. No sorting is necessary. The authors state earlier that the network will not accept out of order transactions, yet here they create an imaginary problem for 0-conf transactions, even though they follow this topological ordering as well.

> Like most theoretically hard algorithmic problems, the actual real-world topology of the 0-conf transaction graph of Bitcoin would probably lend itself to powerful heuristics. For example, it is likely that the 0-conf transaction graph could be near-perfectly partitioned into disconnected components which would lend themselves to low-overhead parallelism. However, while

---

Yes. Which is good. Which is the “make -j” approach. We want that.

> heuristics are tremendously efficient to cope with most situations, they tend to be weak against adversarial behaviors. Again, further heuristics can probably be uncovered to securely handle such adversarial behaviors, but at this point, Bitcoin would be piling sophisticated heuristics on top of other sophisticated heuristics which is an undesirable direction for infrastructure software.

And here’s a disconnect. Just because a heuristic can be used does not mean that a security analysis as well as constraining the security properties of the system is not possible. Discussions on Peter Tschipper’s parallel validation have actually shown how it addresses some of the feared security issues.
Now I agree in principle on the idea that reliance on heuristics are bad for infrastructure software, but we’re talking about the unavoidable nature of the blockchain here! And the reliance can be contained with a proper approach to validation.

> The computing resources to be paid for any given Bitcoin transaction are expected to be fundamentally *local* to the transaction. Indeed, transactions are paid for through *transaction fees* which mirror this locality. However, as pointed out in the previous section, the TTOR introduces a context-dependent cost where the marginal cost for 0-conf transactions built on top of other 0-conf transactions increases as their number grows over time until a block gets emitted.

As far as I can see, this has not at all been properly pointed out “in the previous section”. 0-conf transactions follow the very same partial order as confirmed transactions now, which is actually the intent of the authors to change!

> In particular, TTOR deters participants from building deep transaction graphs, which are of interest for many economic usages[5] where low latencies are expected and where the ambient trust is high, making 0-conf transactions secure thanks to a favorable context. This behavior goes somewhat against the fundamental economic model of Bitcoin. While TTOR remains a minor hindrance for Bitcoin, it cannot be considered a desirable property.

No, natural ordering DOES NOT deter participants from doing so. Natural ordering deters participants from doing so and then being **too lazy** to bring the set of nested transactions into the proper partial order for validation!

And, talking about the issue of "infrastructure software", it is usually a good guideline to make this infrastructure sofware as simple and dumb as possible. This means that the requirements to keep the partial order should be fulfilled at the edges (transaction generators) and not the core (nodes, validators) of the system!

> Despite the successive breakthroughs in algorithmics related to the incremental topological ordering, which happened essentially *after* the publication of the original Satoshi client, this problem remains very difficult. Considering that state-of-the-art results do not benefit from parallelized versions yet, it appears unclear to the authors whether those results will be further improved upon.

Again, the natural order **exists already**. No need to deal with fancy topological sorting algorithms, humanity (and maybe some bots) did that for you already when they sign the transactions. In order. As the form chains that are ordered causally. Causality implies topological transaction order.

On the proposed validation scheme:

> The `IsBalanced` function merely checks that the *bitcoins* in the inputs are greater or equal to the *bitcoins* in the outputs; however as the inputs are only known from the UTXO, this check has to be done in the second loop. This offline validation lends itself to a 2-stage embarrassingly parallel version, the first (resp. second) stage being associated with the second (resp. first) *for each* loop. The correctness of the approach is domain-dependant: the transaction graph is acyclic by construction due to the SHA256 hashing that takes place when an edge is introduced, i.e. when a transaction references another.

Yes, a two stage loop. But with the current natural ordering, it is also a two stage loop to do the validation, so the demonstration that validation is a two stage loop is quite meaningless. And for a single core, the approach works for all transactions, without ever having to break out of this loop due to an external dependency. Not so with parallel validation, and – as far as it seems – even more so when the natural order has been removed.

And it is of course, as the authors have complained about at length, only *mostly* embarrassingly parallel. Because those transactions that are pointing outside the boundaries of the chunks assigned to individual validation workers have to be validated according to topological order!

And what the… heck does a lexicographic ordering of transactions in the BLOCK help with validation here? *Where is the reliance on ordering by TXID id? There has been no motivation at all here on why this is advantageous.*

Yes, of course, there likely needs to be some kind of *UTXO data structure* that keeps transactions accessible by their ID. And of course it essential to have an index by transaction ID on this datastructure. But that is **independent** of the order of transactions in a block!

> Then, the online version of CTOR requires to maintain a *sorted set* which delivers $O(\log_2(n))$ for insertions, with well-known implementations that deliver near perfect distribution efficiency in practice as long as the number of processors - or rather shards - is not too high.

But this is independent of the order of transactions in the block!!!!!

> However, CTOR yields another property which is equally desirable: blocks can now be treated as *sets* of transactions, rather than being *lists* of transactions. Indeed, as the transaction identifiers can be recomputed from the transactions themselves, under the CTOR a block is fundamentally equivalent to a set of transactions, as there is only a single valid ordering for any set of transactions.

There's no motivation at all why this is desirable and I see an utter lack of data to support this view. And, yes, they can be seen of course as a set without an order if you remove the order. Very well. **But for validating them, one has to reinstate a partial order.** A partial order that is right now naturally kept within the blocks and removed – just to create the need to be reconstructed – with a resorting of the transaction in blocks.

> By simplifying the data model toward a *set*, CTOR brings to the blockchain entire classes of algorithms collectively known as the *set reconciliation problem*, see [8] for an introduction. Applying those results to Bitcoin yields efficient block propagation protocols, see [9]. Also, in terms of macro architecture, Bitcoin remains - as a whole - topologically sorted at the blockchain level.

But set reconciliation is already proven to work with the caveat of additionally transmitting ordering information, see the BU Graphene implementation. By the authors' own admission, the current ordering allows a huge deal of freedom, which translates into a large fraction of the ordering entropy that can be saved when miners reorder their transactions while keeping them in **a** validation order before transmitting them.

> Those results are important because they allow participants to make the most of their bandwidth capacity, propagating as much information as possible ahead of time prior to the emission of a new block.

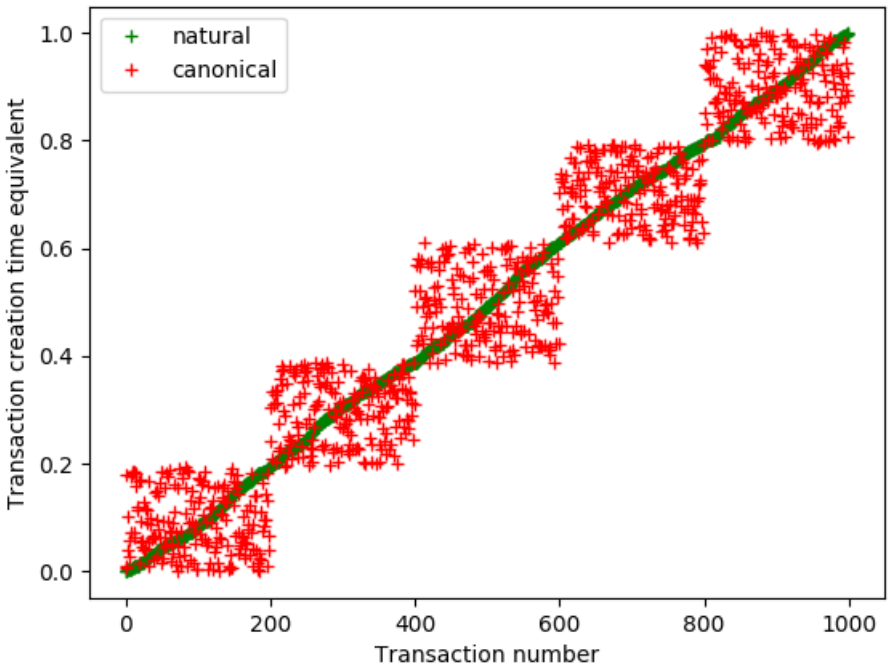
This propagation ahead of time happens already and it happens in natural order. Which is, curiously enough, most amenable to validation!

> The CTOR offers the possibility for any participant to zoom into a block to identify whether a transaction is found or not without processing the whole block. This property is of high interest because *chainless*[6] apps gain the possibility to verify flows of transactions without being encumbered by an arbitrarily large blockchain.

These "chainless apps" exist right now and they are also not encumbered by the arbitrarily large blockchain. The ability to serve and query blockchain data is independent of the transaction order in a block and is rather a question of database layout. One might reorder blocks to more easily serve data by transaction ID directly from this block data, but this comes with the trade-off that has been repeatedly point out above that it breaks the validation order!

If anything, an order will create further trouble by encumbering transactions within a block. Whereas the transaction order (if you could look at the inaccessible creation / signing time) right

now follows a pattern like the green crosses, with the new lexicographic ordering, it would follow rather a pattern like the red ones which break order at the block boundary in the following, admittedly crude illustration:



The block is a structured timestamp in the context of validation. Ordering by TXID and breaking block boundaries does not help at all with keeping with the flow of transactions in an even manner and NO argument has been given so far why that should be the case and the above illustration demonstrates that one creates an additional, artificial boundary in to the system that only seems to have the potential make things worse in this regard. See also above regarding the adapters needed to bring it back into partial order needed for validation. In any case, no proper arguments why this is beneficial have been brought forward.

On to “Opt-in locality”:

> The CTOR offers a new degree of control to the participants of Bitcoin: by using a process similar to vanity hashing, albeit focused on the transaction identifier rather than the address, a payer can *localize* her transaction within a block[7]. Assuming that some opt-in guidelines are established, close participants could decide to colocalize their respective transactions within specified identifier ranges within blocks.

Yes, but what for, exactly? I can localize transactions in a block, but I can use the same process right now to localize TXIDs in an UTXO database. What does it matter that they are localized in a block? Any method to query the block data still has to cut out the desired data. In any case.

Earlier on, the authors have identified the validation being the issue with scaling Bitcoin that they want to address with this proposal. **However, given this and the surrounding paragraphs the authors should ask themselves honestly whether the changes they propose are basically rather driven by somewhat easier *querying and presentation of the blockchain*, rather than validation of it.**

> As we have seen, coping with TTOR when considering blocks beyond 10GB will require complex data structures. It is reasonable to regard those future data structures as pending security liabilities, with security problems waiting to happen. In contrast, CTOR requires only semi-trivial algorithms, which are also straightforward to distribute.

That is again an assertion without proof. And I assert the opposite here, and have at least the sound but simple argument from above to back it up: The needed partial order is the validation order. Such a partial order is always needed. It currently comes with Bitcoin for free. This proposal wants to remove and replace this order, but there has been no argument forward why the lexicographic order can in any way solve the problem of having a transaction dependency graph!

Finally let me also use this place to address a point that isn't directly made in this paper but has been brought up as an advantage of by-TXID ordering but also weakens drastically upon closer inspection of the necessary preconditions and scenarios it could be applied in.

That is the idea of being able to "Prove the absence of a transaction in a block".

With a lexicographic ordering, the question of "Is this transaction with ID X in Block at height Y (or with hash Z)?" can be answered in the negative in an efficient way that it cannot be answered without this canonical ordering. The reason is that a two neighboring leafs in the merkle treedown to the transaction in question can act as "pliers" of sorts and prove exclusion of a given TXID.

But this is simply the result of asking the wrong question. The question that could be asked instead is "Is the transaction with ID X in Block at height Y (or with hash Z) and merkle-tree encoded position P?".

And this can be answered in all cases in the negative and the positive with the current implementation.

Conclusions

I think the above demonstrates why we should all really ask ourselves whether we want to go forward with this in November, or postpone this to have a second look. Especially the *in-the-field observation* by Tom Zander that at least one current implementation suffers from a reduction in validation speed concerns me greatly and tells me that it is rather a good idea to wait a bit more with a further analysis and proper addressing of the above criticisms. Which are not only made by me or Tom Zander btw., I am sure I have seen some of them being made elsewhere, in comments deep on Reddit and similar. So any reader who wants to be properly cited as having prior art here, please drop me a link.