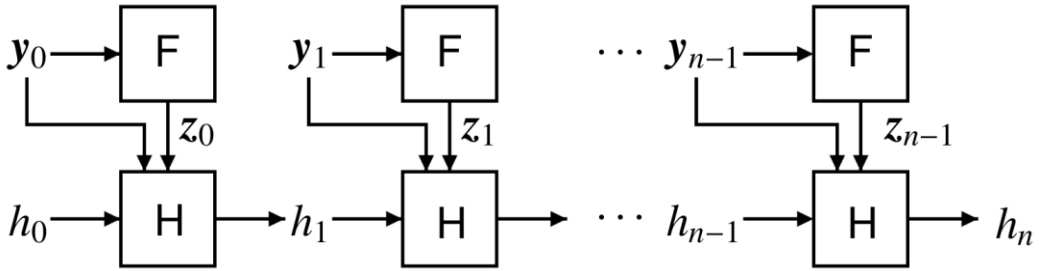
Scalable Batch Verification of ECDSA for Blockchain using IVC

Li LIU, Puwen WEI, Shuchang LIU, Zirui WANG, Da HU, Zengjie KOU

Frontiers of Computer Science, DOI: 10.1007/s11704-025-41269-5

Problems & Ideas

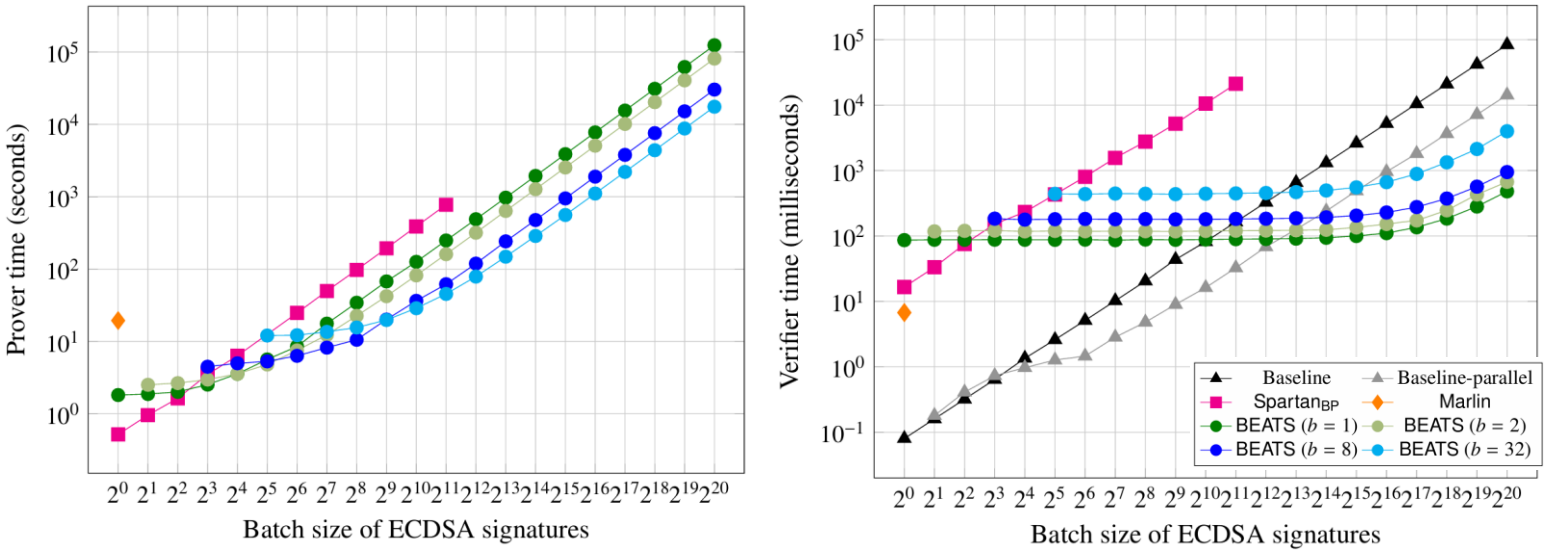
- Problems of signatures verification on blockchain:
  - The escalating volume of transactions on blockchains has made signature verification a critical performance bottleneck.
  - Existing approaches of packing signatures inside SNARK often incur significant computational overhead, demanding substantial time and memory resources for proof generation.
- Ideas: Apply a modified, memory-friendly incremental verifiable computation protocol optimized for specific signatures in blockchain to enhance scalability and reduce memory consumption



The main idea of our construction. The function F represents a batch of signature verification. The input to each F is a vector $\boldsymbol{y}_i = \{(pk_{i,j}, \sigma_{i,j}, m_{i,j})\}$, and the output of F is $\boldsymbol{z}_i$, which indicates the verification result of the signatures in $\boldsymbol{y}_i$. The function H represents the collision-resistant hash function, which is used as a commitment to these inputs and outputs.

Main Contributions

- Contributions:
  - A general approach to batch verification of arbitrary signatures on blockchain that enhance scalability, reduce memory consumption, and ensure compatibility with common devices while supporting an arbitrary number of signature verifications;
  - A concrete instantiation, BEATS, for batch verification of the ECDSA signature. The experiment, evaluated on a virtual machine with 8 cores and 16 GB RAM, shows significant performance gains compared to $\text{Spartan}_{\text{BP}}$ and Marlin.



Comparison of the prover and verifier time between BEATS, $\text{Spartan}_{\text{BP}}$ and Marlin at different batch sizes of ECDSA signatures. Left: Comparison of the prover time; Right: Comparison of the verifier time.