



# Data Transfer Batching Techniques

feat. Blake3, CAR Mirror,  
GraphSync, and more

Philipp Krüger

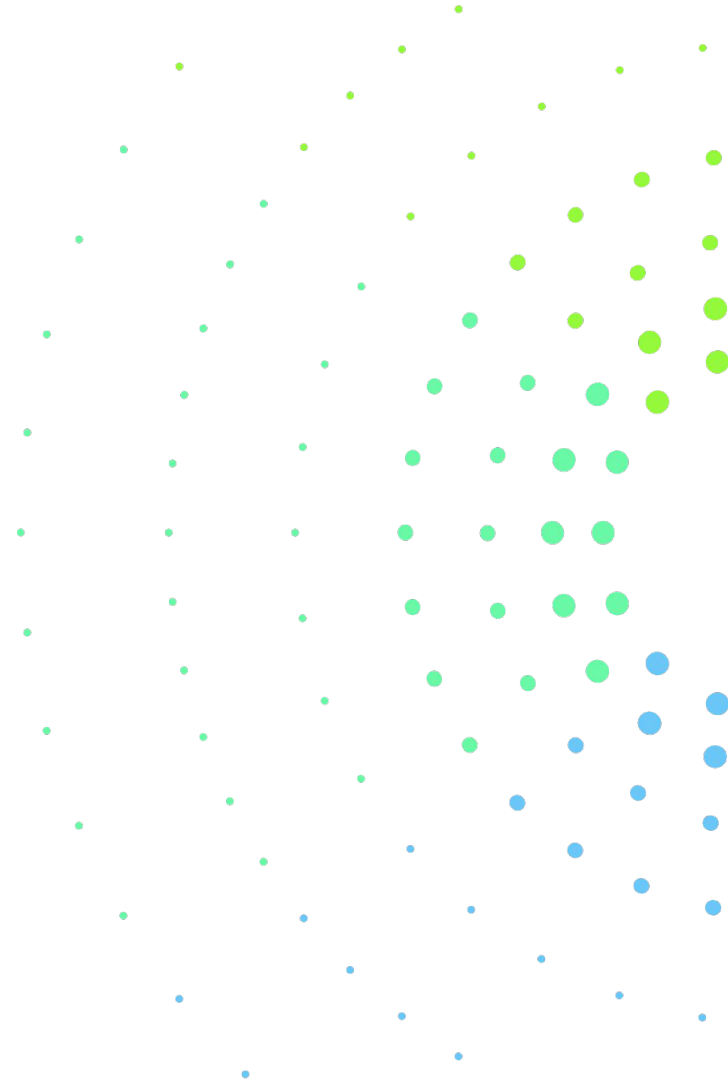


## Data Transfer in IPFS?

`ipfs pin bafy... !`



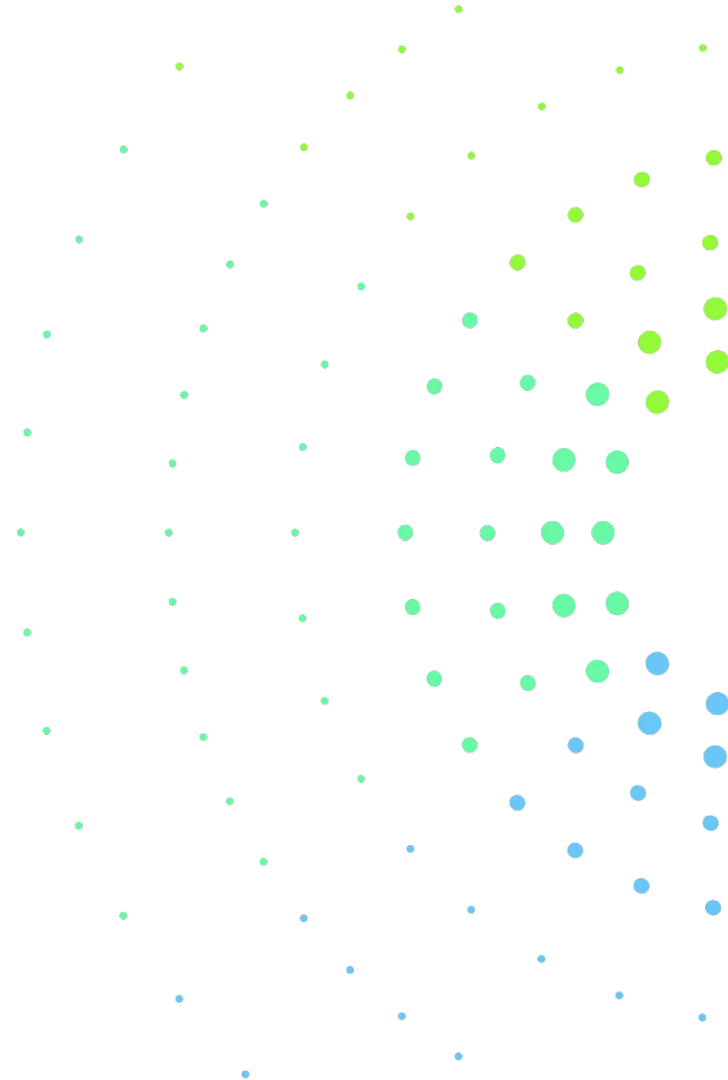
- DHT issues:
  - Connection Management
  - Lots of WANTS
- Transfer issues:
  - Fetch root block
  - Find more CIDs
  - Fetch block
  - Find more CIDs
  - ...
  - Round-trips!



## Solution: Batching

But how?

- Bigger Blocks?
- Send CAR files?

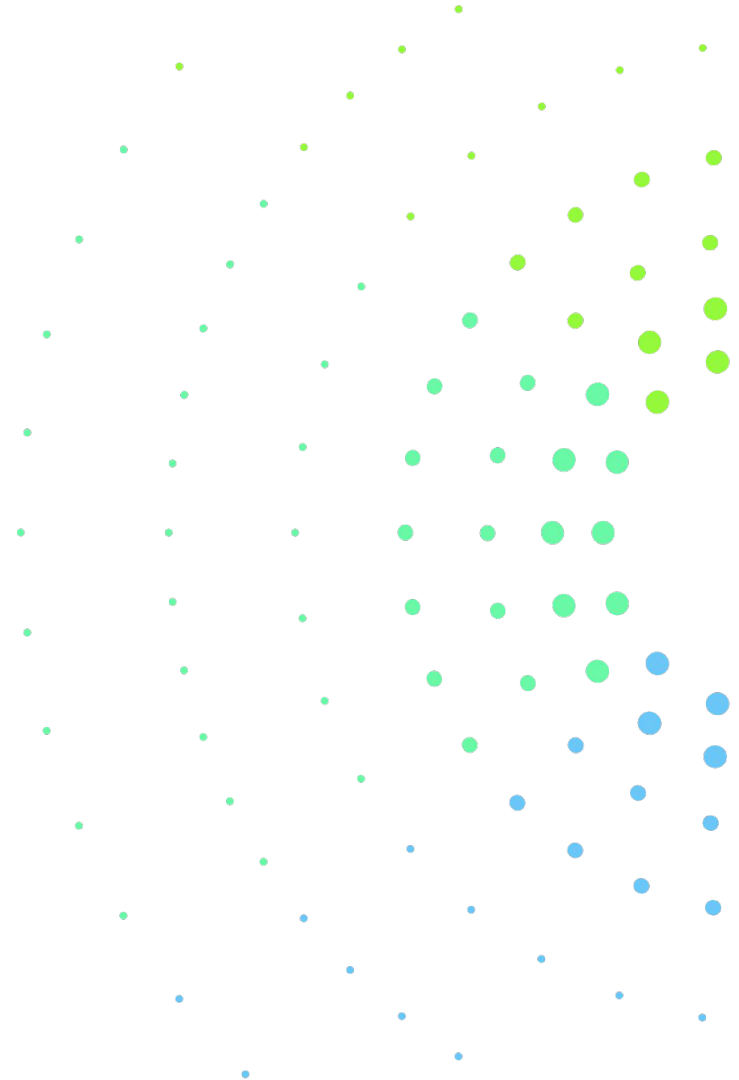




**IPFS PING**  
Brussels, Belgium  
2023

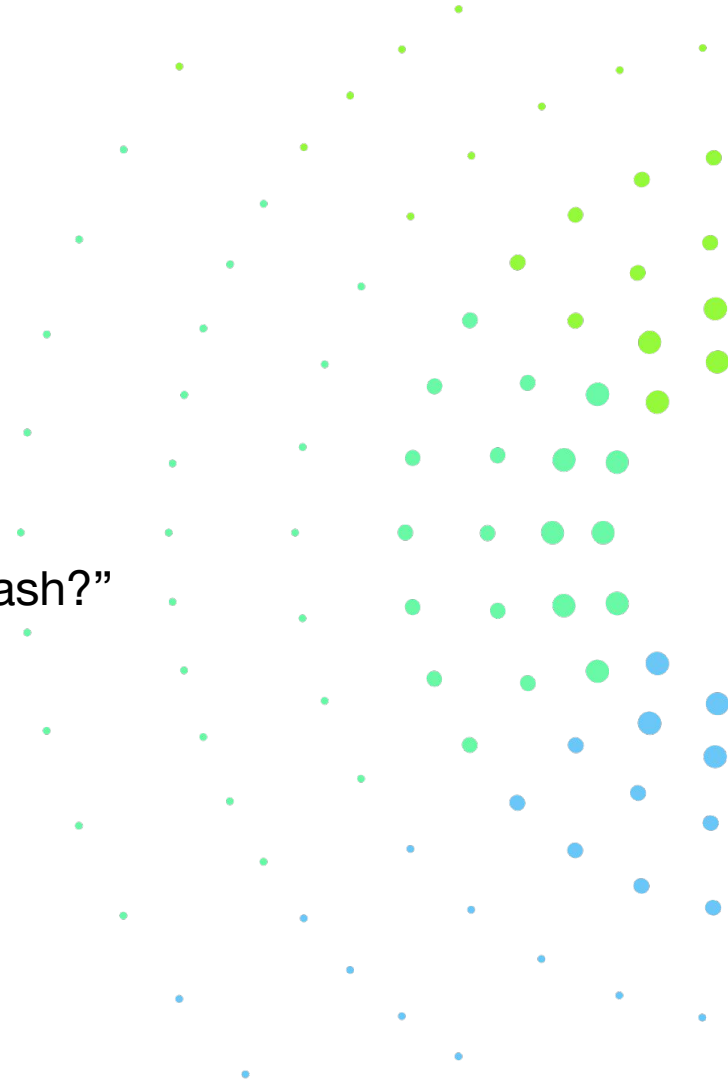
## Data Transfer: Design Goals

- Few round-trips



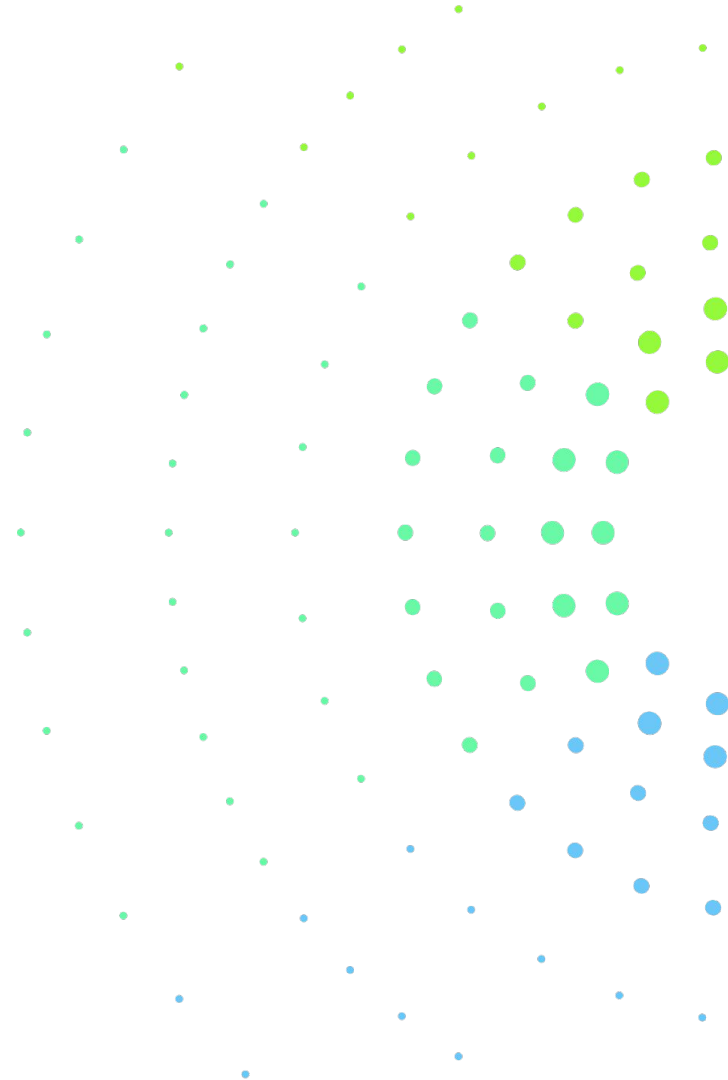
## Data Transfer: Design Goals

- Few round-trips
- Incremental Verifiability
  - “Is the data that I got sent part of this hash?”
  - “Untrusted data buffer size”
  - anti-DoS / part of minimizing trust



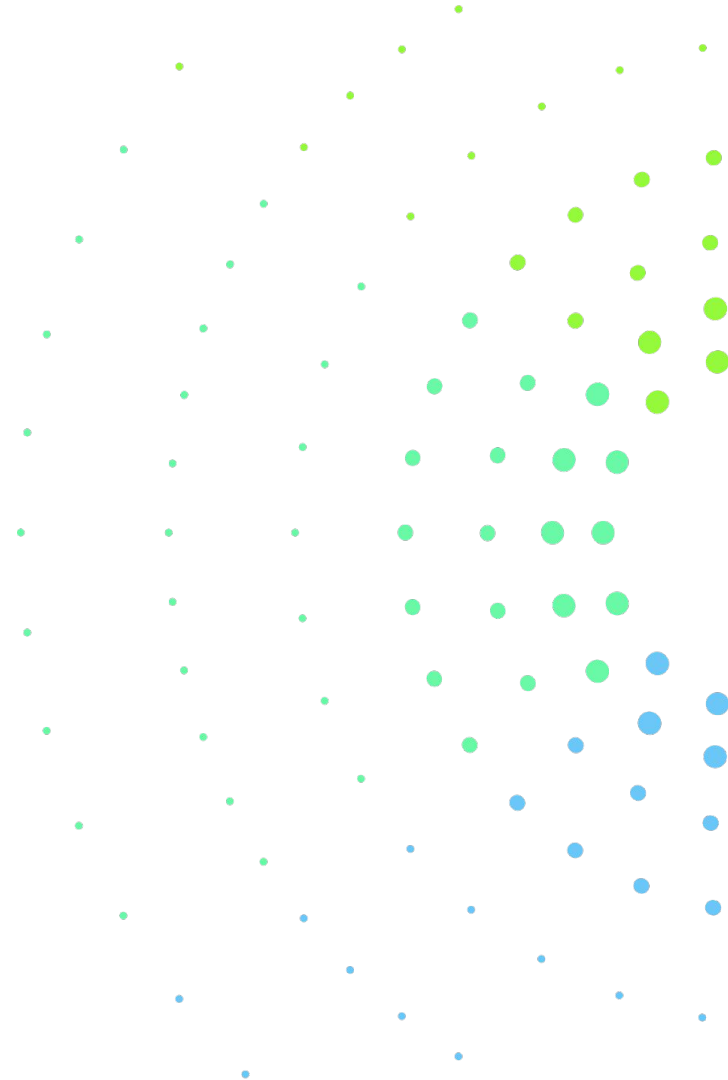
## Data Transfer: Design Goals

- Few round-trips
- Incremental Verifiability
- Query support
  - E.g. root CID + IPLD selector
  - Or: range queries on files



## Data Transfer: Design Goals

- Few round-trips
- Incremental Verifiability
- Query support
- Deduplication
  - When you actually want to *sync* DAGs
  - Or resume partial transfers
  - Or take advantage of structural sharing



## Data Transfer: Design Goals

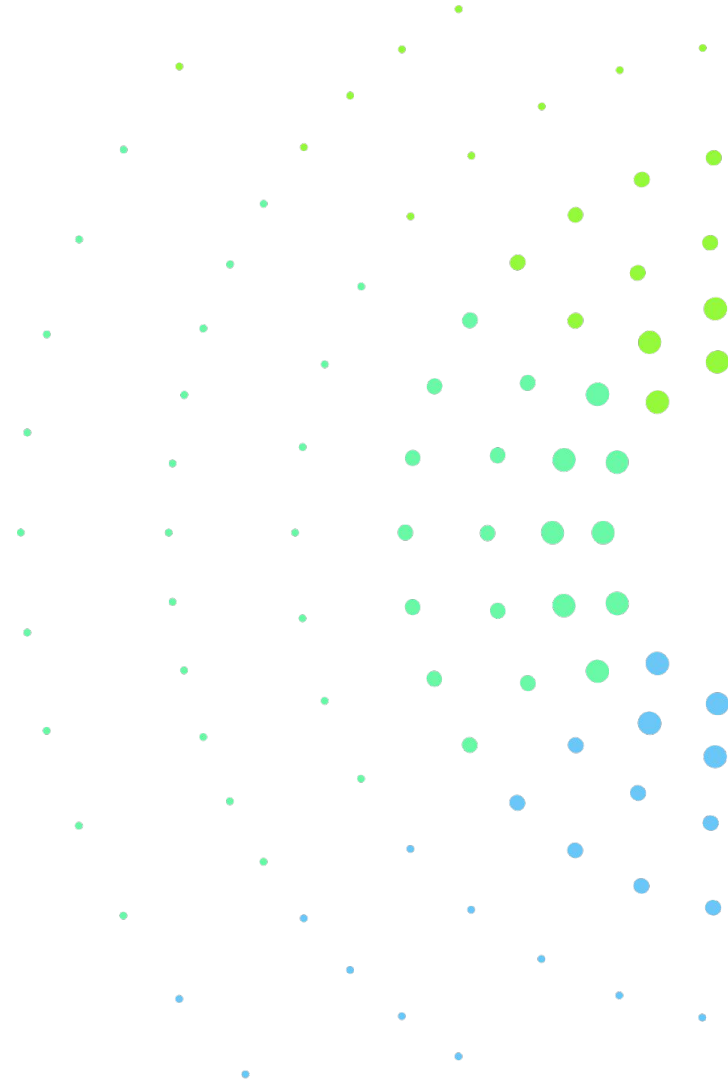
- Few round-trips
- Incremental Verifiability
- Query support
- Deduplication
- DAG layout flexibility
  - Allow small blocks, useful for frequent-write systems





## Data Transfer: Design Goals

- Few round-trips
- Incremental Verifiability
- Query support
- Deduplication
- DAG layout flexibility



ipfs pin  
/bitwap

CAR file transfer

CAR mirror

GraphSync

Blake3 + Bao

Few round-trips

Incremental Verifiability

Query support

Deduplication

DAG layout flexibility

ipfs pin  
/bitswap

CAR file transfer

CAR mirror

GraphSync

Blake3 + Bao

Few round-trips



Incremental Verifiability



Query support



Deduplication





DAG layout flexibility



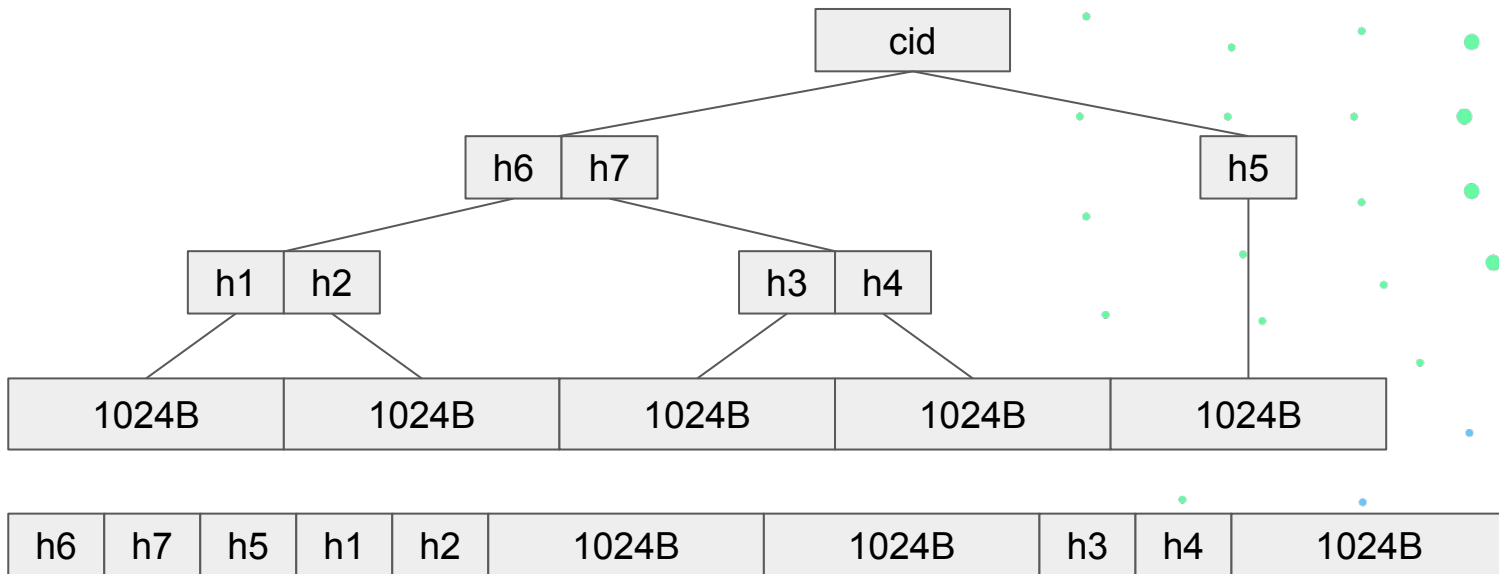
	ipfs pin /bitswap	CAR file transfer	CAR mirror	GraphSync	Blake3 + Bao
Few round-trips	✗	✓			
Incremental Verifiability	✓	■			
Query support	✓	✗			
Deduplication	✓	✗			
DAG layout flexibility	✓	✓			

	ipfs pin /bitswap	CAR file transfer	CAR mirror	GraphSync	Blake3 + Bao
Few round-trips	✗	✓	✓		
Incremental Verifiability	✓	■	✓		
Query support	✓	✗	✗		
Deduplication	✓	✗	✓		
DAG layout flexibility	✓	✓	✓		

	ipfs pin /bitswap	CAR file transfer	CAR mirror	GraphSync	Blake3 + Bao
Few round-trips	✗	✓	✓	✓	
Incremental Verifiability	✓	■	✓	✓	
Query support	✓	✗	✗	✓	
Deduplication	✓	✗	✓	■	
DAG layout flexibility	✓	✓	✓	✓	

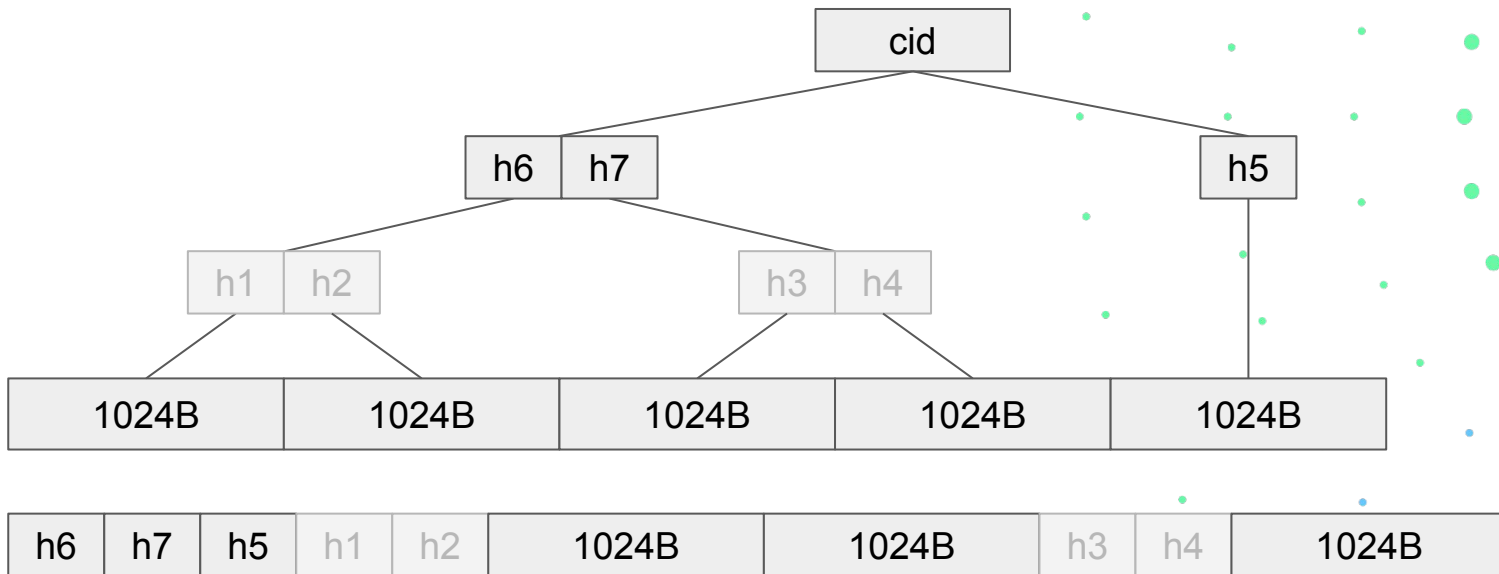
	ipfs pin /bitswap	CAR file transfer	CAR mirror	GraphSync	Blake3 + Bao
Few round-trips	✗	✓	✓	✓	✓ 
Incremental Verifiability	✓	■	✓	✓	✓ 
Query support	✓	✗	✗	✓	■
Deduplication	✓	✗	✓	■	■
DAG layout flexibility	✓	✓	✓	✓	✗

# Idea: Configurable Incremental Verifiability in Blake3+Bao





# Idea: Configurable Incremental Verifiability in Blake3+Bao



# Idea: Configurable Incremental Verifiability: General IPLD

```
[  
  [  
    64,  
    0  
  ],  
  [  
    { "/": { "bytes": "kBcpNaugz ... " } },  
    { "/": "bafkreieyjowsrfsrjb7n3qcw37cbiqfql6t7ic6bw4x5uslndmzcumuhre" }  
  ]  
]
```

```
{ "/": { "bytes": "HWIyQ ... " } }
```



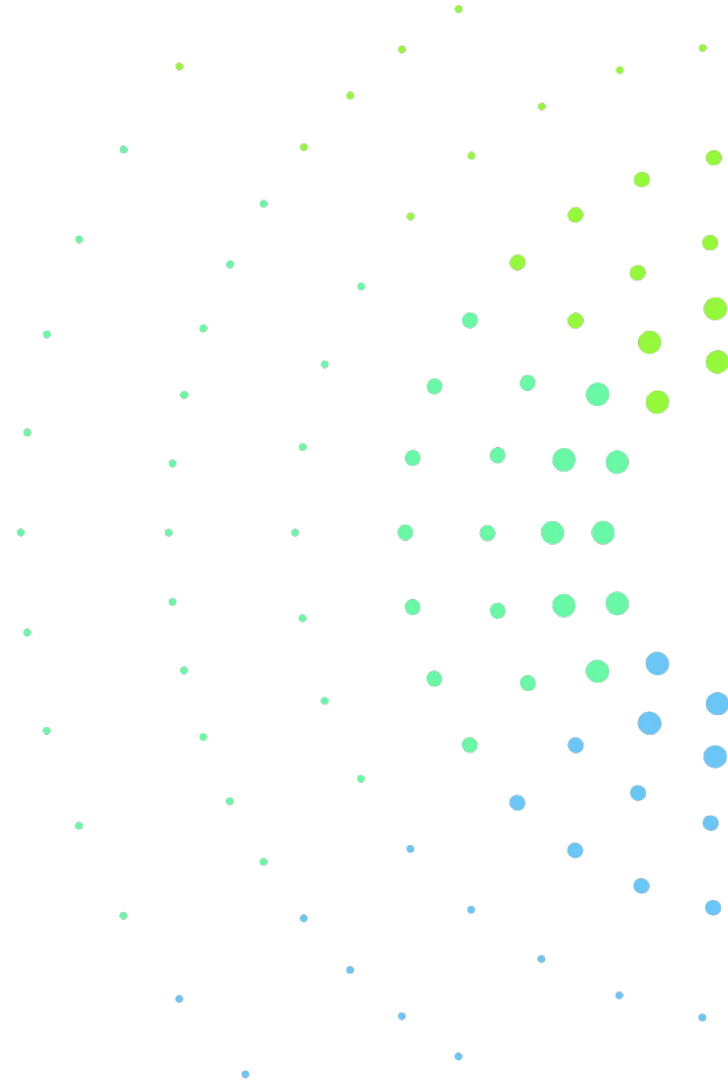
# Idea: Configurable Incremental Verifiability: General IPLD

```
[  
  [  
    64,  
    0  
  ],  
  [  
    {  
      "/": {  
        "bytes": "kBcpNaugz ... "  
      }  
    },  
    {  
      "/": {  
        "inline": {  
          "mh": "sha2",  
          "codec": "dag-cbor",  
          "/": {  
            "bytes": "HWIyQ ... "  
          }  
        }  
      }  
    }  
  ]  
]
```



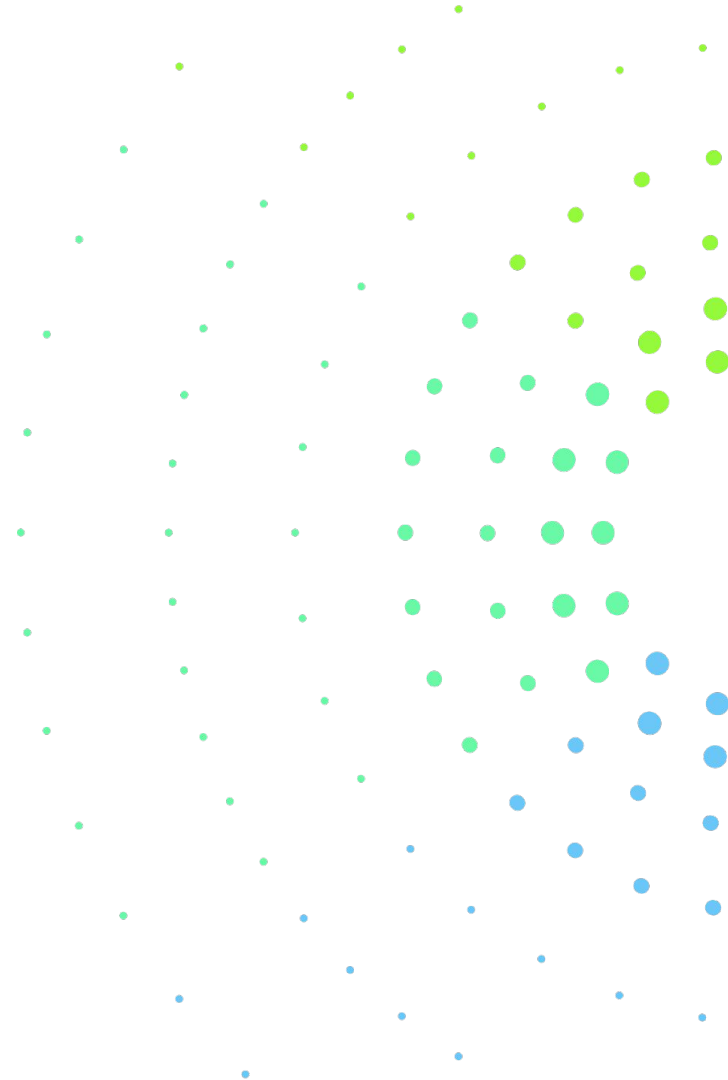
## Idea: Configurable Incremental Verifiability: General IPLD

- Not the same as identity hashes!
  - Replaced with corresponding CID when parent block's hash is calculated
- Can approach lower limit of “unverified data buffer” size!
  - Inline blocks as long as the parent block is  $<$  unverified data buffer size
- ➔ Support lots of small blocks
- Is this a CAR v3?



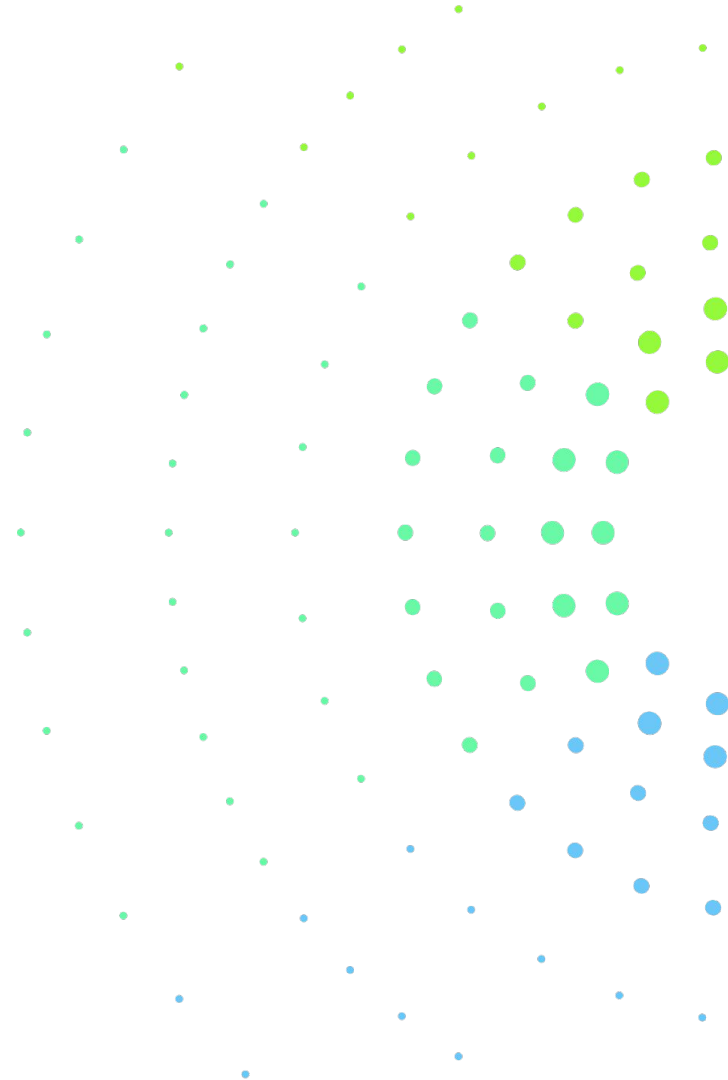
## Idea: Queries are code

- codecs/ADLs are pre-loaded code for computing a set of blocks from a path
  - Needs agreement on set of ADLs
  - This slows down “new ADLs”, e.g. proly trees
- Use Wasm to run “arbitrary selectors”?
  - Wasm has mostly been “agreed upon”
  - However:
  - Need to transfer .wasm
  - Concerns regarding DoS
- Easier once we have IPVM?



## Idea: Deduplication ... is a Query

- Select all blocks except the ones I already have
- E.g. send a Wasm with a hard-coded bloomfilter/roots
- Split data transfer conceptually into
  - Preprocessing and
  - Transfer





IPFS PING

Brussels, Belgium

2023

## That's it!

- I'm excited for what's happening
- Thank you!

