IPVM

The Long-Fabled Execution Layer

IPVM

The Long-Fabled Execution Layer

"The Easiest Way to Run Wasm Everywhere"

IPVM

The Long-Fabled Execution Layer

"The Easiest Way to Run Wasm Everywhere"

The Fastest Way to Ship IPFS Features to Users

IPVM

The Long-Fabled Execution Layer

"The Easiest Way to Run Wasm Everywhere"

The Fastest Way to Ship IPFS Features to Users

A Step Towards IPOS

Agenda

1. What
2. Why
3. Security
4. Scheduling
5. Compute Commons
6. Open Discussion 🌶️

Super early days 🌋🦖

IPVM

IP What?

IPVM

IP What?

- A blessed VM (Wasm) in every IPFS node
- Transparent IPFS node upgrades (like the web) 🎭
- Support features like Autocodec
- Compute without (required) consensus
- Global adaptive optimization
- Mobile (ambient) computing

IPVM

What This Gets You

IPVM

What This Gets You

- Like data, compute should be ubiquitous
- End users and IPFS teams can depend on having compute around
- Deep integration with tooling
- Fully consistent functionality between clients
- Replace AWS's proprietary Lambda with an open protocol + nodes
  - **"The HTTP of Compute"** 🤩

Requirements

Working Backwards

Requirements

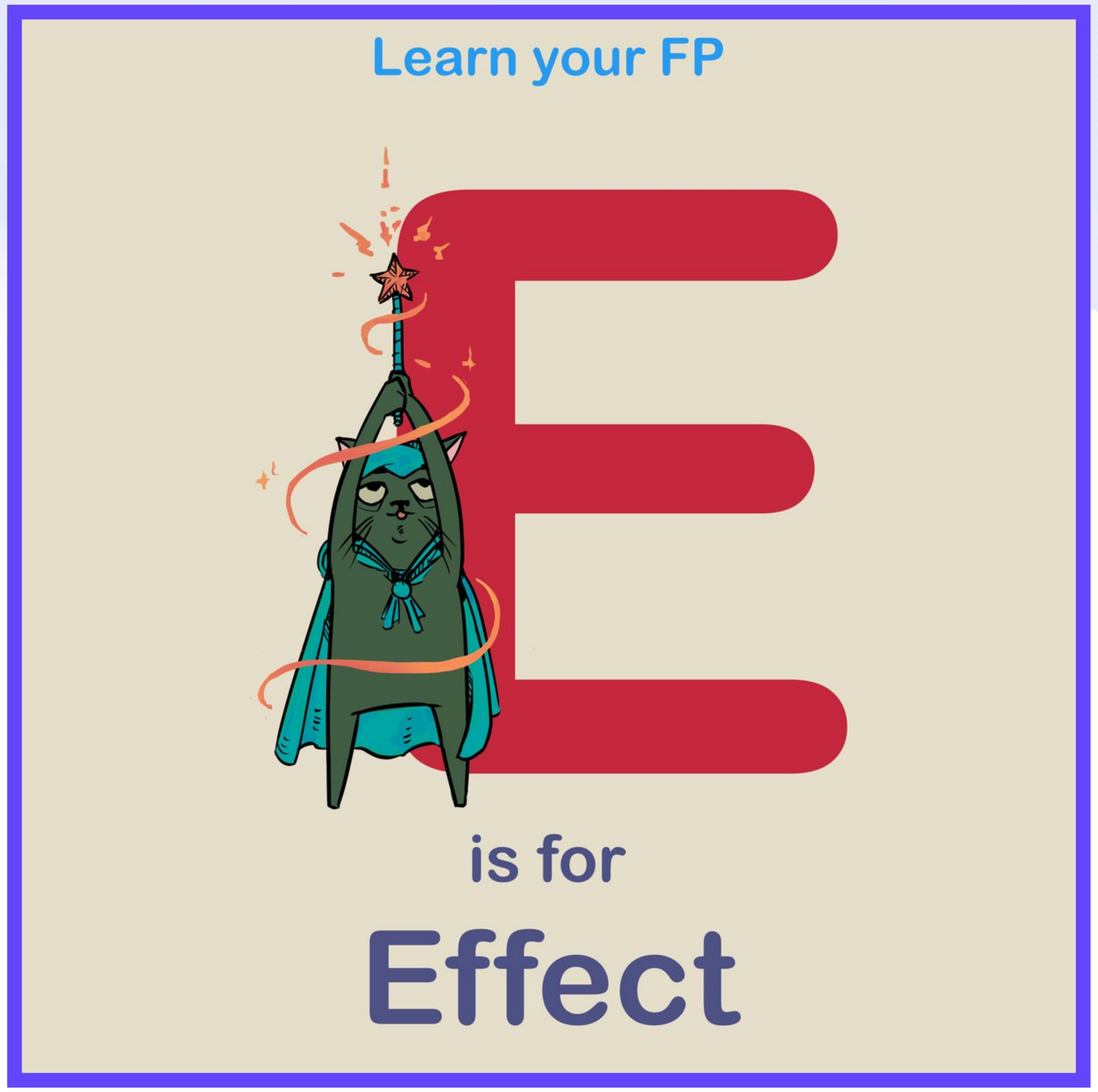
The Basics

Requirements
The Basics

- Portable
- Deterministic, verifiable
- Purity? Managed effects?
- Enforced termination (enabled-by-default?)
- Move compute to data / move data to compute
  - Push & pull both important
  - (e.g.) UCAN for remote invocation

Requirements
The Basics

- Portable
- Deterministic, verifiable
- Purity? Managed effects?
- Enforced termination (enabled-by-default?)
- Move compute to data / move data to compute
  - Push & pull both important
  - (e.g.) UCAN for remote invocation



https://twitter.com/impurepics

Requirements

Familiarity as Adoption Tactic

Requirements

Familiarity as Adoption Tactic

- Bring your own language
- Common patterns (e.g. manifests, cron, systemd, build packs)

Requirements

Deep Integration

Requirements

Deep Integration

- Lean into content addressing
- More than the sum of its parts
- Local and remote execution
- Reuse Wasm effort, infra, tooling, community experience, etc
    - FVM, Aquamarine, CloudFlare Workers, Bacalhau, web3.storage invocations, IPFS-FAN

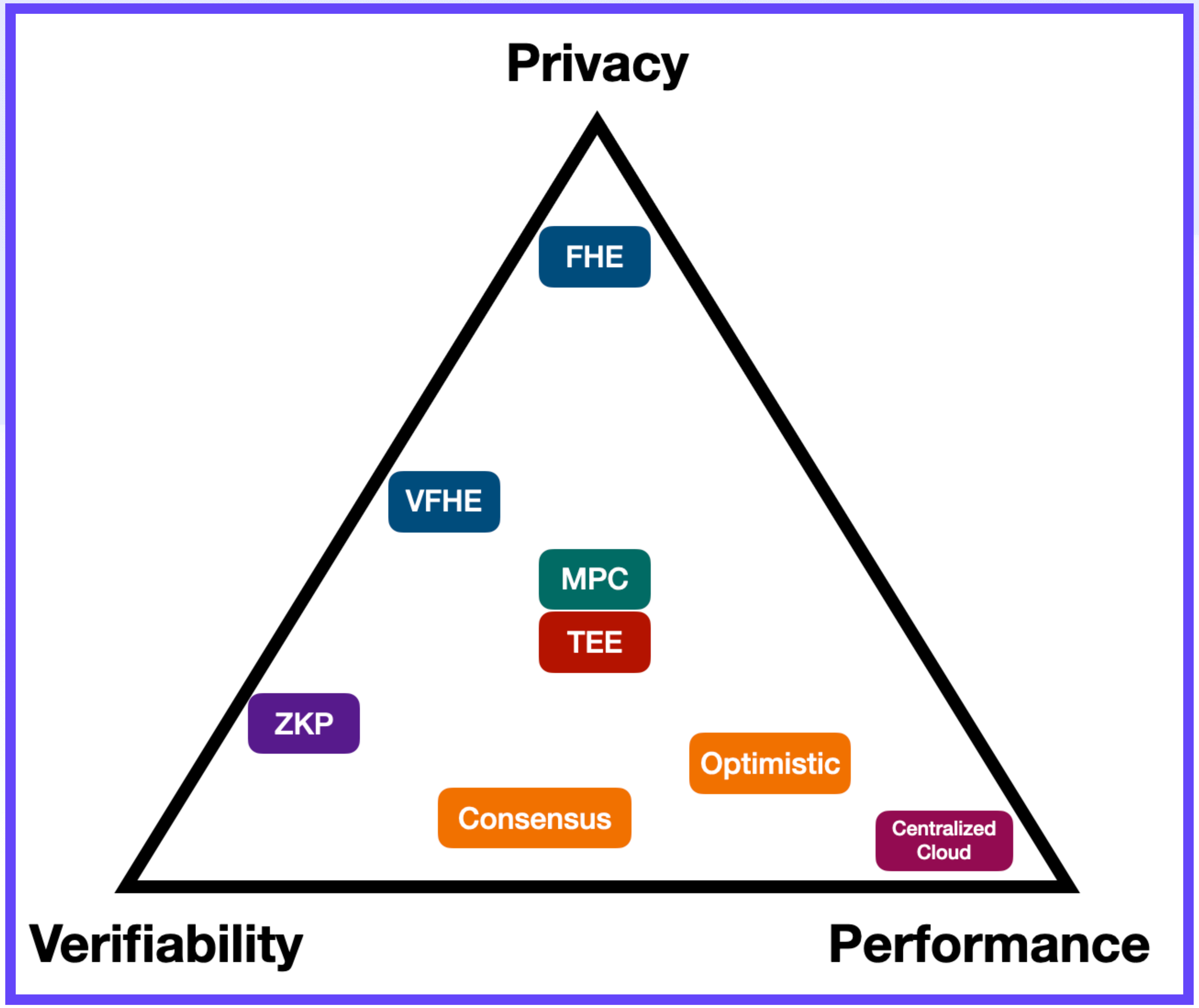
Requirements

Deep Integration

- Lean into content addressing
- More than the sum of its parts
- Local and remote execution
- Reuse Wasm effort, infra, tooling, community experience, etc
    - FVM, Aquamarine, CloudFlare Workers, Bacalhau, web3.storage invocations, IPFS-FAN

Requirements

Juan's Triangle



Requirements

Antigoals?

Requirements
Antigoals?

- Suggestions? 🙏

Execution-as-IPLD

Interplanetary Linked Invocation (IPLI)

Execution-as-IPLD

Invocation via IPLD

Execution-as-IPLD

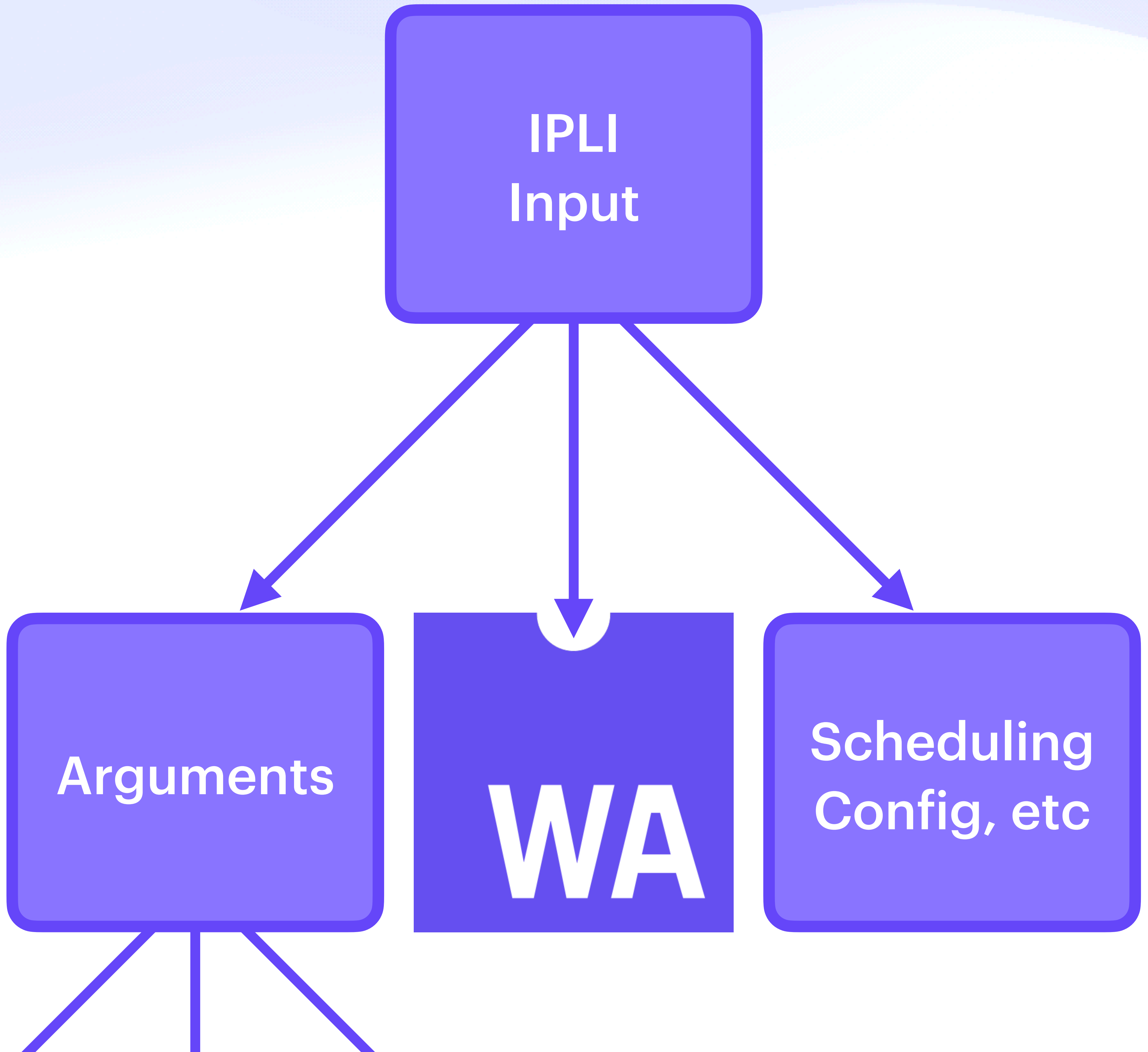
Invocation via IPLD

- Description of jobs & results
- Index and/or names for later lookup
- Streams of results per machine

Execution-as-IPLD

Invocation via IPLD

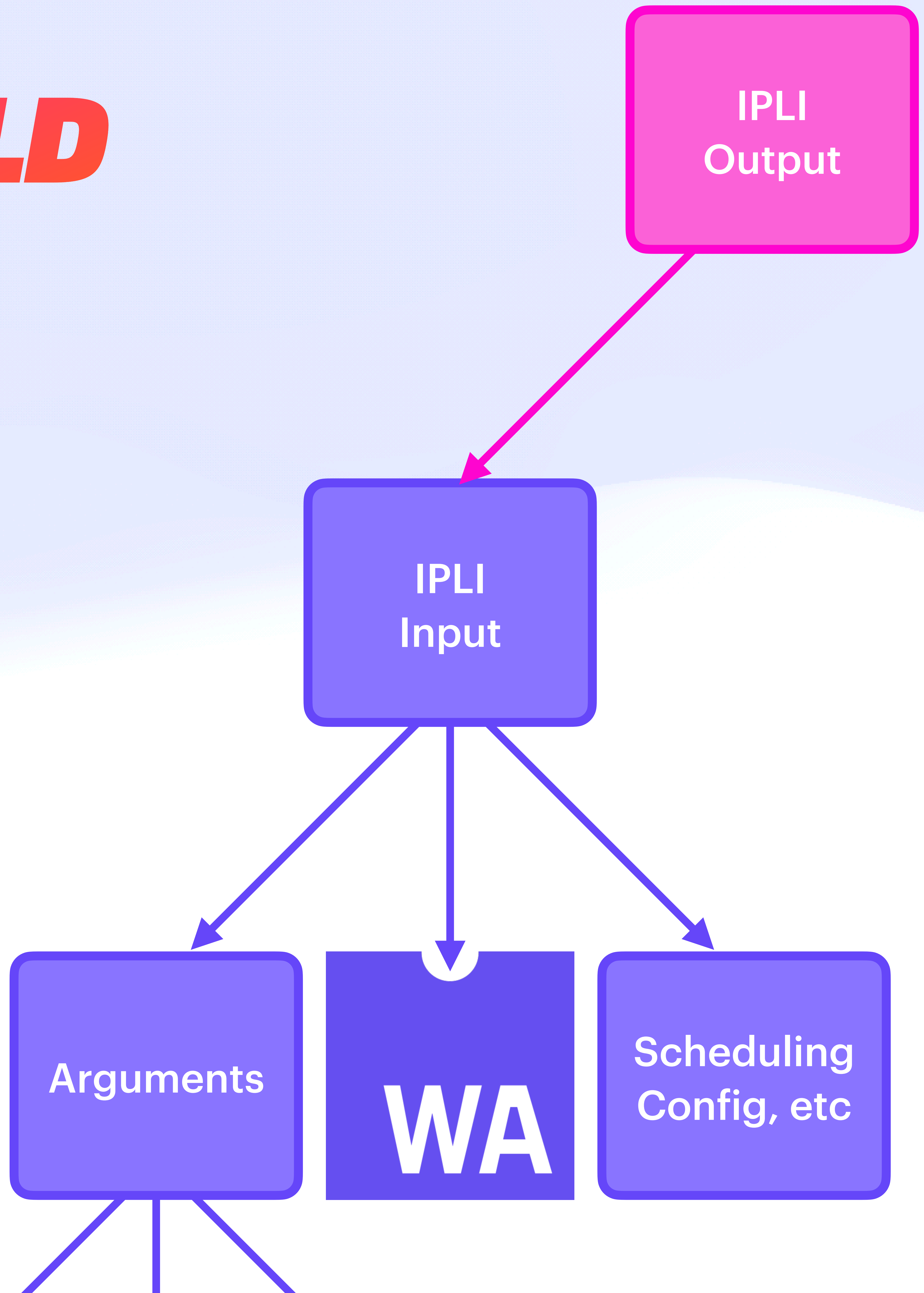
- Description of jobs & results
- Index and/or names for later lookup
- Streams of results per machine



Execution-as-IPLD

Invocation via IPLD

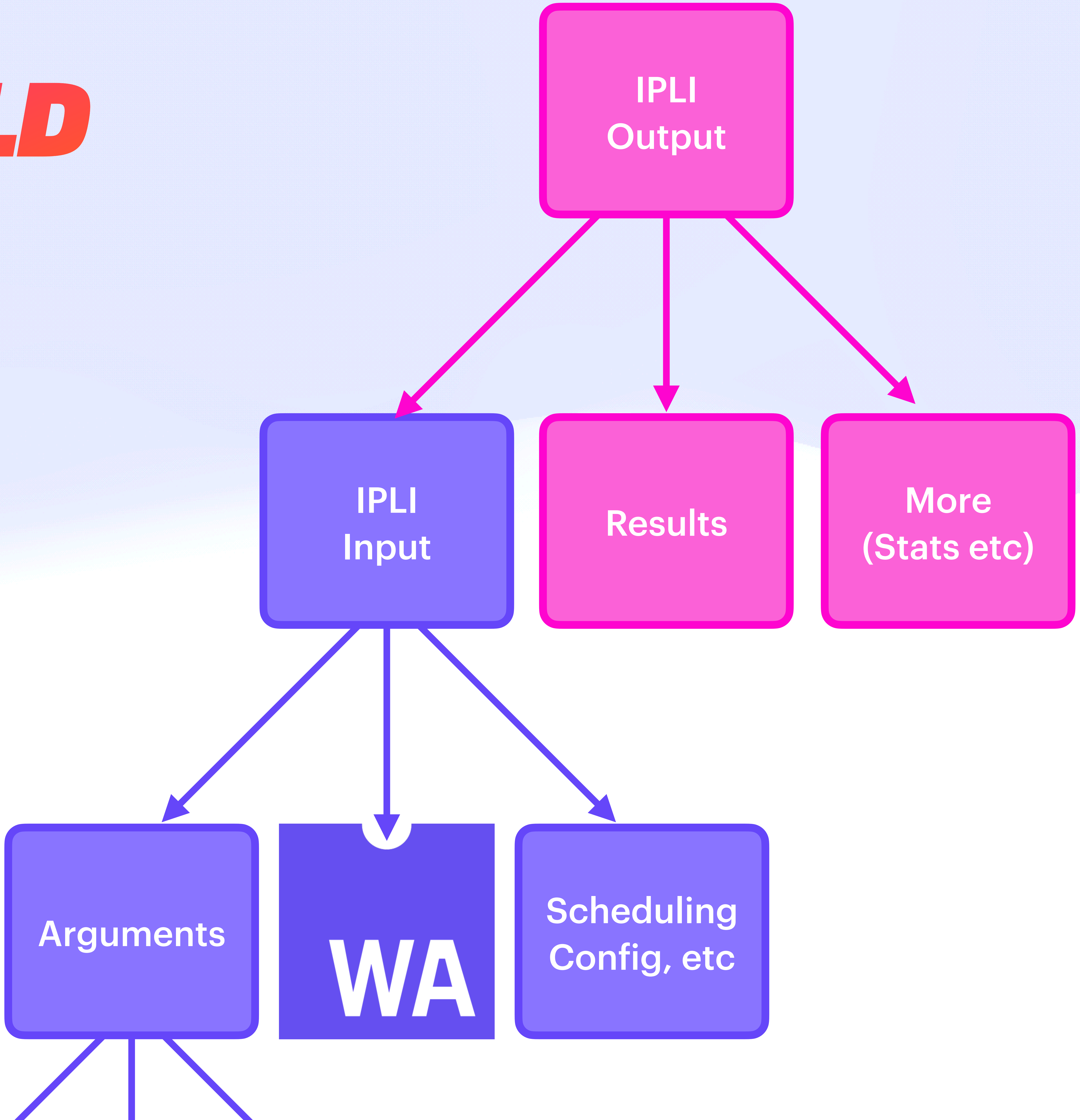
- Description of jobs & results
- Index and/or names for later lookup
- Streams of results per machine



Execution-as-IPLD

Invocation via IPLD

- Description of jobs & results
- Index and/or names for later lookup
- Streams of results per machine

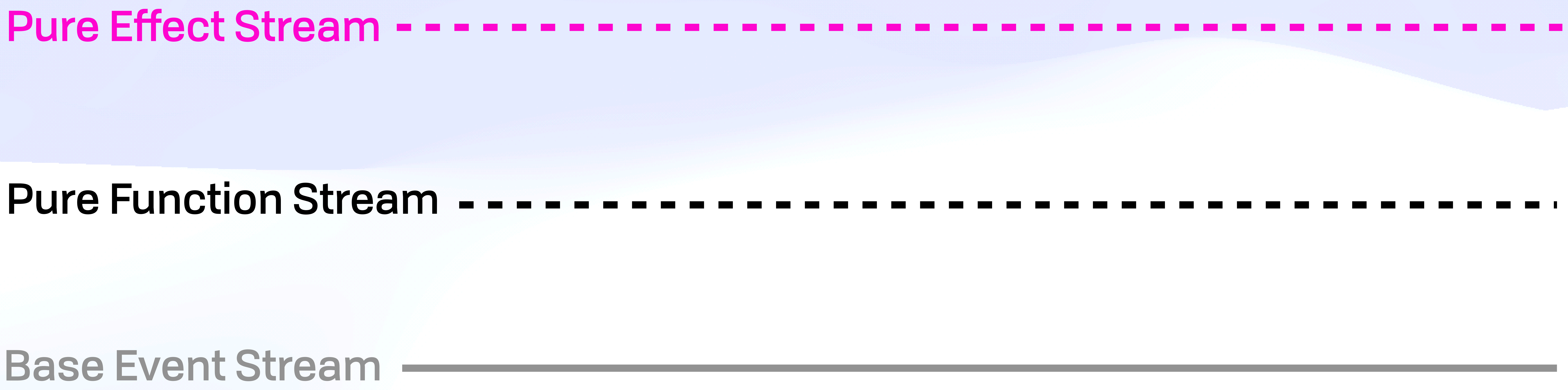


Execution-as-IPLD

Job Streams (Scheduler + Events)

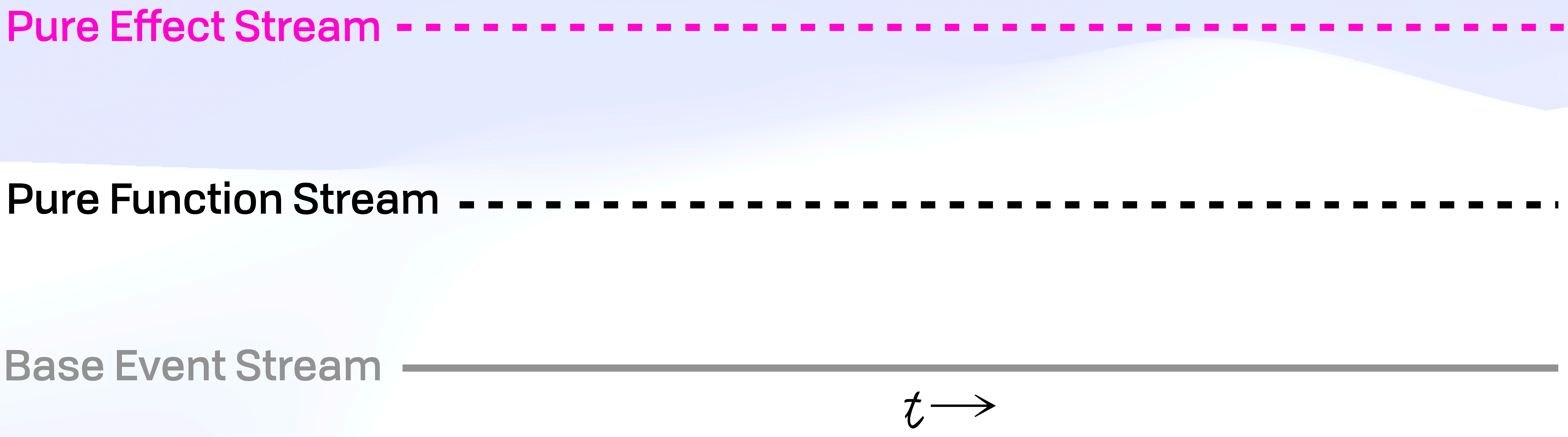
Execution-as-IPLD

Job Streams (Scheduler + Events)



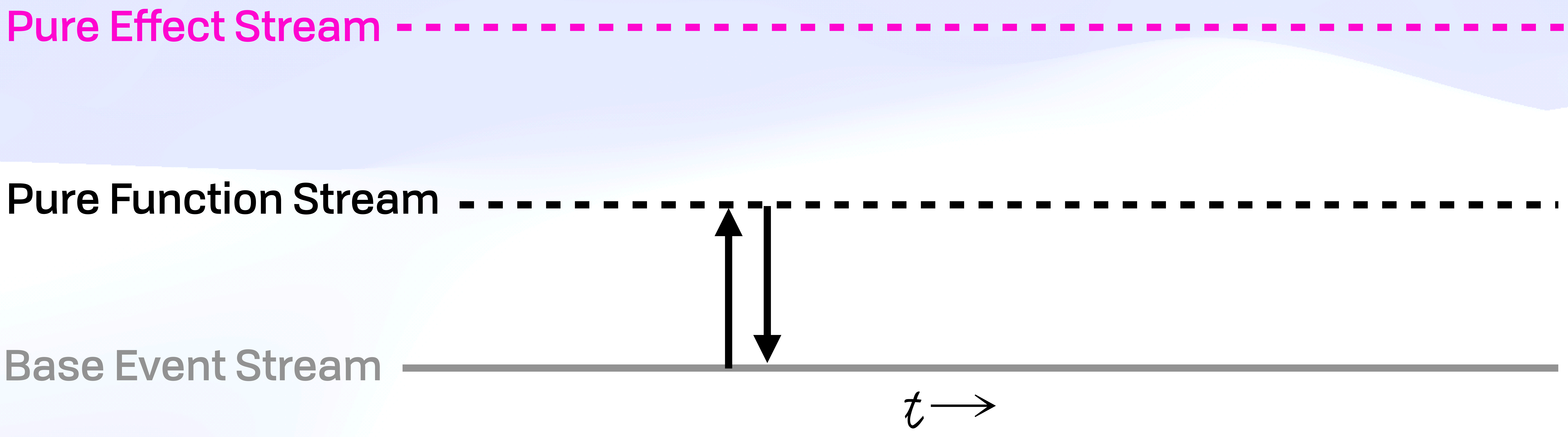
Execution-as-IPLD

Job Streams (Scheduler + Events)



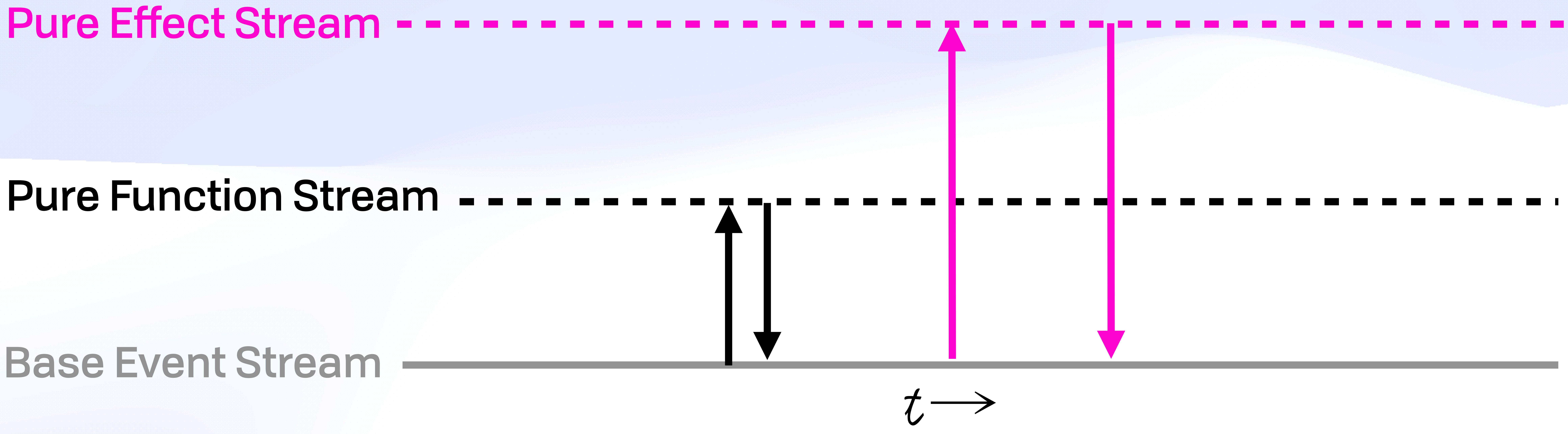
Execution-as-IPLD

Job Streams (Scheduler + Events)



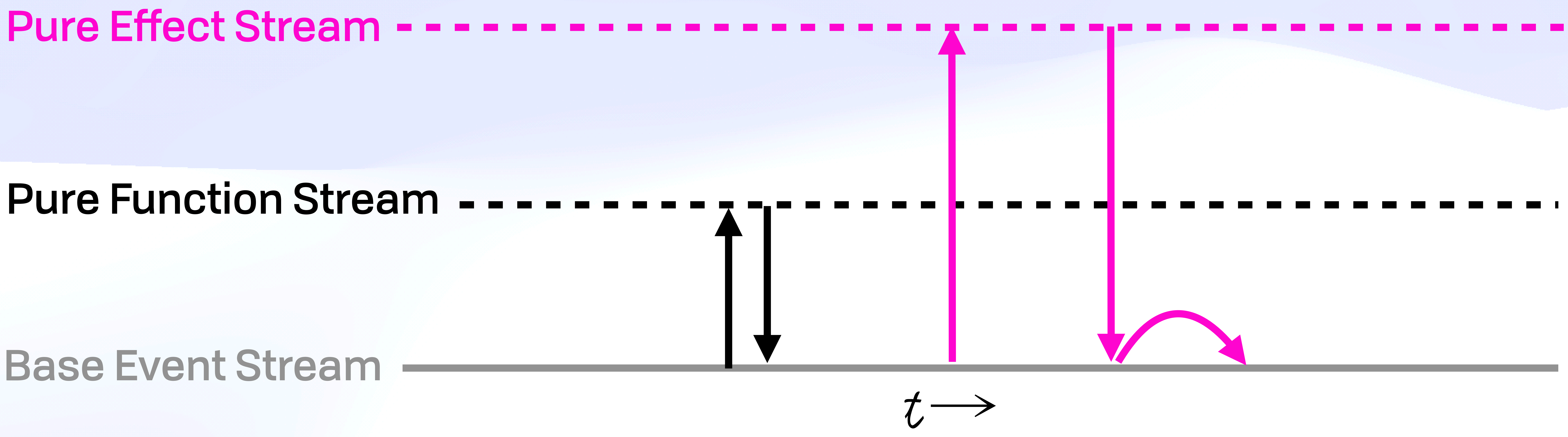
Execution-as-IPLD

Job Streams (Scheduler + Events)



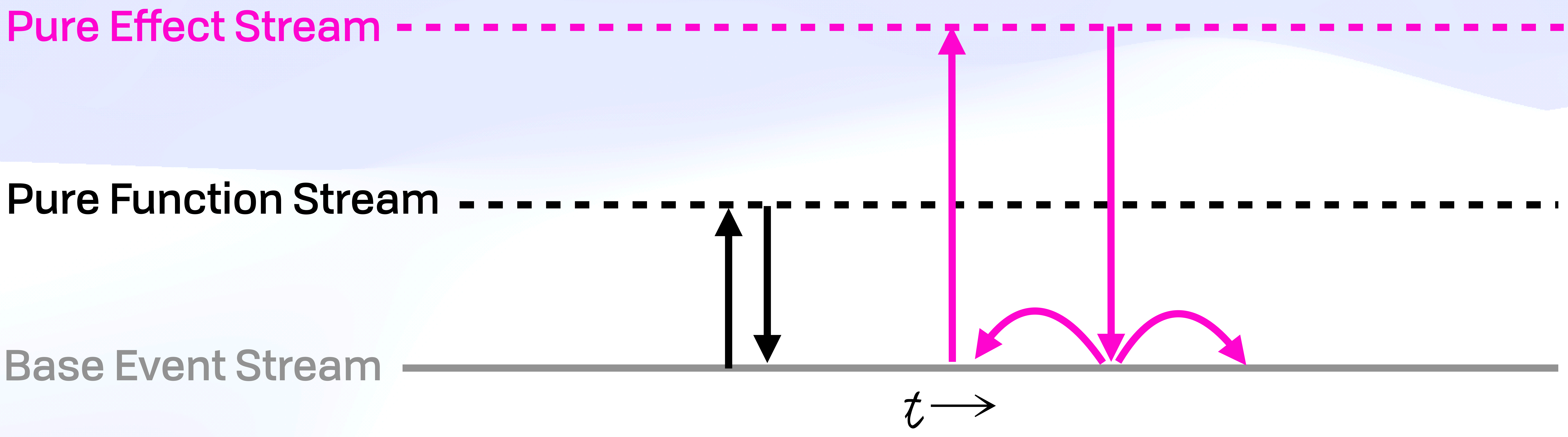
Execution-as-IPLD

Job Streams (Scheduler + Events)



Execution-as-IPLD

Job Streams (Scheduler + Events)

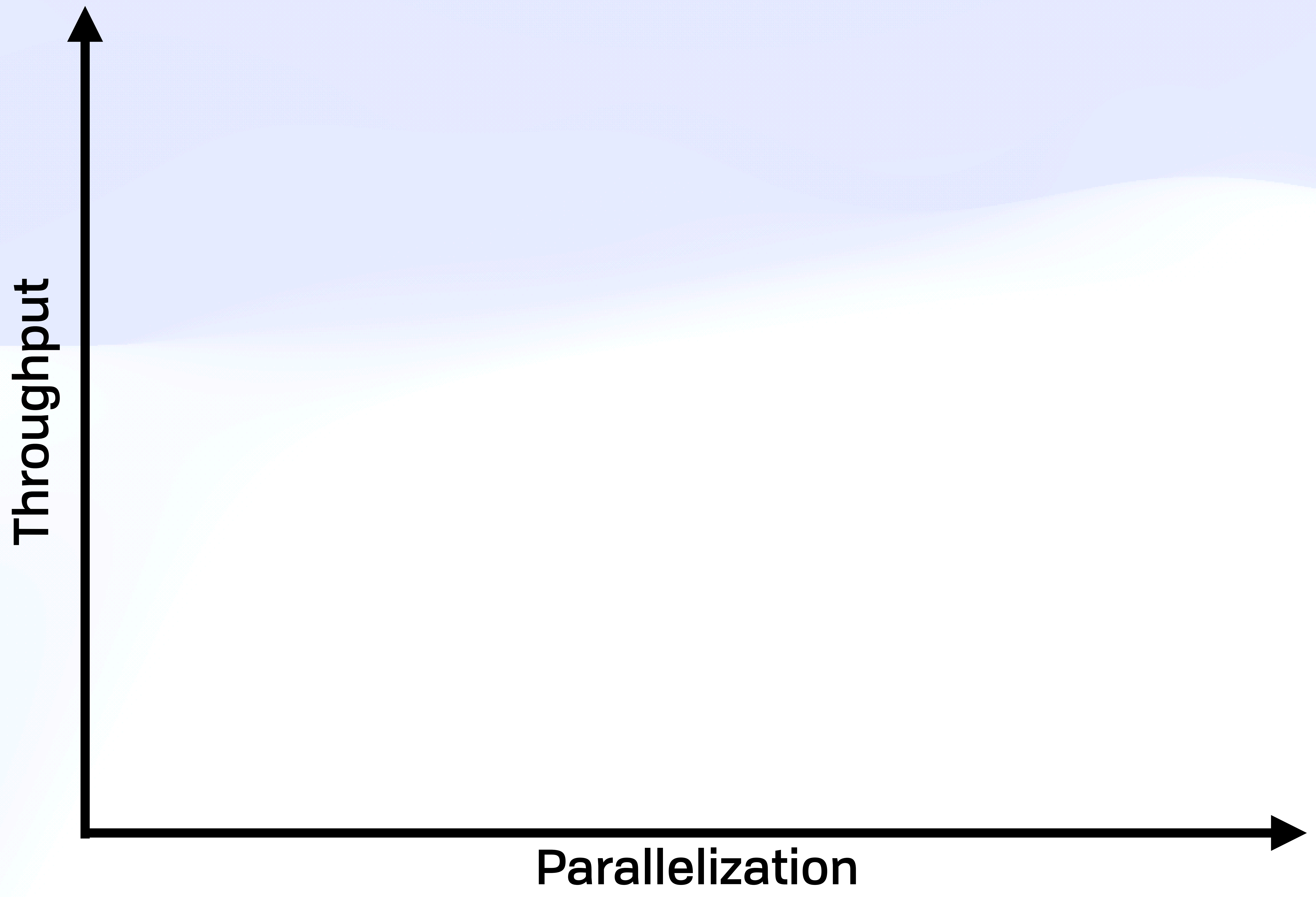


Execution-as-IPLD

With a Little Scale From My Friends

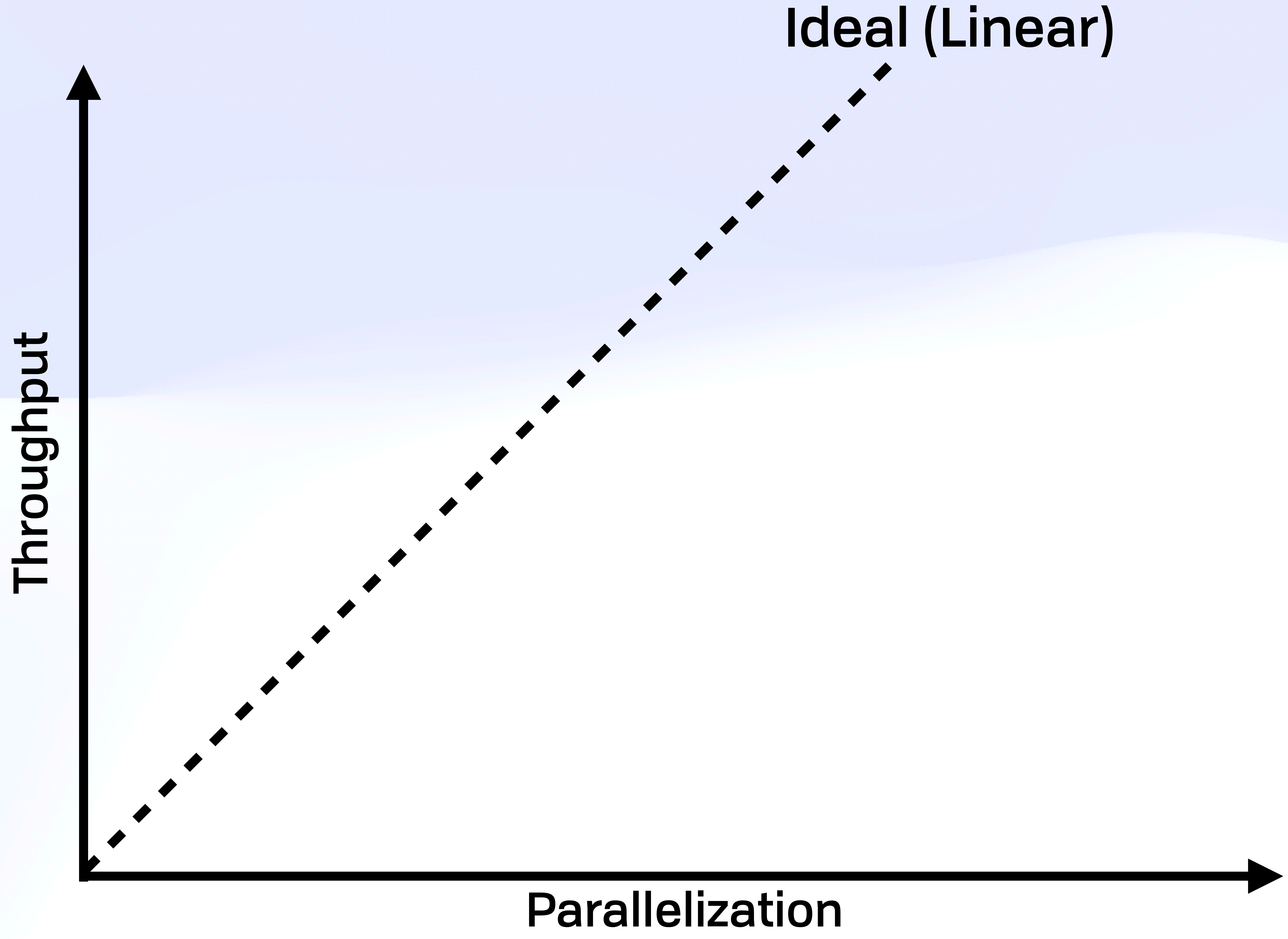
Execution-as-IPLD

With a Little Scale From My Friends



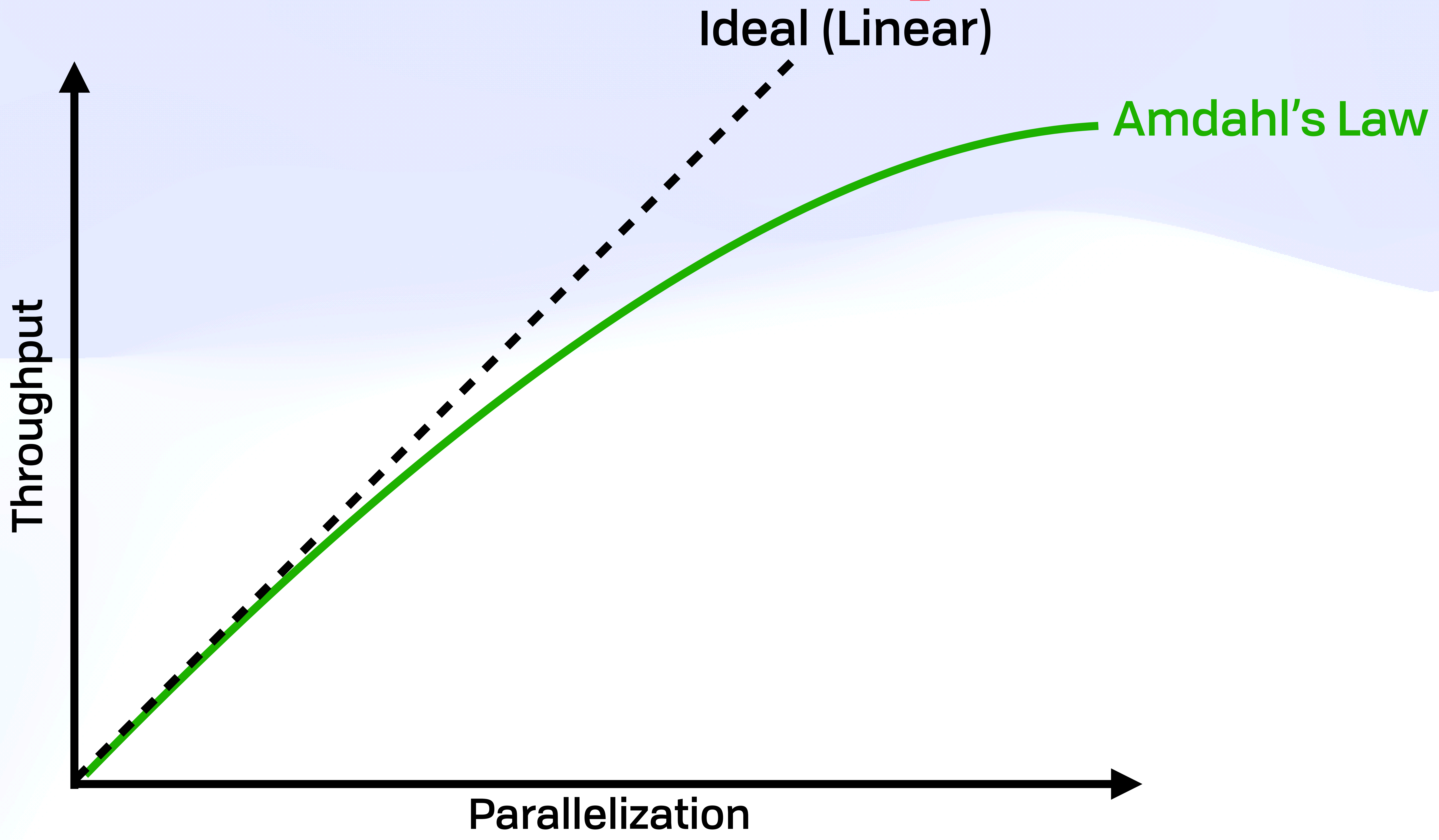
Execution-as-IPLD

With a Little Scale From My Friends



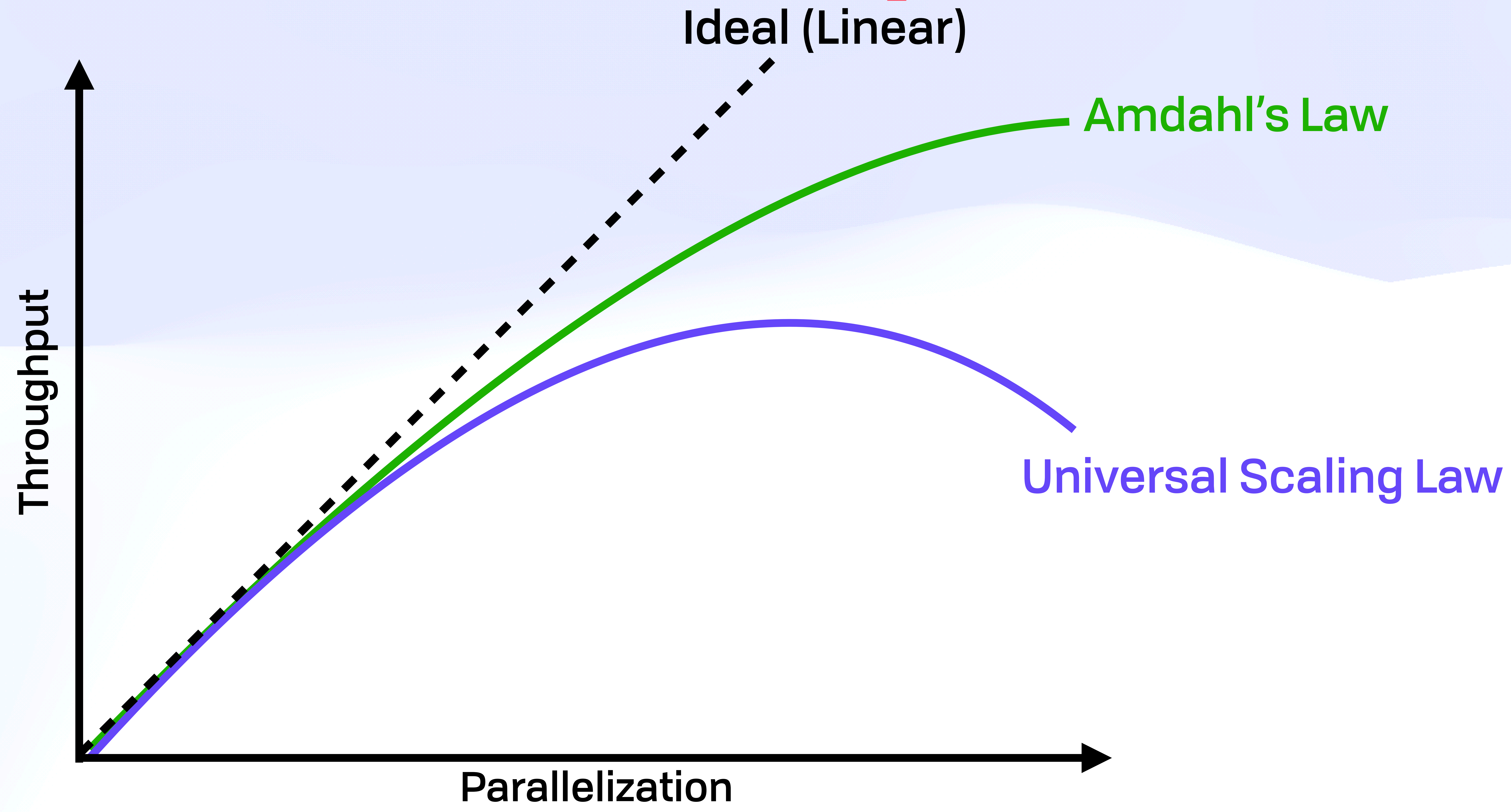
Execution-as-IPLD

With a Little Scale From My Friends



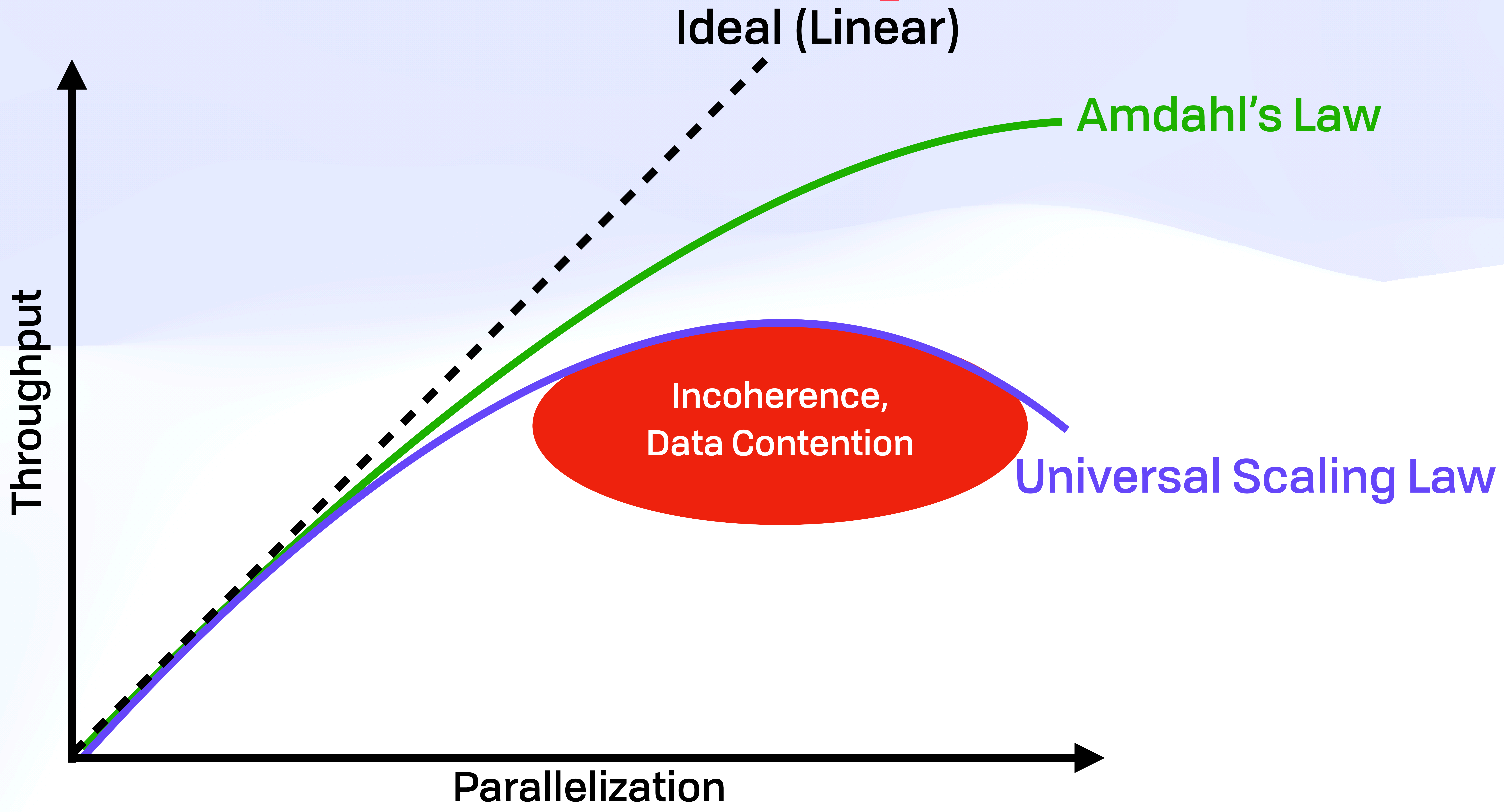
Execution-as-IPLD

With a Little Scale From My Friends



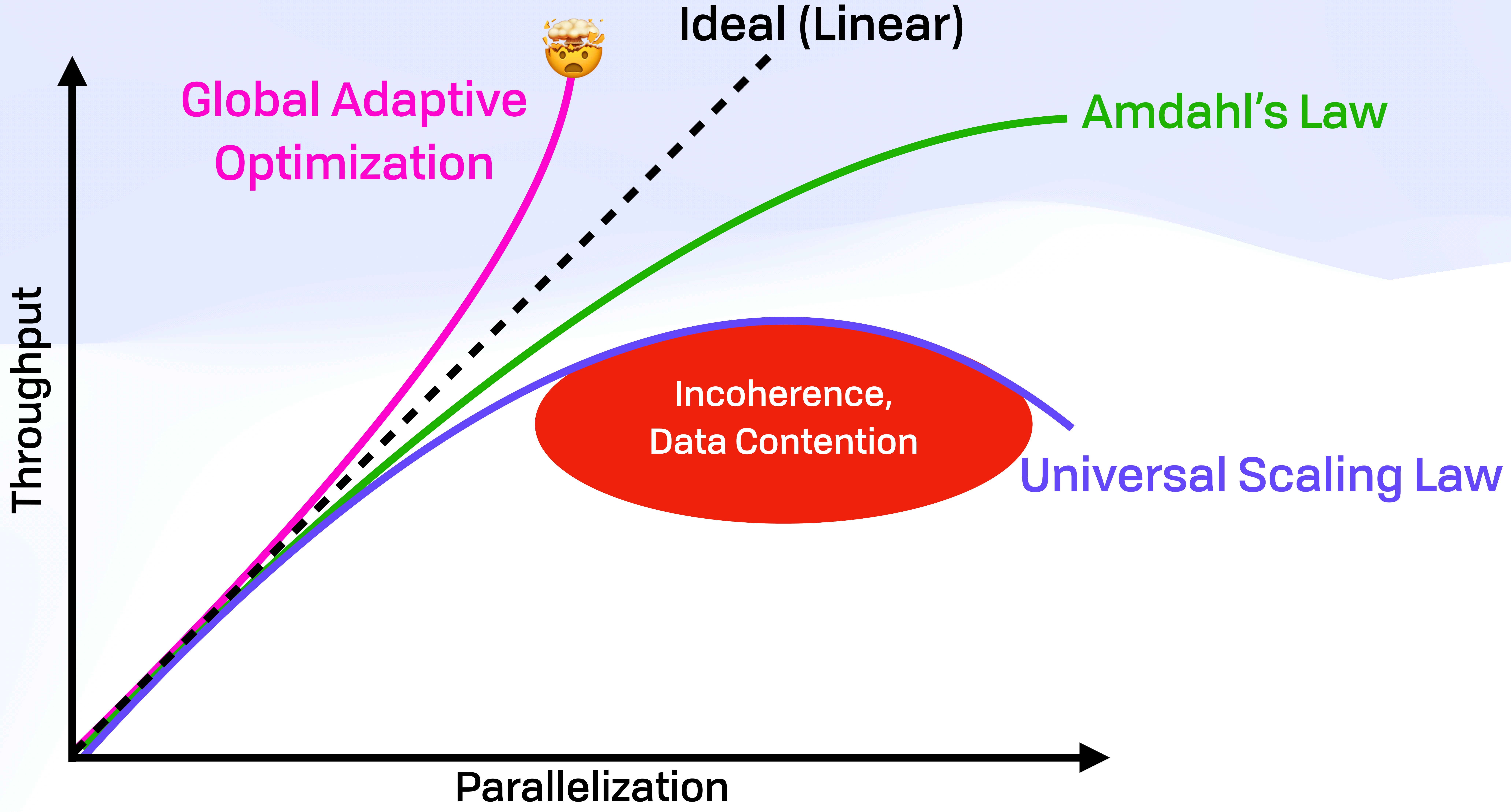
Execution-as-IPLD

With a Little Scale From My Friends



Execution-as-IPLD

With a Little Scale From My Friends

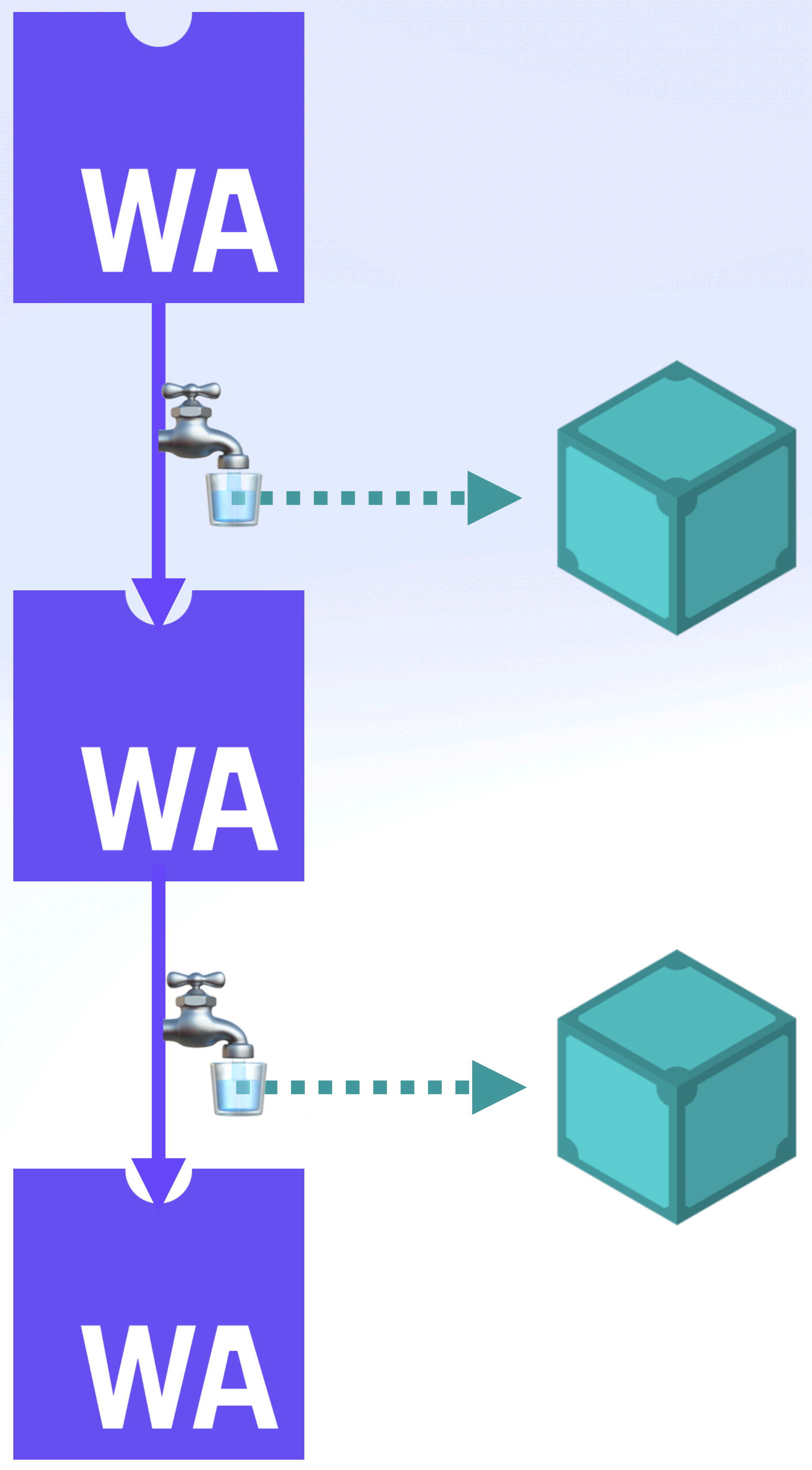


Execution-as-IPLD

Cache Intermediate Results

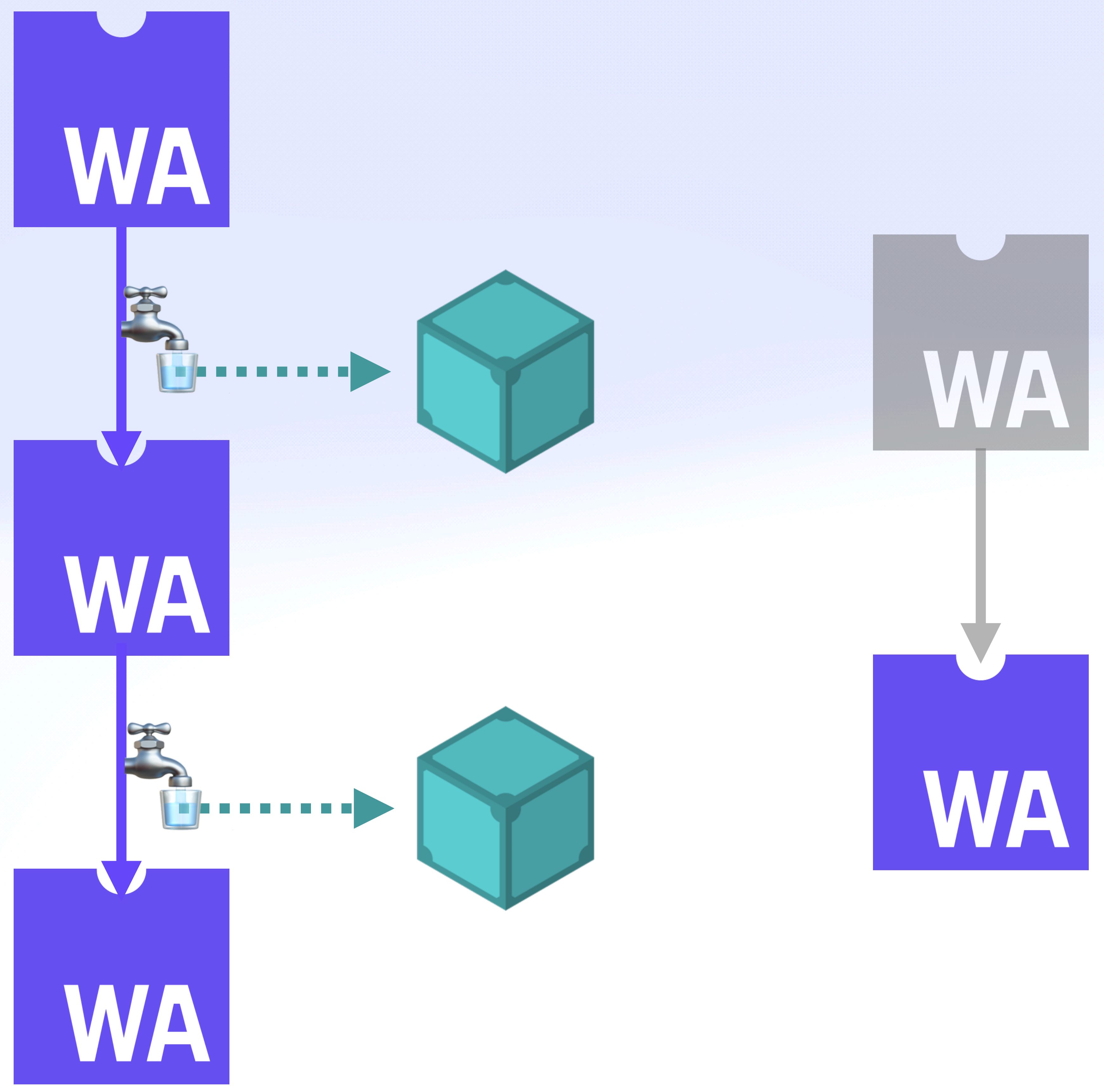
Execution-as-IPLD

Cache Intermediate Results



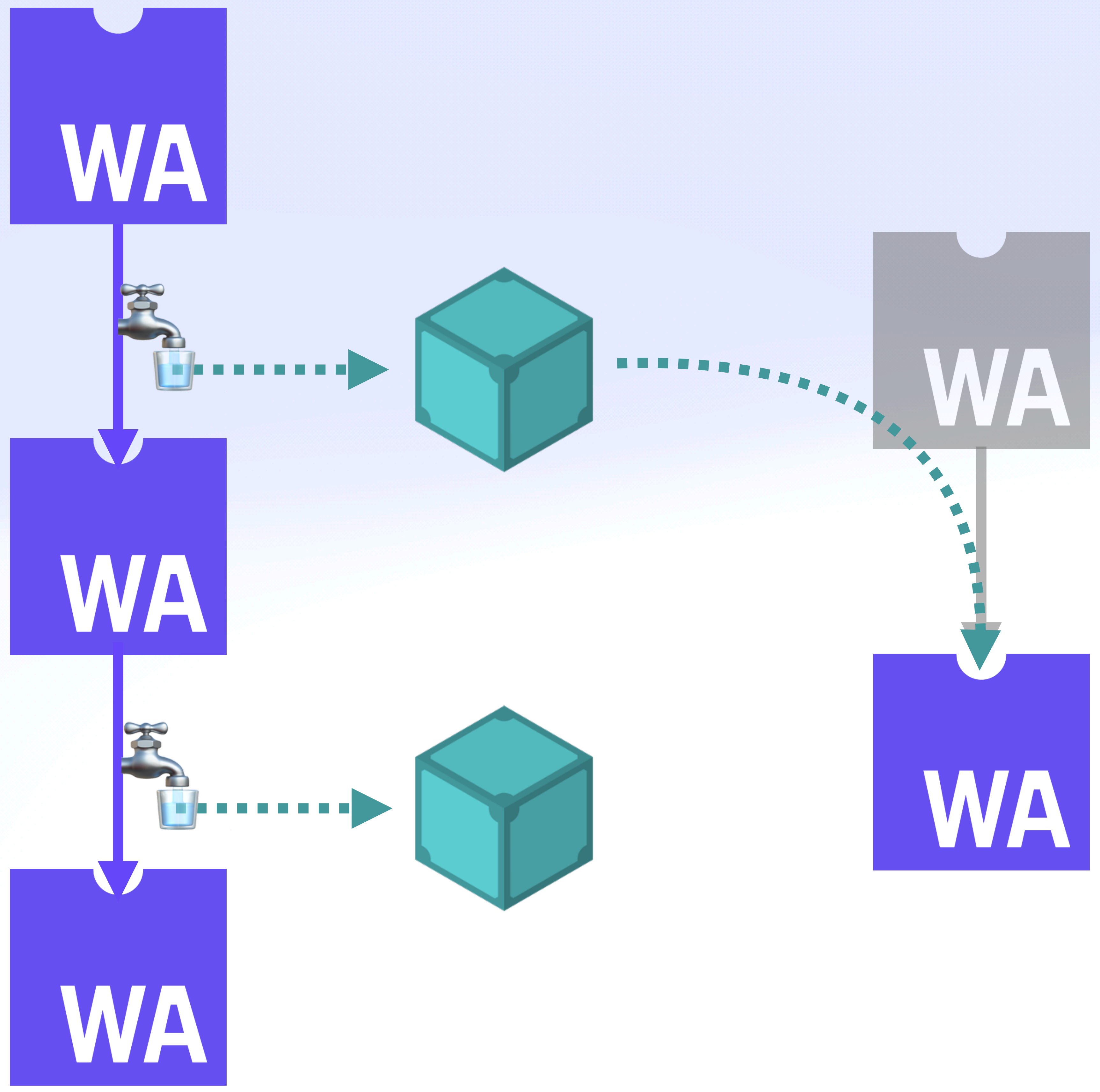
Execution-as-IPLD

Cache Intermediate Results



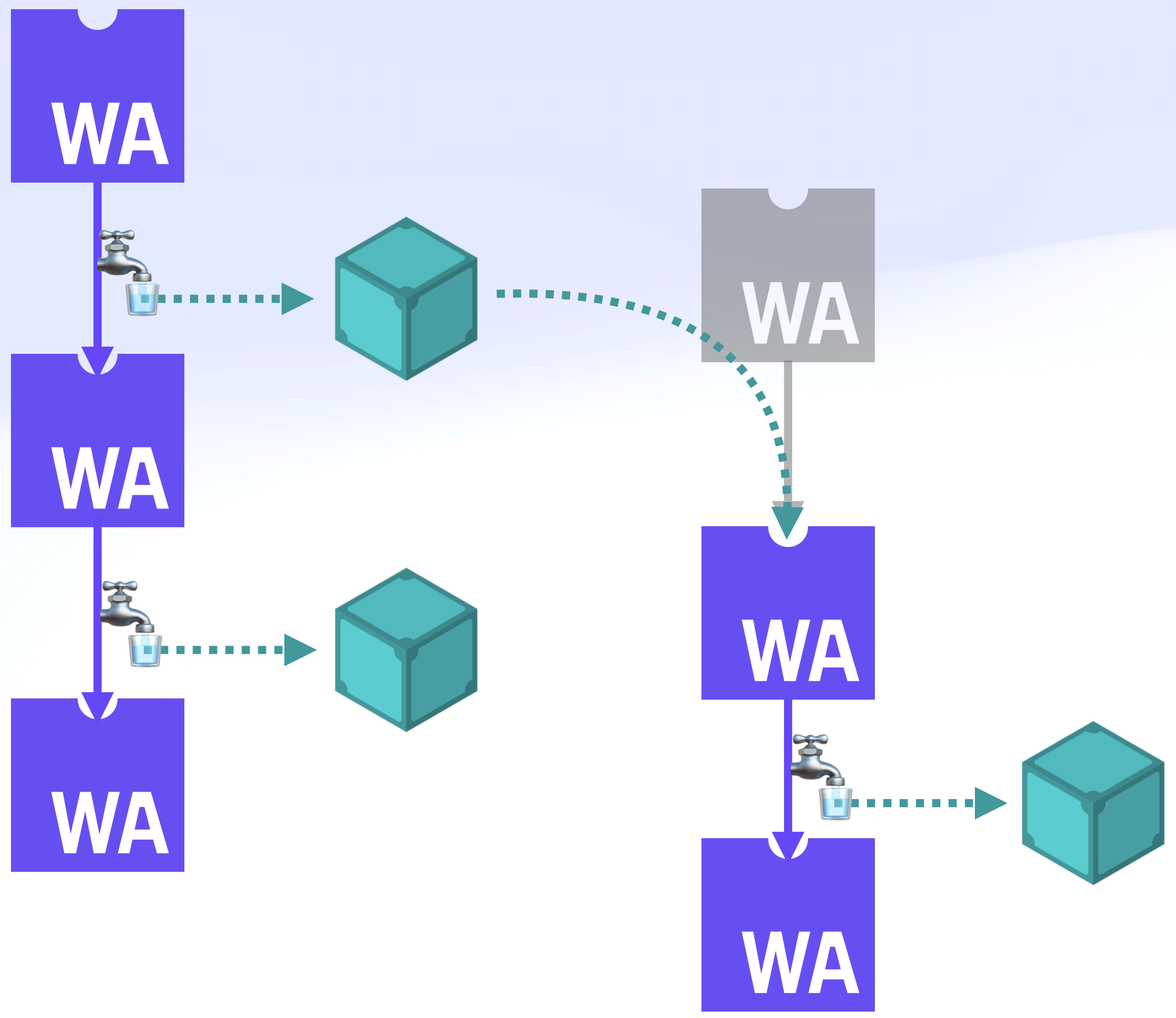
Execution-as-IPLD

Cache Intermediate Results



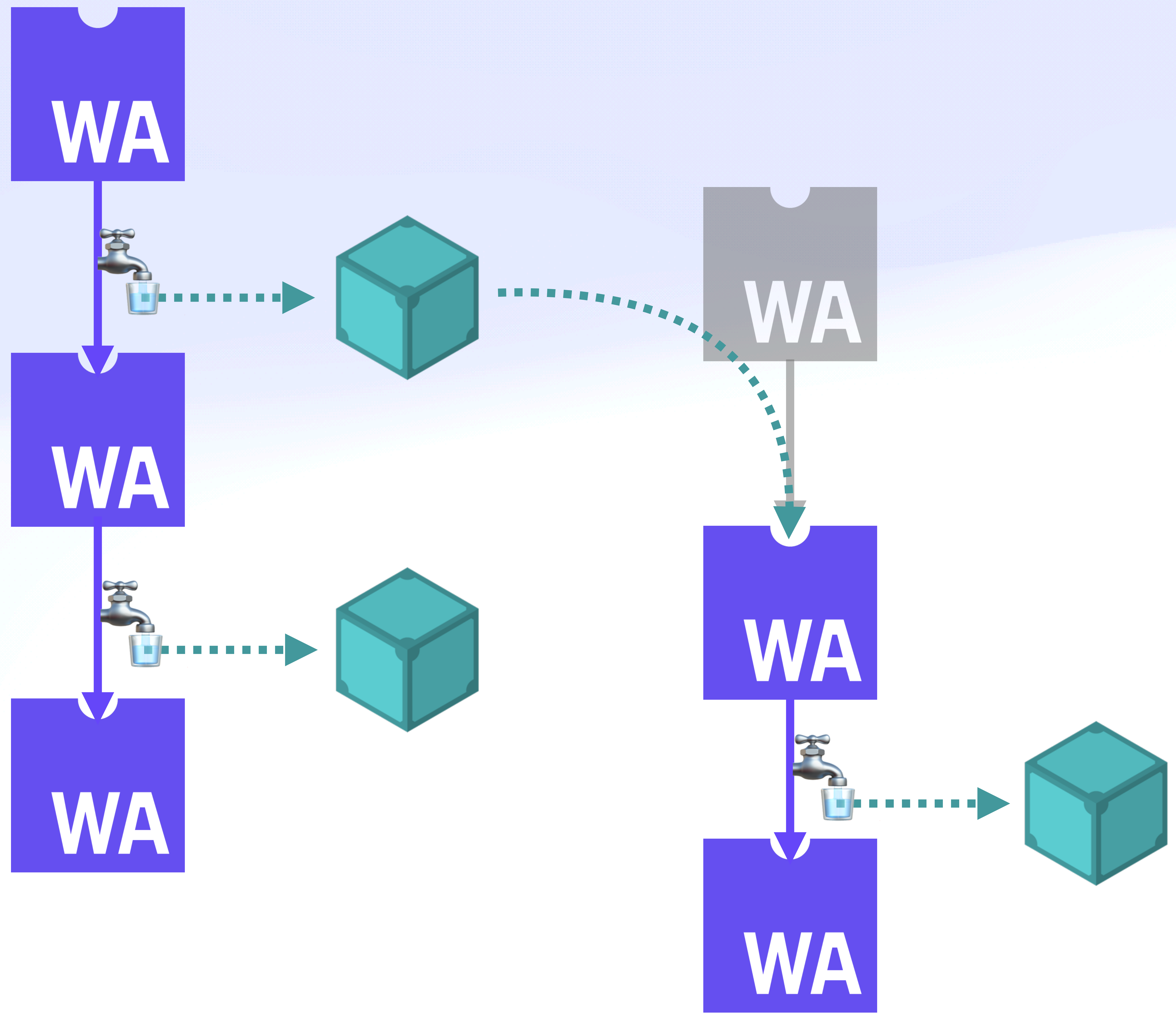
Execution-as-IPLD

Cache Intermediate Results



Execution-as-IPLD

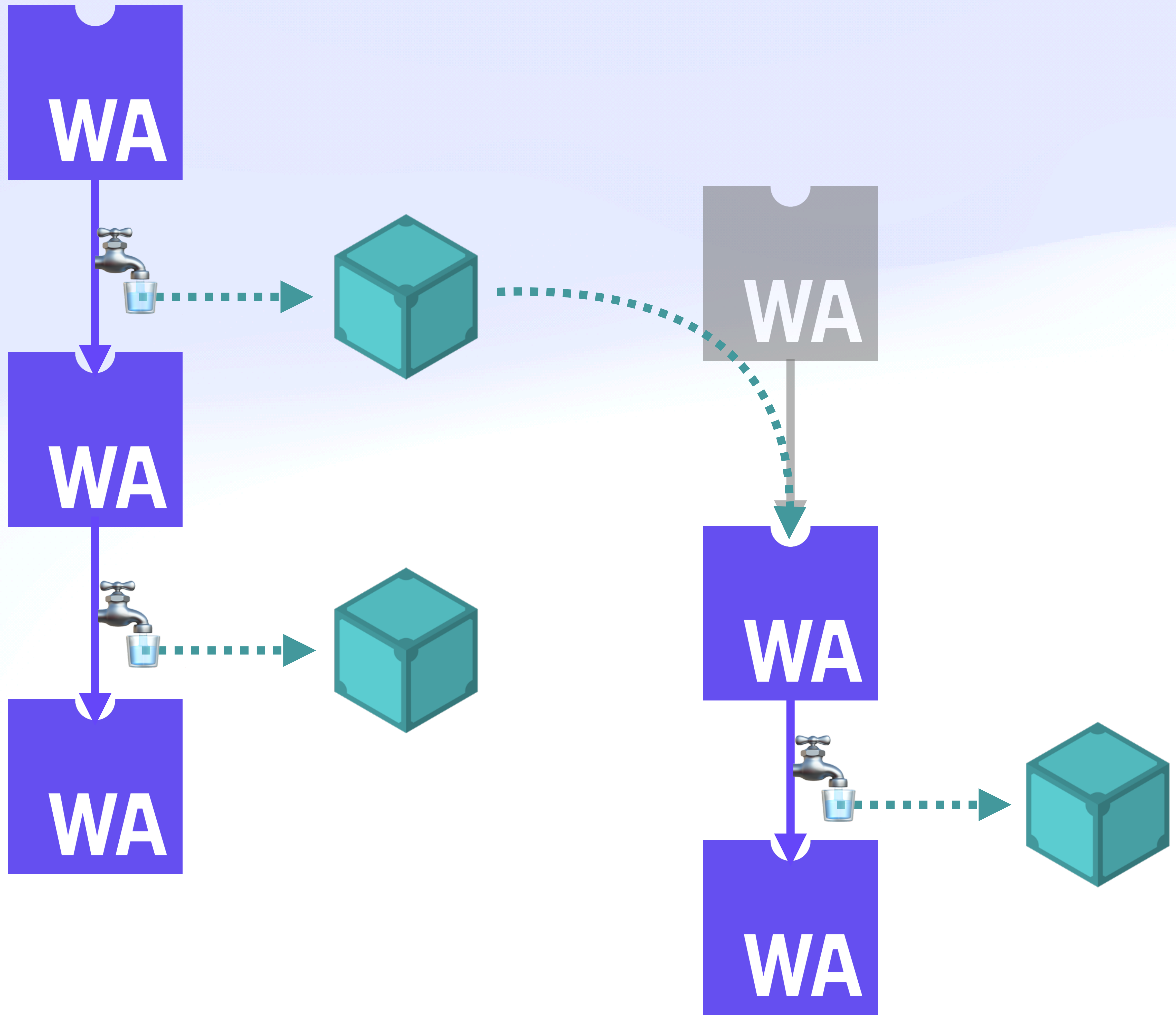
Cache Intermediate Results



Suspend / resume

Execution-as-IPLD

Cache Intermediate Results



Suspend / resume

LURK

Execution-as-IPLD

Feedback Loop

Execution-as-IPLD

Feedback Loop

1. Fetch data

Execution-as-IPLD
Feedback Loop

1. Fetch data
2. Compute on data

Execution-as-IPLD

Feedback Loop

1. Fetch data
2. Compute on data
3. Output more data

Execution-as-IPLD

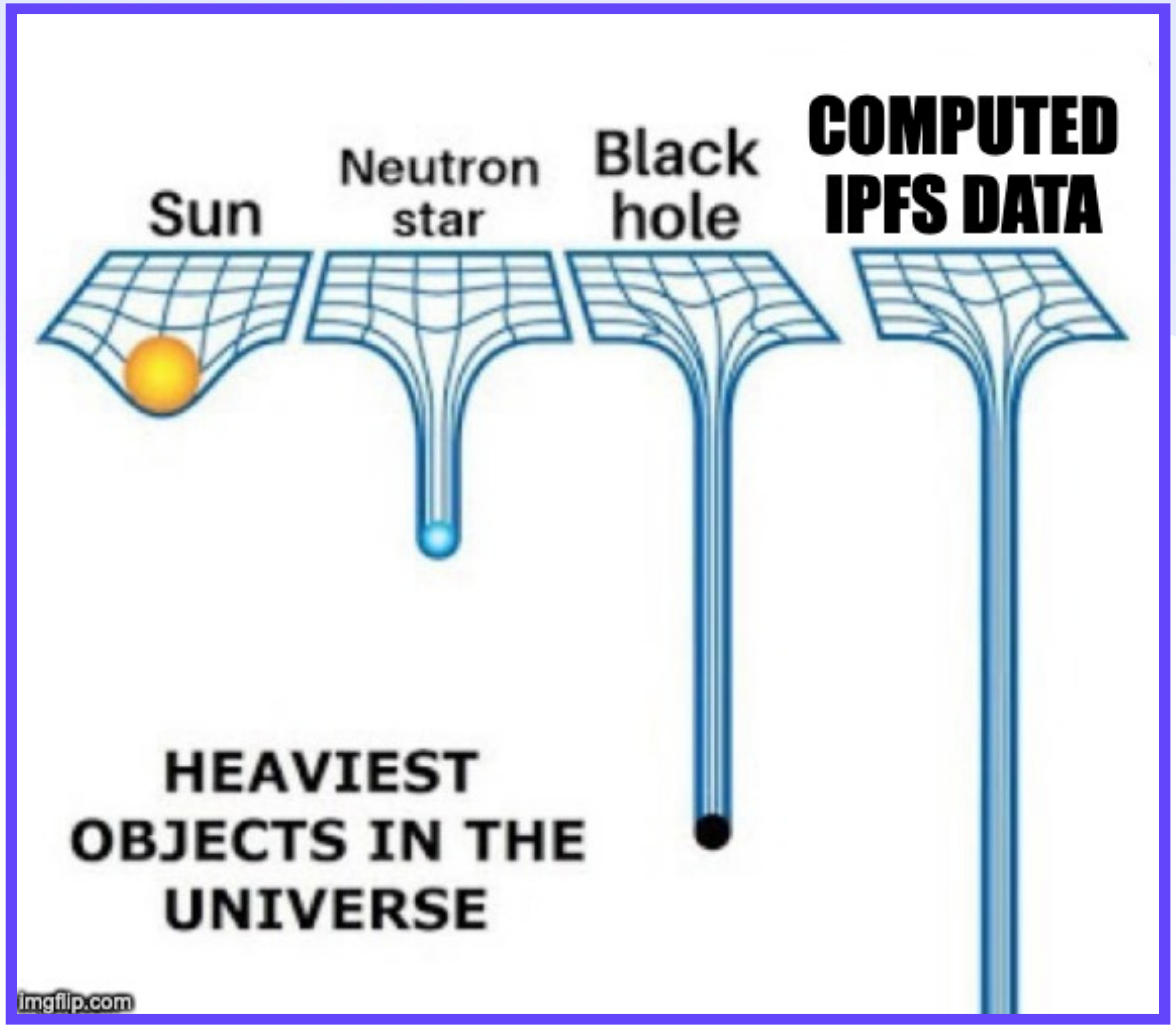
Feedback Loop

1. Fetch data
2. Compute on data
3. Output more data
4. GOTO 2

Execution-as-IPLD

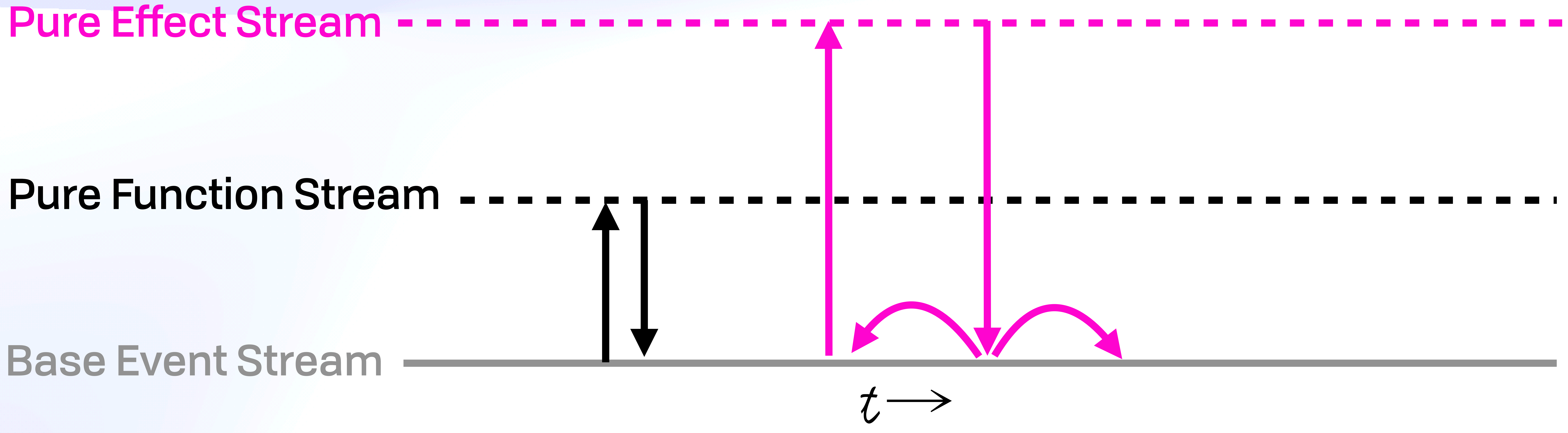
Feedback Loop

1. Fetch data
2. Compute on data
3. Output more data
4. GOTO 2



Execution-as-IPLD

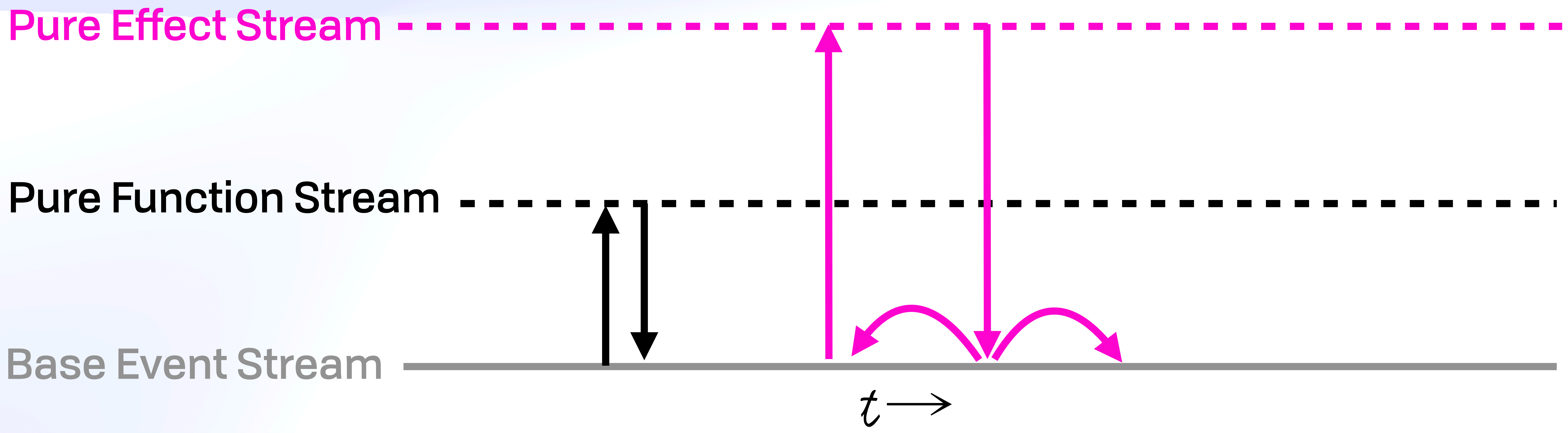
Managed Effects



Execution-as-IPLD

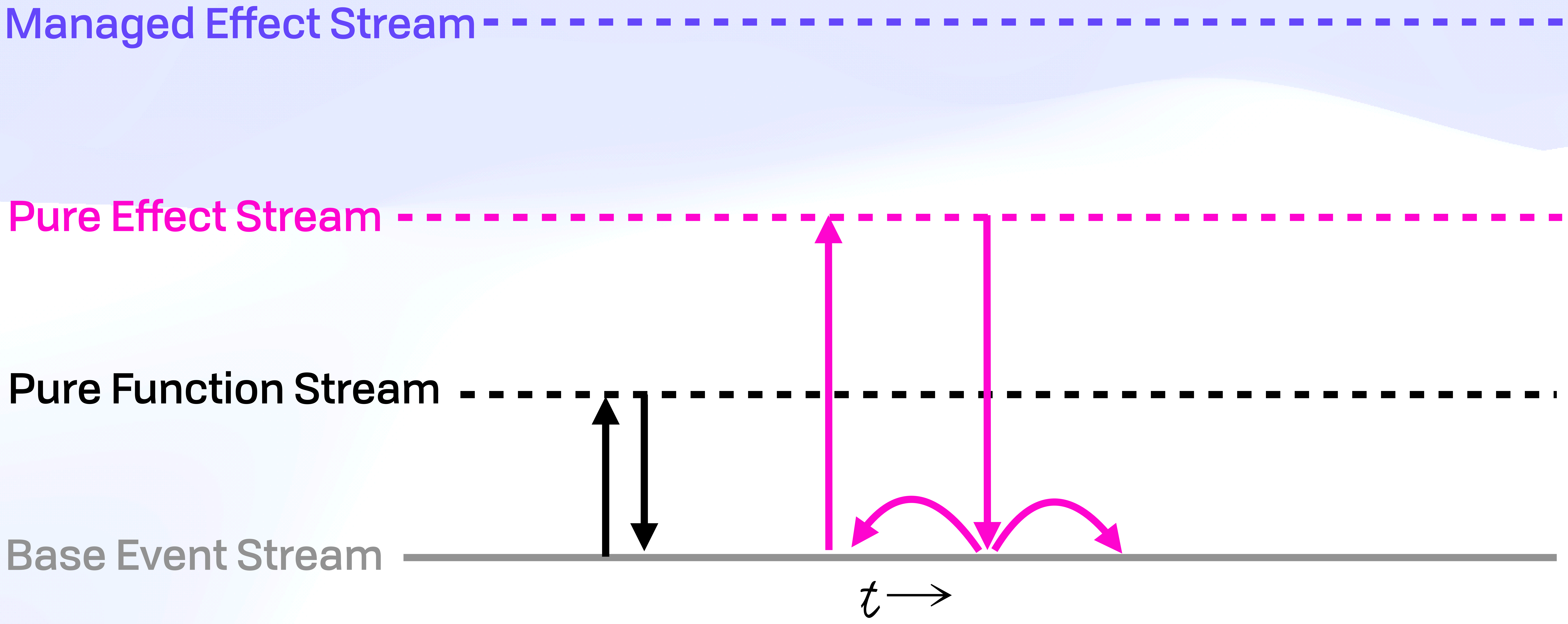
Managed Effects

Managed Effect Stream



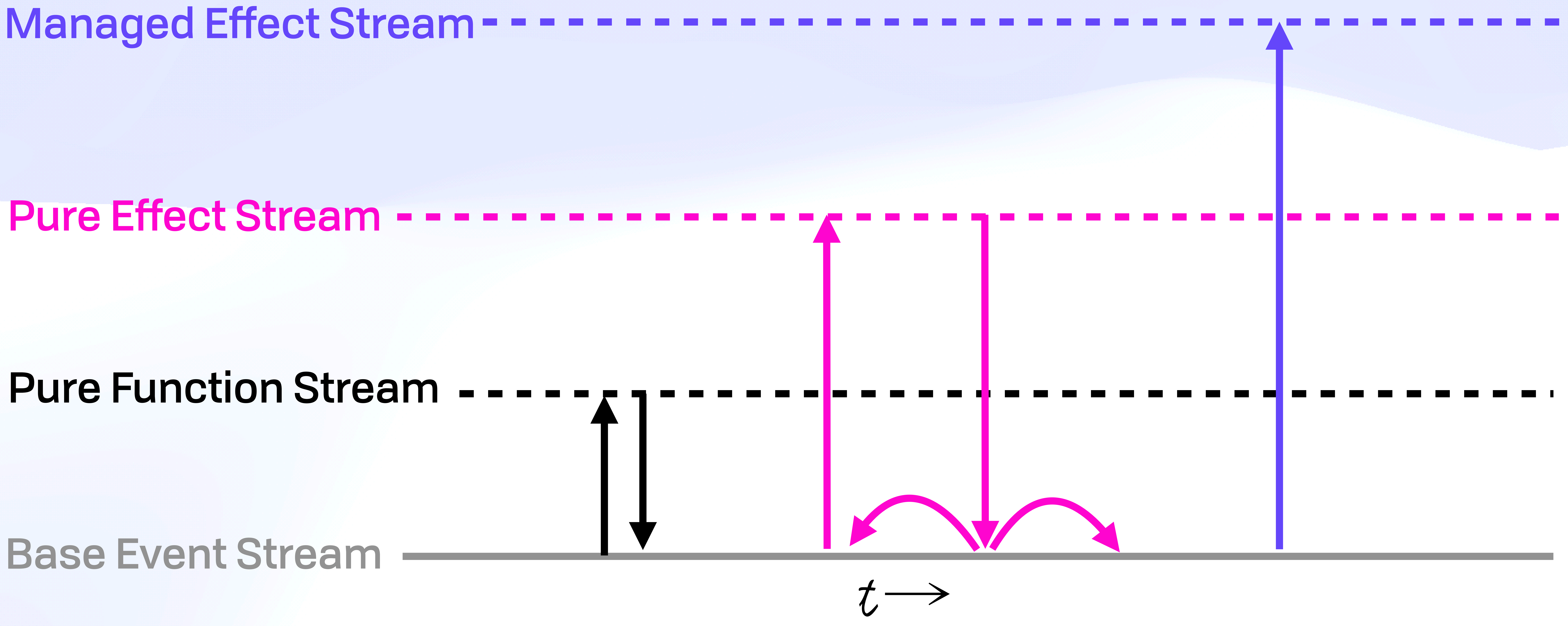
Execution-as-IPLD

Managed Effects



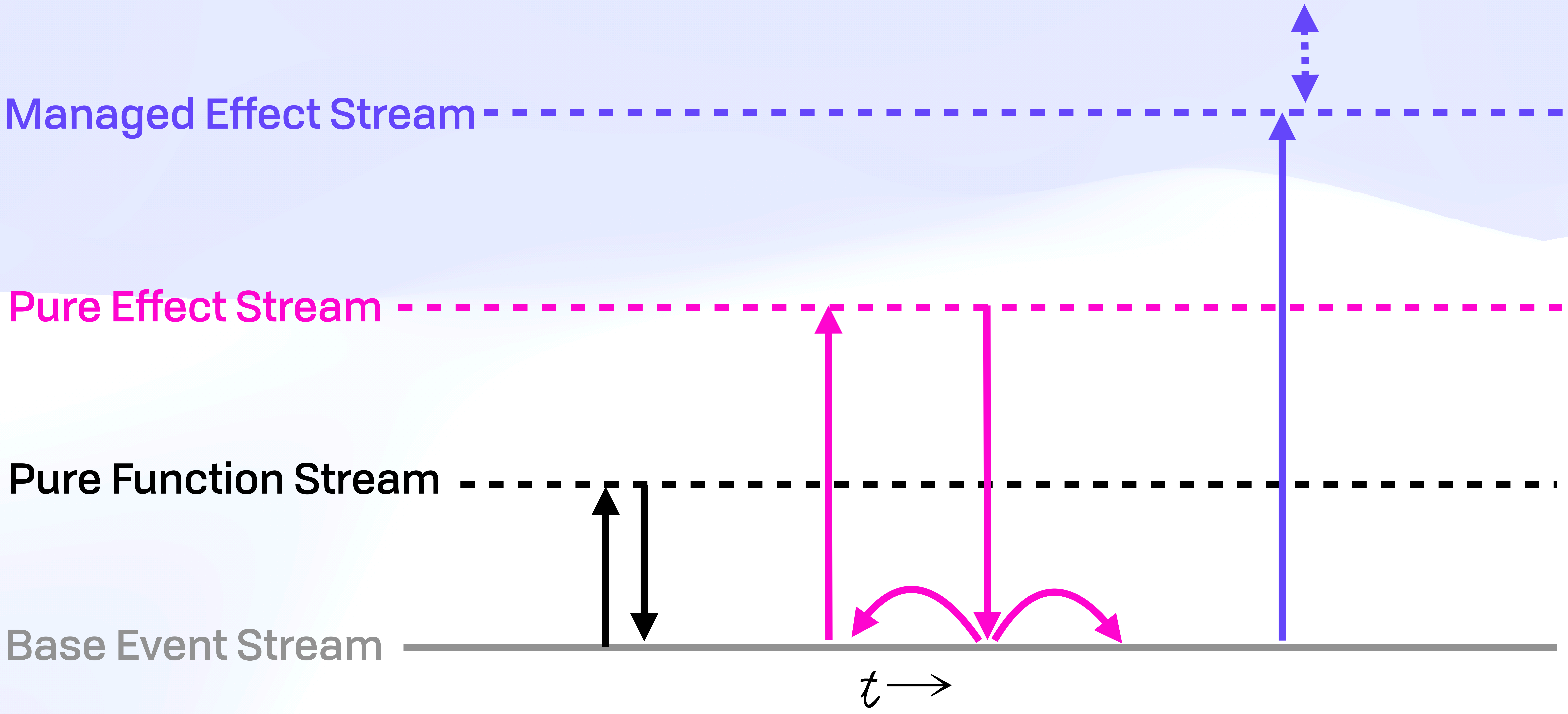
Execution-as-IPLD

Managed Effects



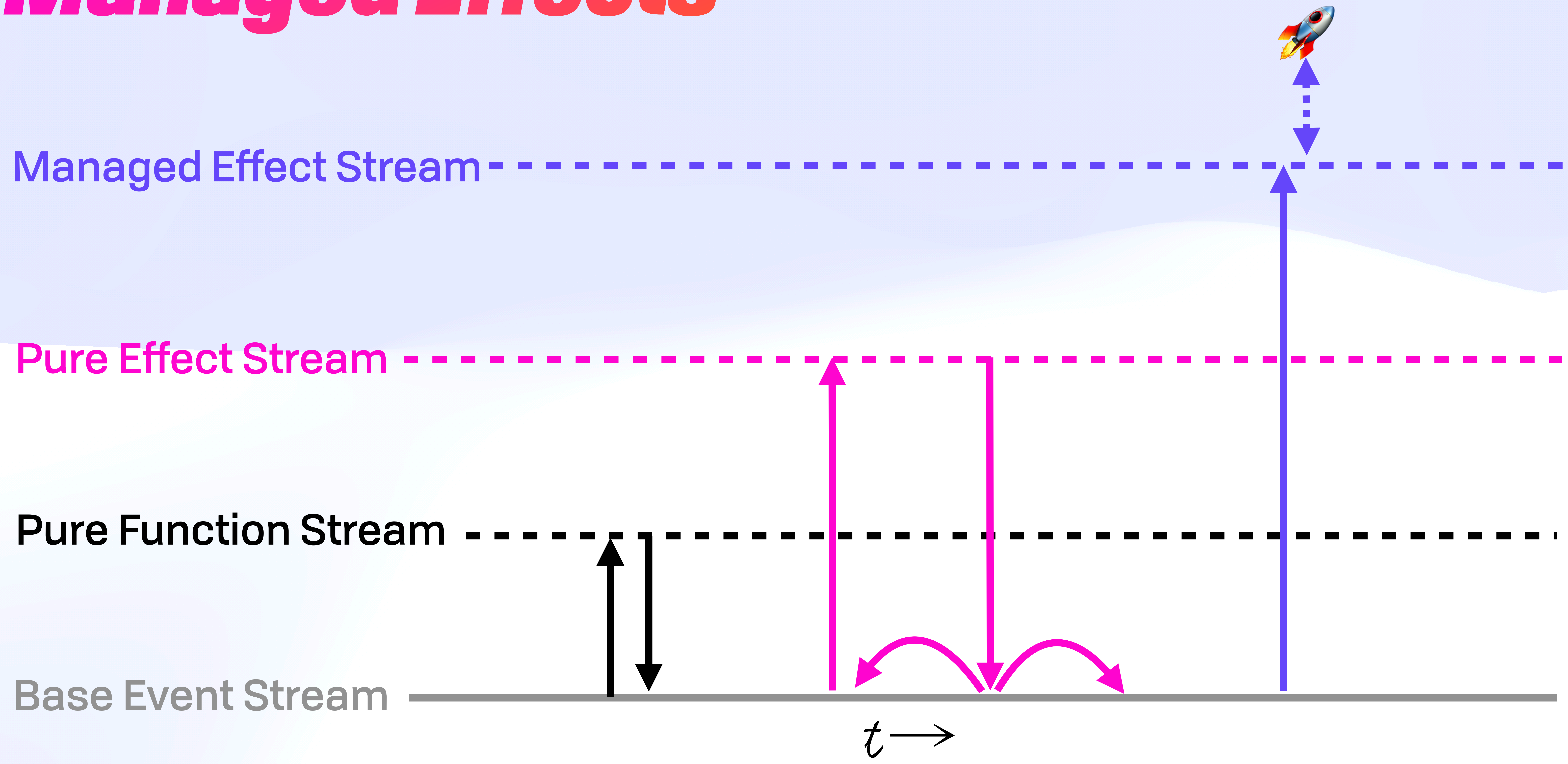
Execution-as-IPLD

Managed Effects



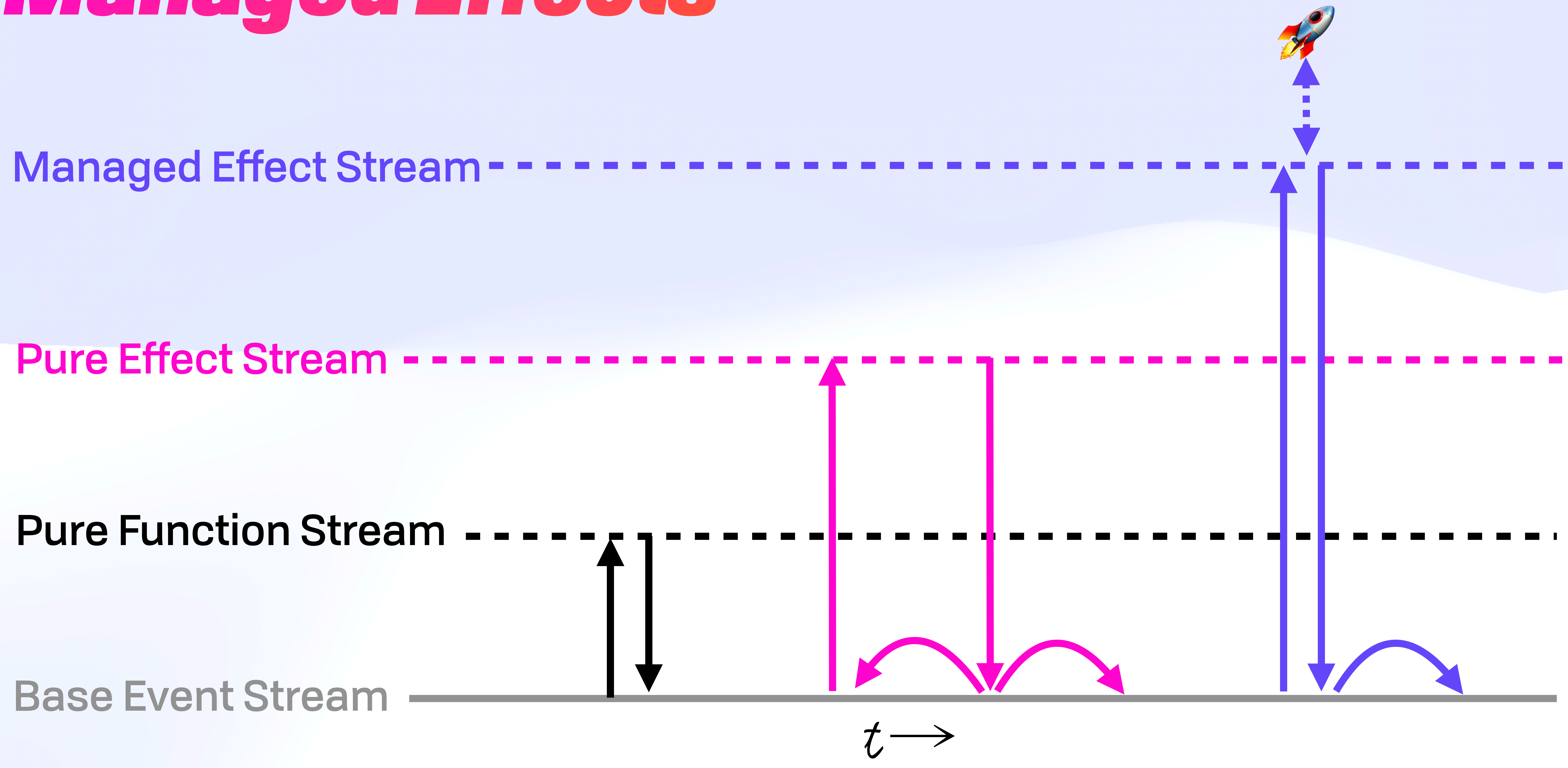
Execution-as-IPLD

Managed Effects



Execution-as-IPLD

Managed Effects



IPVM for IPFS Internals

Shipping IPFS in IPFS

IPVM for IPFS Internals

Content Addressing IPFS Itself

IPVM for IPFS Internals

Content Addressing IPFS Itself

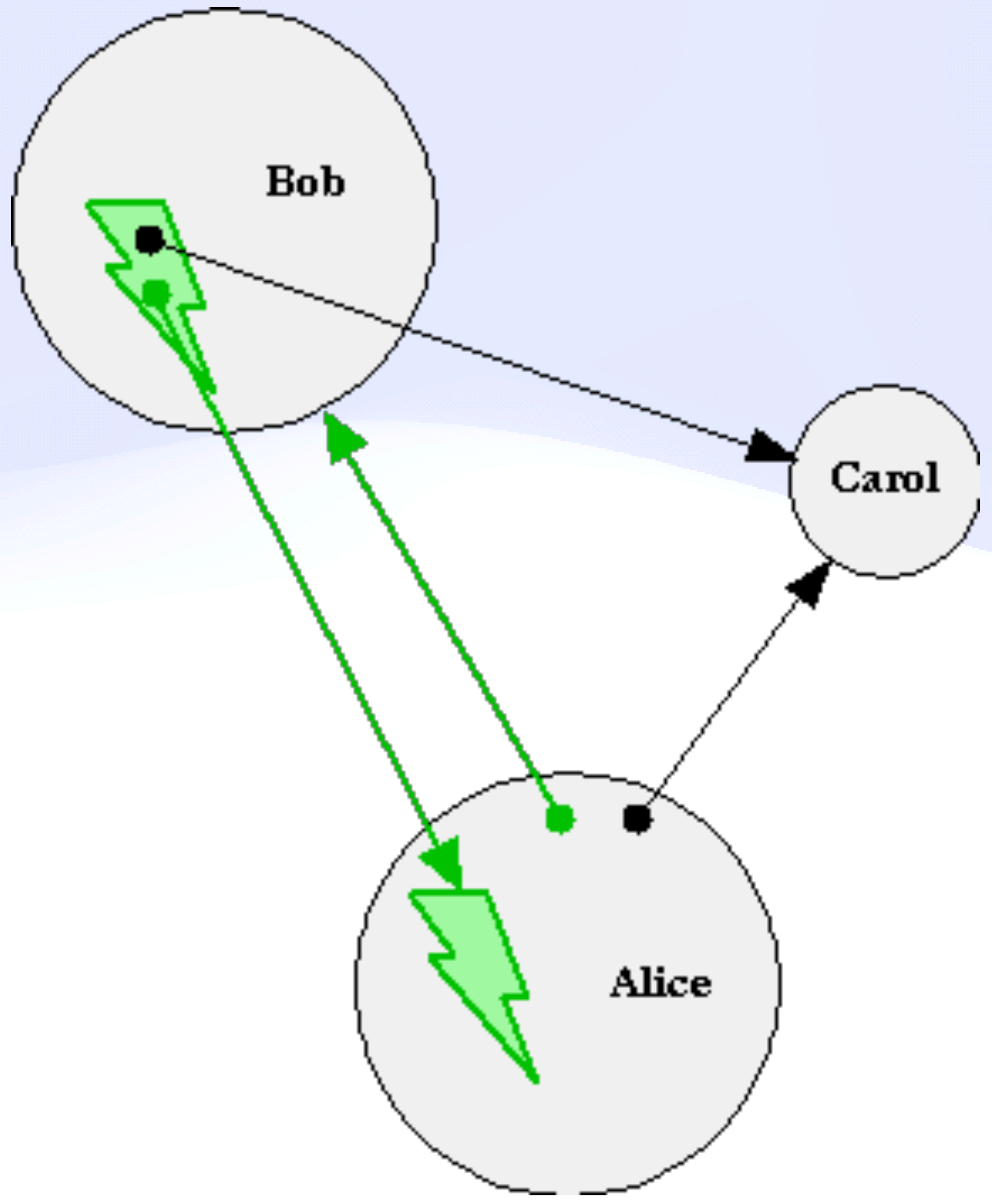
- New codecs (autocodec)
- Cryptography
- Smarter chunkers, incremental verifiers
- Critical bugfixes
- **Share effort between projects (Kubo, Iroh, UCAN, WNFS, Skip Ratchet, etc)**

Security

How Trusted is Your Execution?

Security
Models

- Mobile computing
- OCAP, eRights, encryption, UCAN



http://erights.org/elib/concurrency/msg-passing.html

Security

More Capabilities, More Problems

Security

More Capabilities, More Problems

- Minimum Viable Capability
- Option to switch on more powerful, trusted features
- Remote capabilities on other people's systems (service providers, peers, etc)

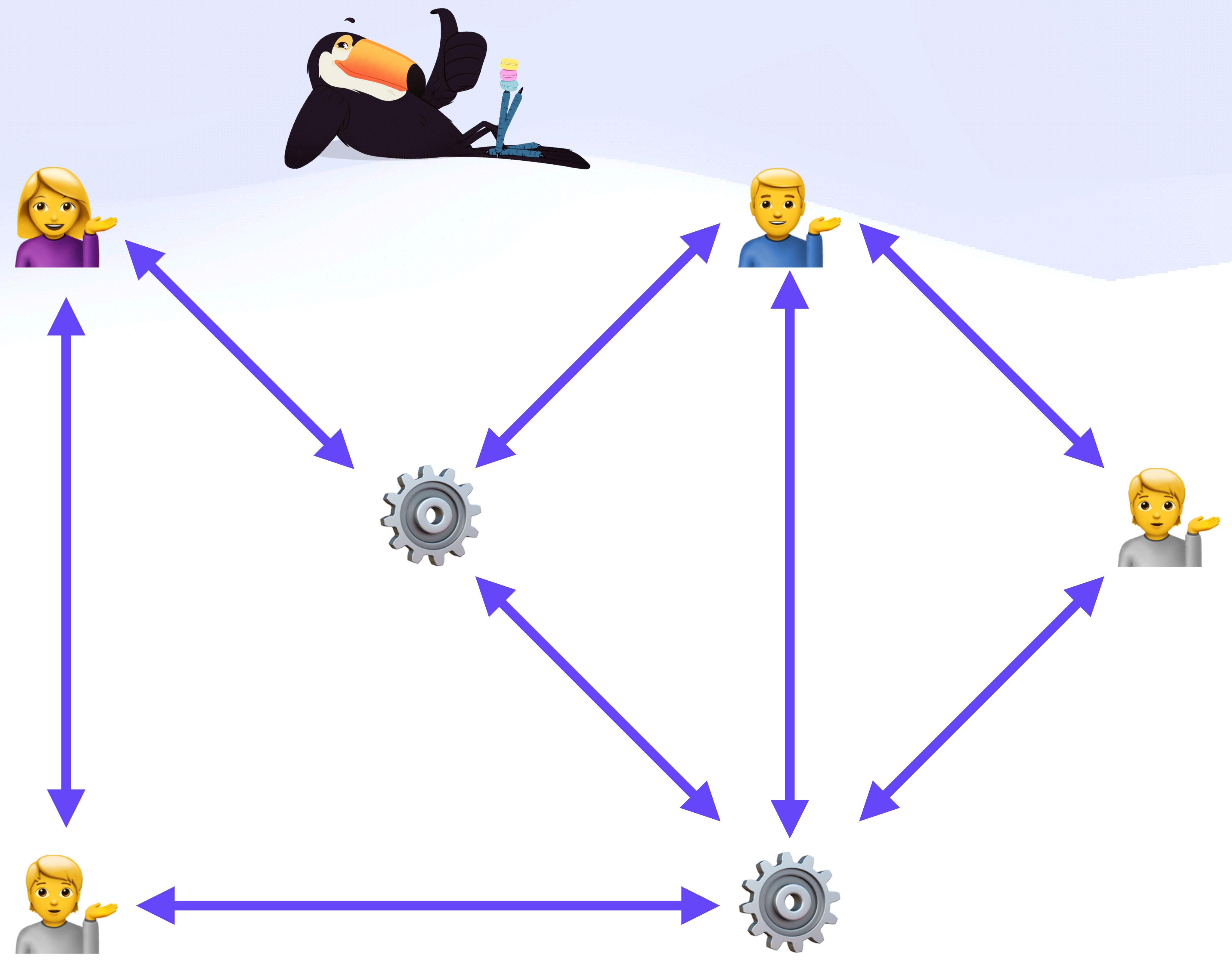
State Channels

Remote Deals & Execution

State Channels

Eager Job Discovery

- Discover providers ("matchmaking DHT")
- Register providers, just like a bootstrap list
- (Optional) reputation



State Channels

Payments

State Channels

Payments

- Reuse generalized state channels for (optional) payment, reputation, etc
    - (Future: hierarchical consensus)

Where to Start

🗺️

Where to Start

Bootstrapping First Steps: Brass Tacks

Where to Start

Bootstrapping First Steps: Brass Tacks

- Ship Wasm into an IPFS implementation
- Manual invocation from CLI
- IPLI format experimentation
- Concurrent job scheduler, trust & resource limits
- Figure out sensible default configs from experience
- Experiment with deeper integration: wasm-ipld or similar
- Cron, event triggers, etc
- Push jobs, associated authZ

Open Discussion 🌶️

Tell me why you ❤️ or 💔 these ideas!

Open Discussion 🌶️

Tell me why you ❤️ or 💔 these ideas!

🎉 Thank You, IPFS þing 🇮🇸

https://fission.codes
brooklyn@fission.codes
@expede

Therefore, ***the worse-is-better software first will [...] be improved to a point that is almost the right thing***. In concrete terms, even though Lisp compilers in 1987 were about as good as C compilers, ***there are many more compiler experts who want to make C compilers better*** than want to make Lisp compilers better.

– Richard P. Gabriel, Lisp: Good News, Bad News, How to Win Big (1991)