Karlsruhe Institute of Technology

Institute of Experimental Particle Physics

Steinbuch Centre for Computing

CMS Germany

BMBF FSP-CMS

Modelling of Heterogeneous Distributed Systems with Coordinated Caches

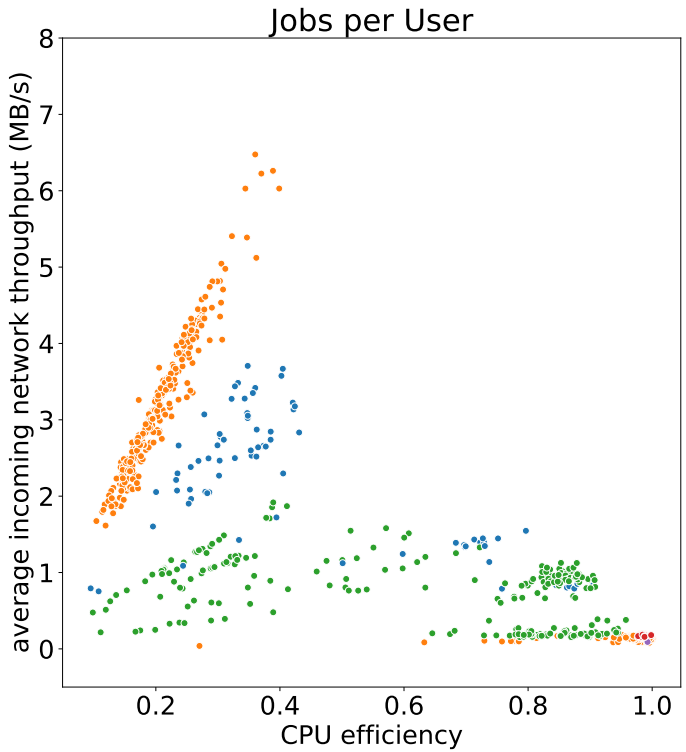
ErUM Data IDT Collaboration Meeting 2021

R.F. von Cube, T. Feßenbecker, M. Fischer, M. Giffels, C. Heidecker, R. Hofsaess, **Maximilian M. Horzela**, E. Kühn, G. Quast, M. Schnepf, P. Skopnik | 11. May 2021

Solving the HEP Computing Challenge

- Proposed solutions for the HEP-Computing-Challenge
  - Software improvements
  - Integration of additional non-HEP resources
  - **Optimization of existing computing model**
- Opportunistic resources $\rightarrow$ Occupation of network increases

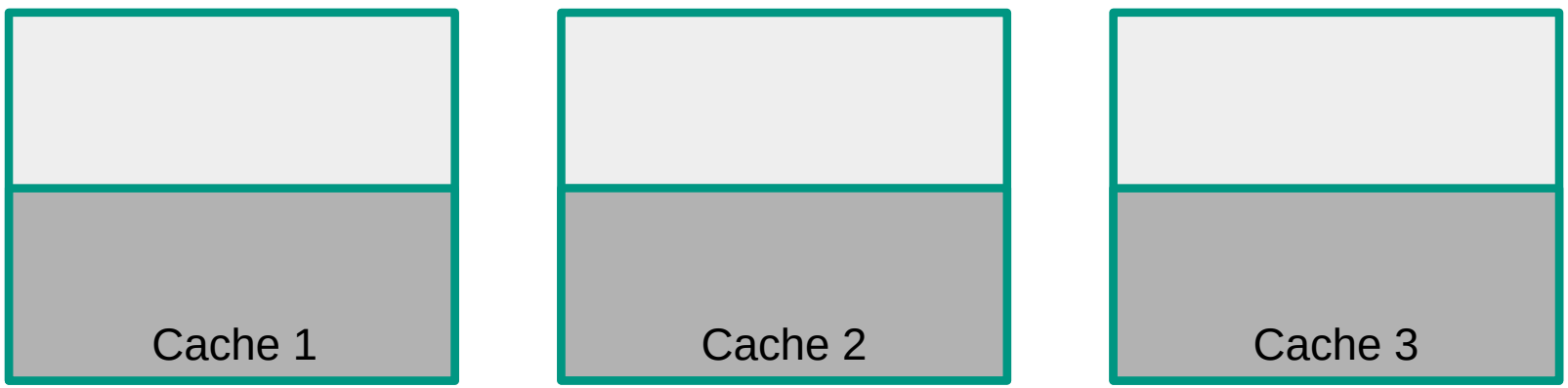
$\Rightarrow$ With existing workflows: improvement only when network is not saturated



M. Schnepf

Data Flow

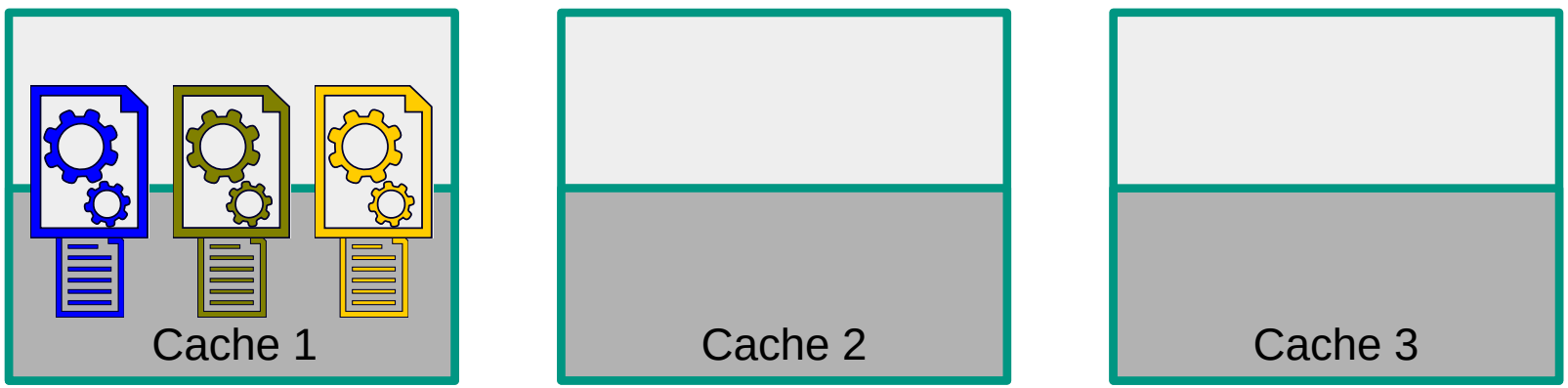
How do we achieve the most efficient data processing with a given resource composition?



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

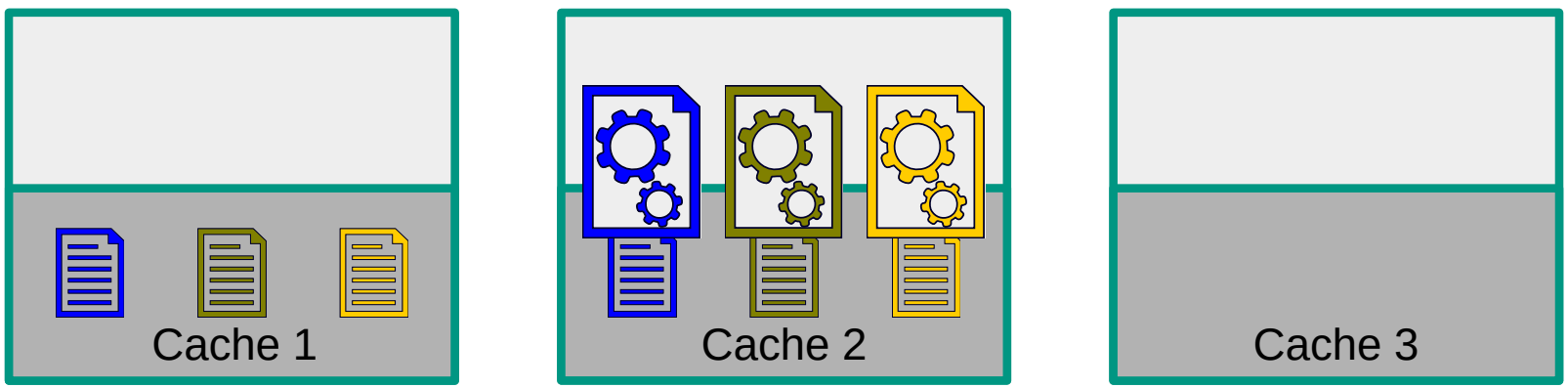
- Jobs starting on a resource and load data into caches



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

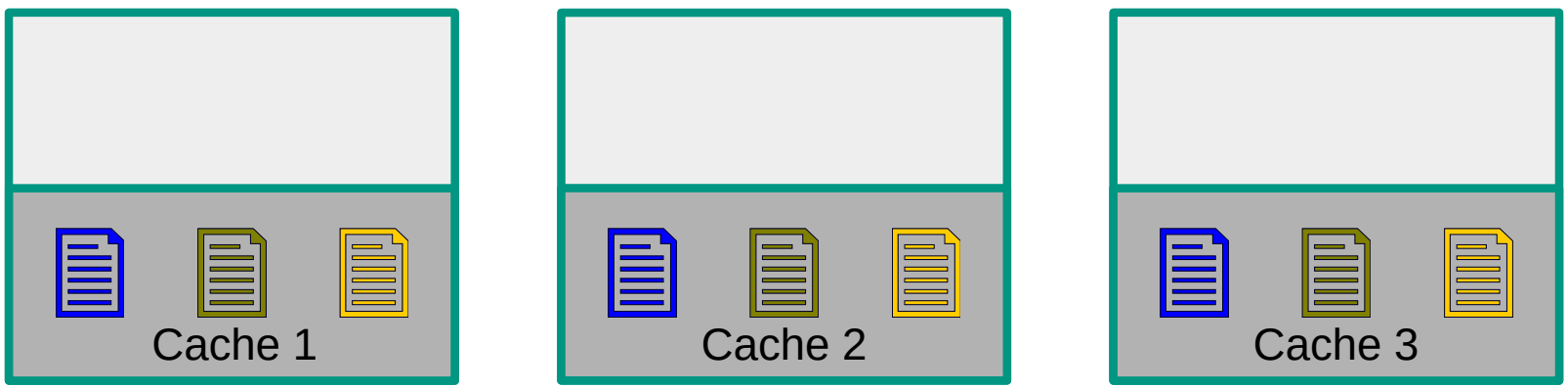
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

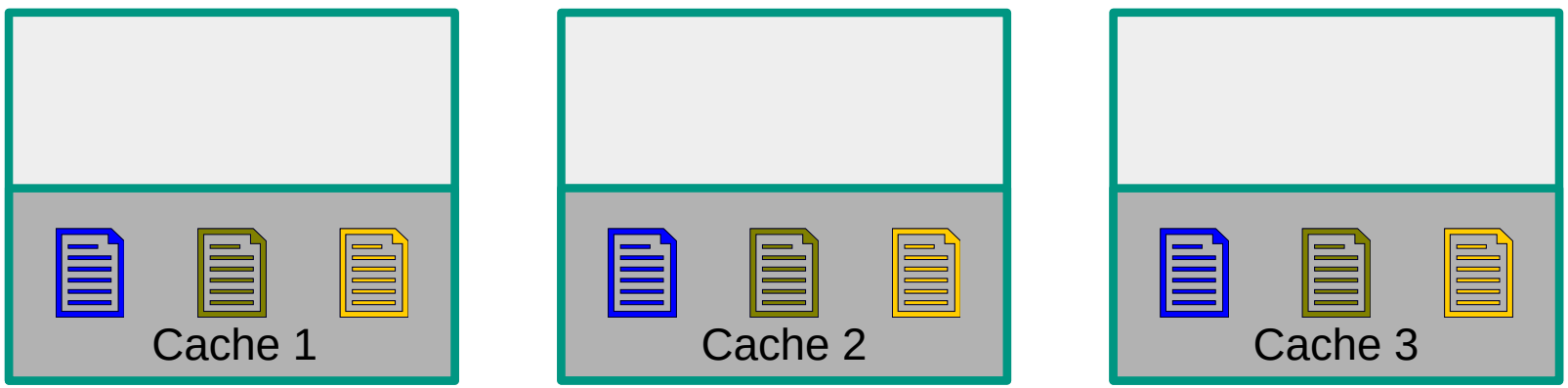
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches
- Continuing until all caches are filled → Replications of same data on different resources



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

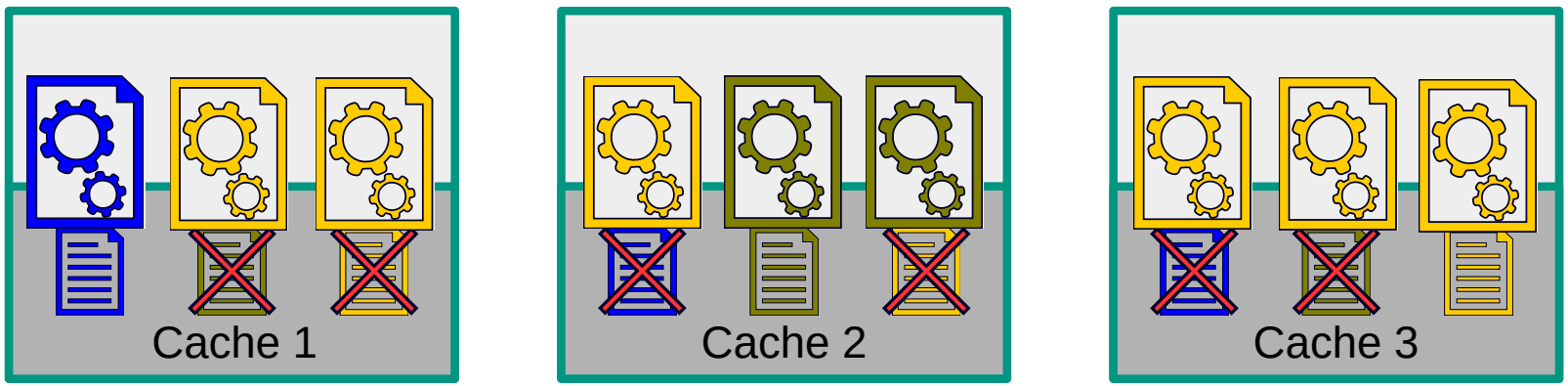
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches
- Continuing until all caches are filled → Replications of same data on different resources
- Coordinate jobs and configure caches ...



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

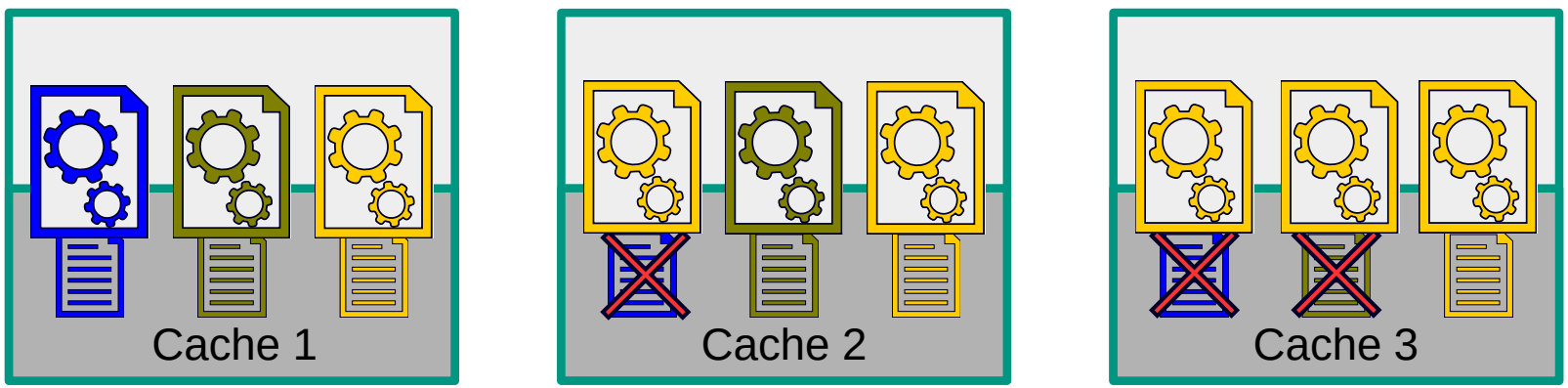
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches
- Continuing until all caches are filled $\rightarrow$ Replications of same data on different resources
- Coordinate jobs and configure caches ...
  - ... to reduce waste of storage capacity?



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

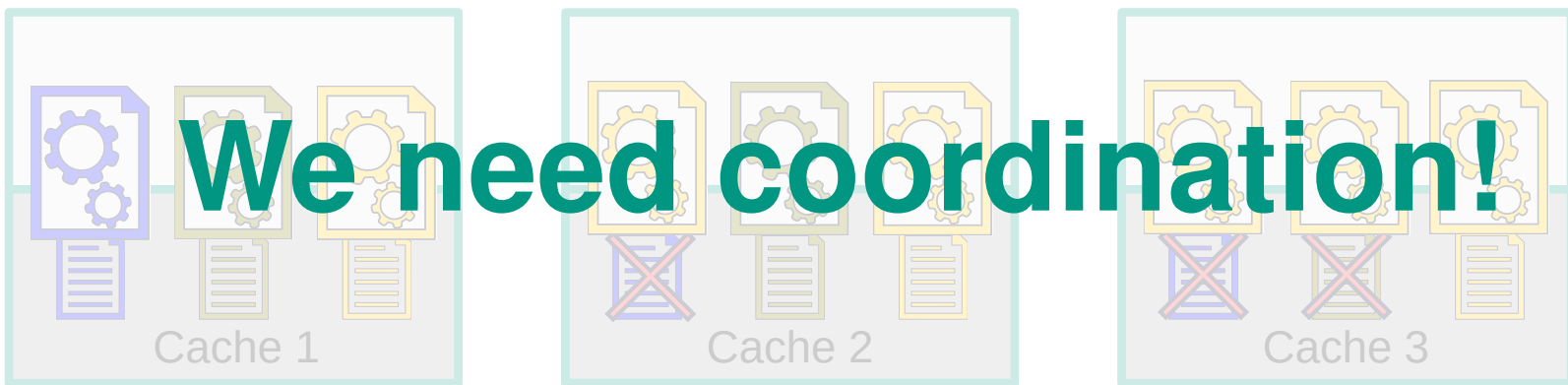
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches
- Continuing until all caches are filled → Replications of same data on different resources
- Coordinate jobs and configure caches ...
  - ... to match current job requirements?



Data Flow

How do we achieve the most efficient data processing with a given resource composition?

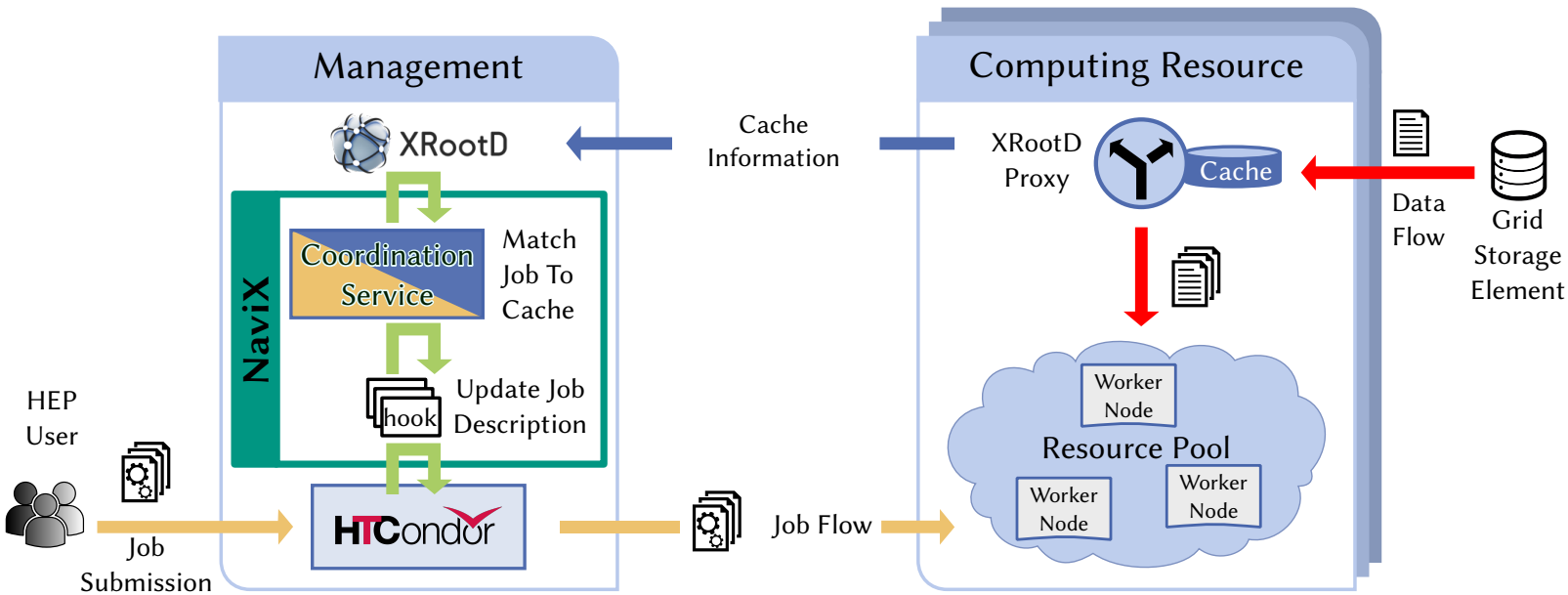
- Jobs starting on a resource and load data into caches
- Repeatedly accessed data is loaded also into other caches
- Continuing until all caches are filled → Replications of same data on different resources
- Coordinate jobs and configure caches . . .
  - . . . to match current job requirements?



We need coordination!

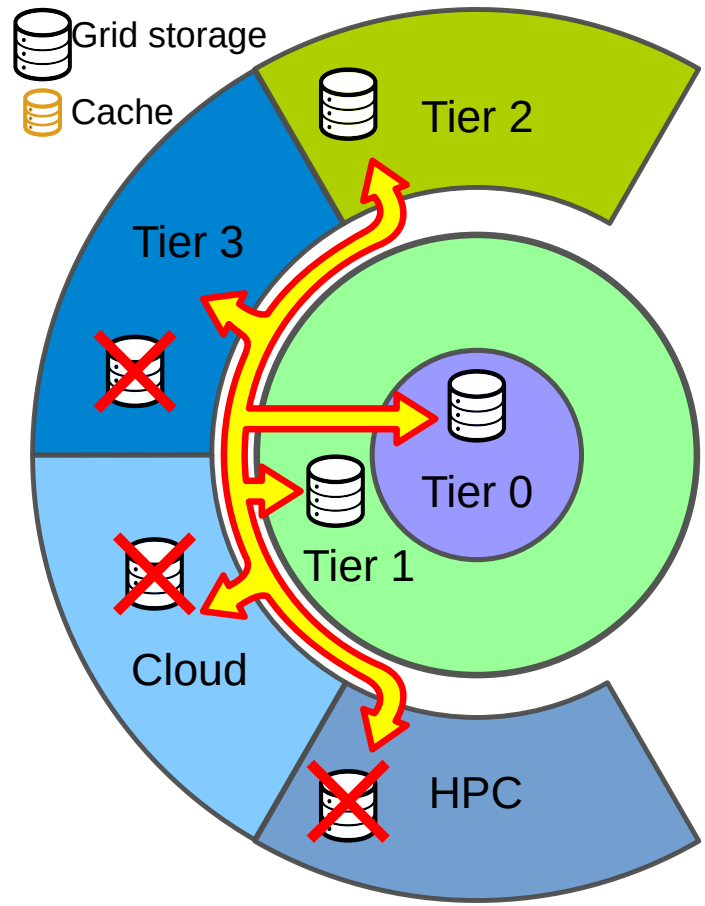
Coordinated Distributed Caching

- First prototype for Coord. Caching: NaviX 10.1051/epjconf/201921404007



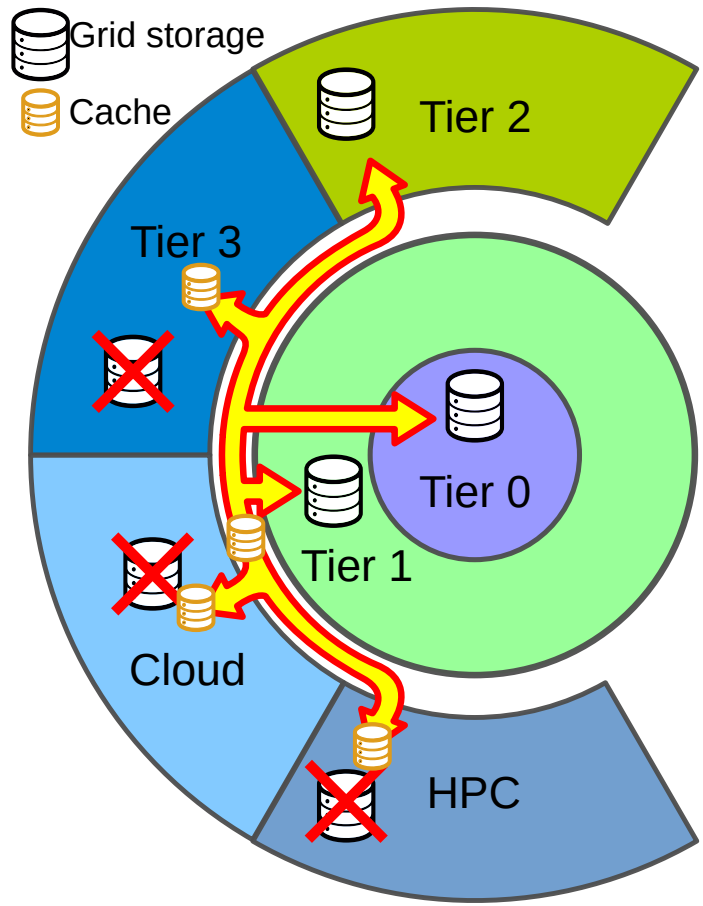
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?



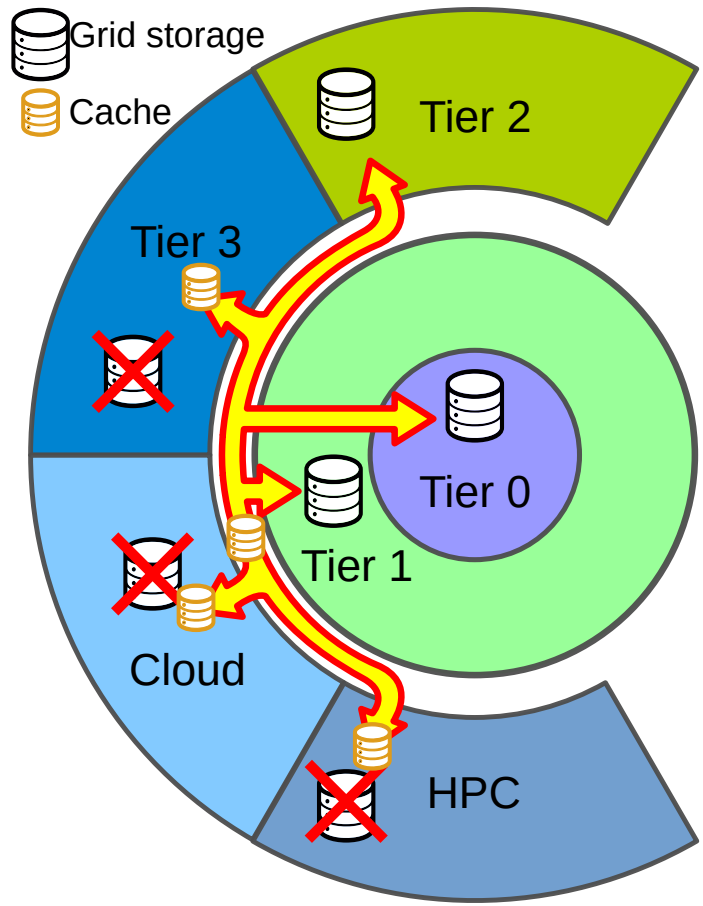
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?



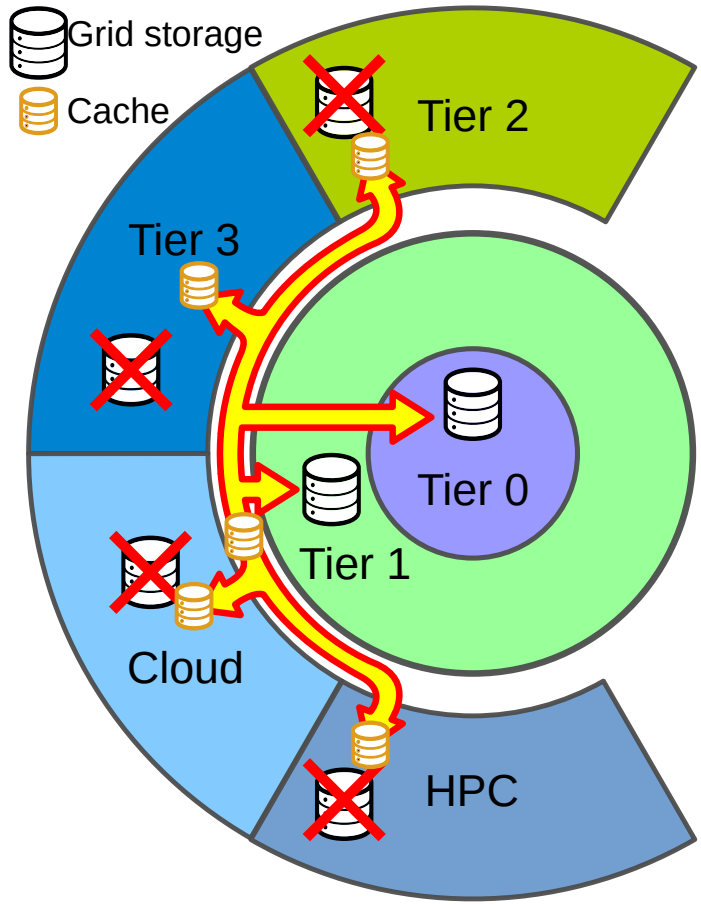
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?



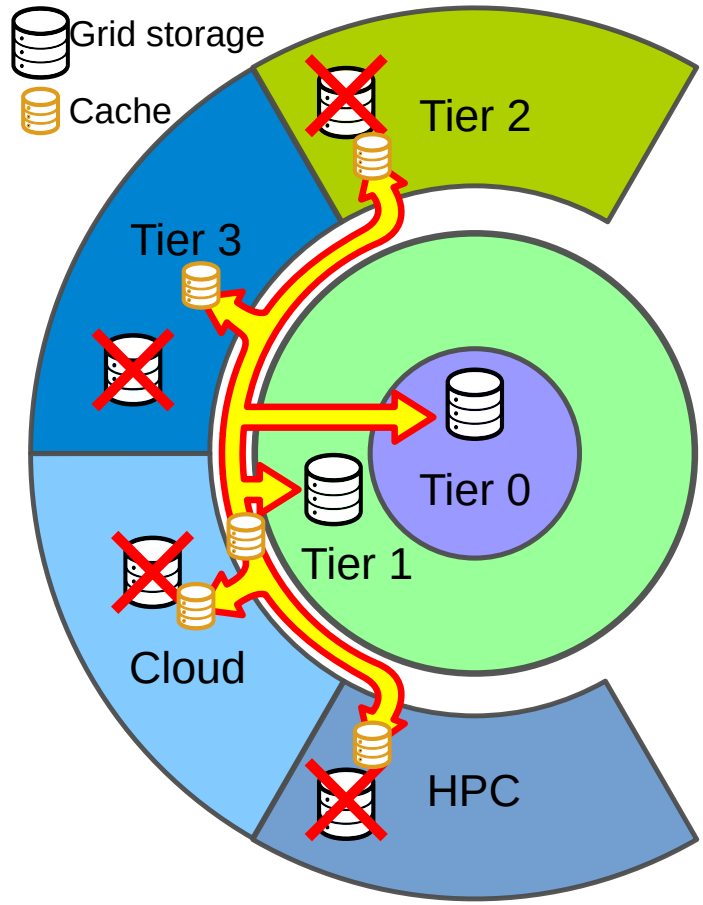
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?
  - Can we replace managed storage with caches without losing momentum?



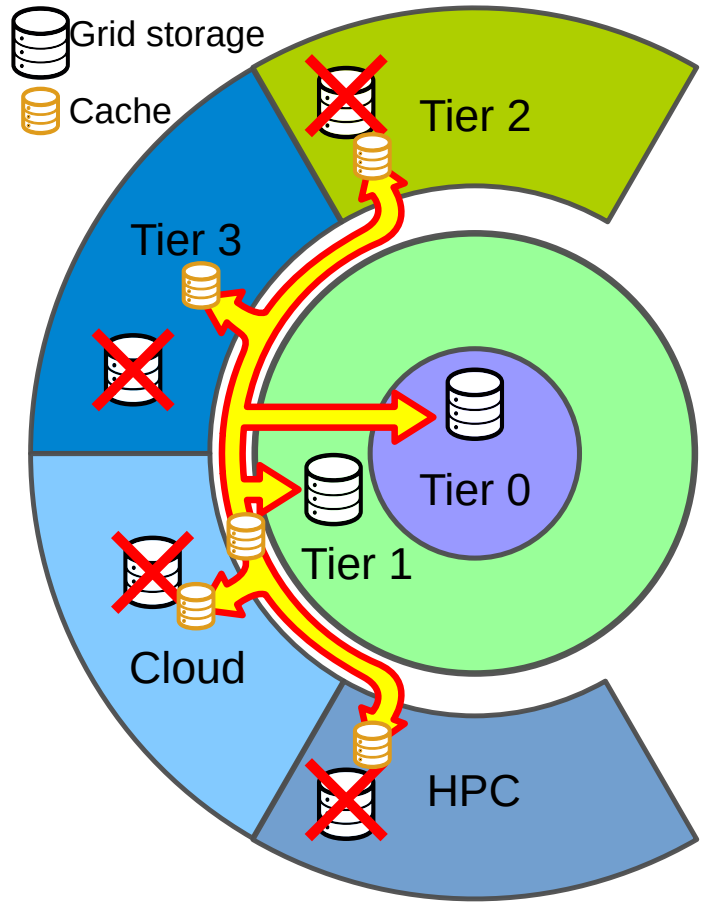
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?
  - Can we replace managed storage with caches without losing momentum?
  - Which figure of merit do we want to optimize? User walltime? Monetary costs? WAN bandwidth?



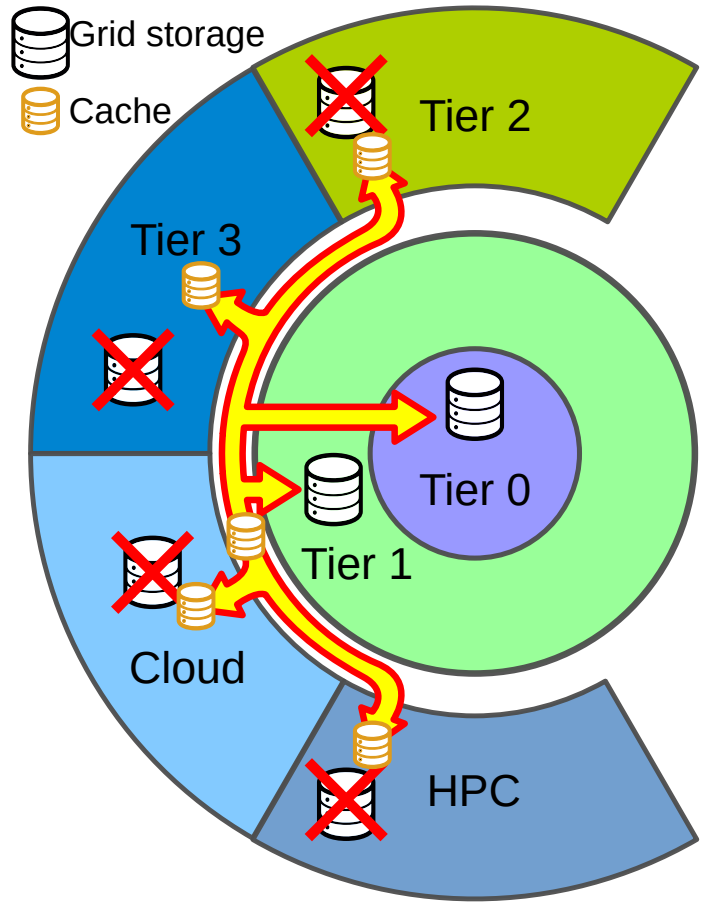
Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?
  - Can we replace managed storage with caches without losing momentum?
  - Which figure of merit do we want to optimize? User walltime? Monetary costs? WAN bandwidth?
  - Do we cache for all workflow types? If not, for which?



Urging Questions Considering Caching

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?
  - Can we replace managed storage with caches without losing momentum?
  - Which figure of merit do we want to optimize? User walltime? Monetary costs? WAN bandwidth?
  - Do we cache for all workflow types? If not, for which?
  - . . . and many many more



Problem-solving Approach for Caching

Karlsruhe Institute of Technology

Fixing workload and scheduler: How do we achieve the most efficient data processing?

- More specific:
  - How do we design the network?
  - How many caches do we need to place? Where?
  - Can we replace managed storage with caches without loosing momentum?
  - ...
- Short: What is the desired computing architecture for the collective German sites?

Simulation can answer these questions!

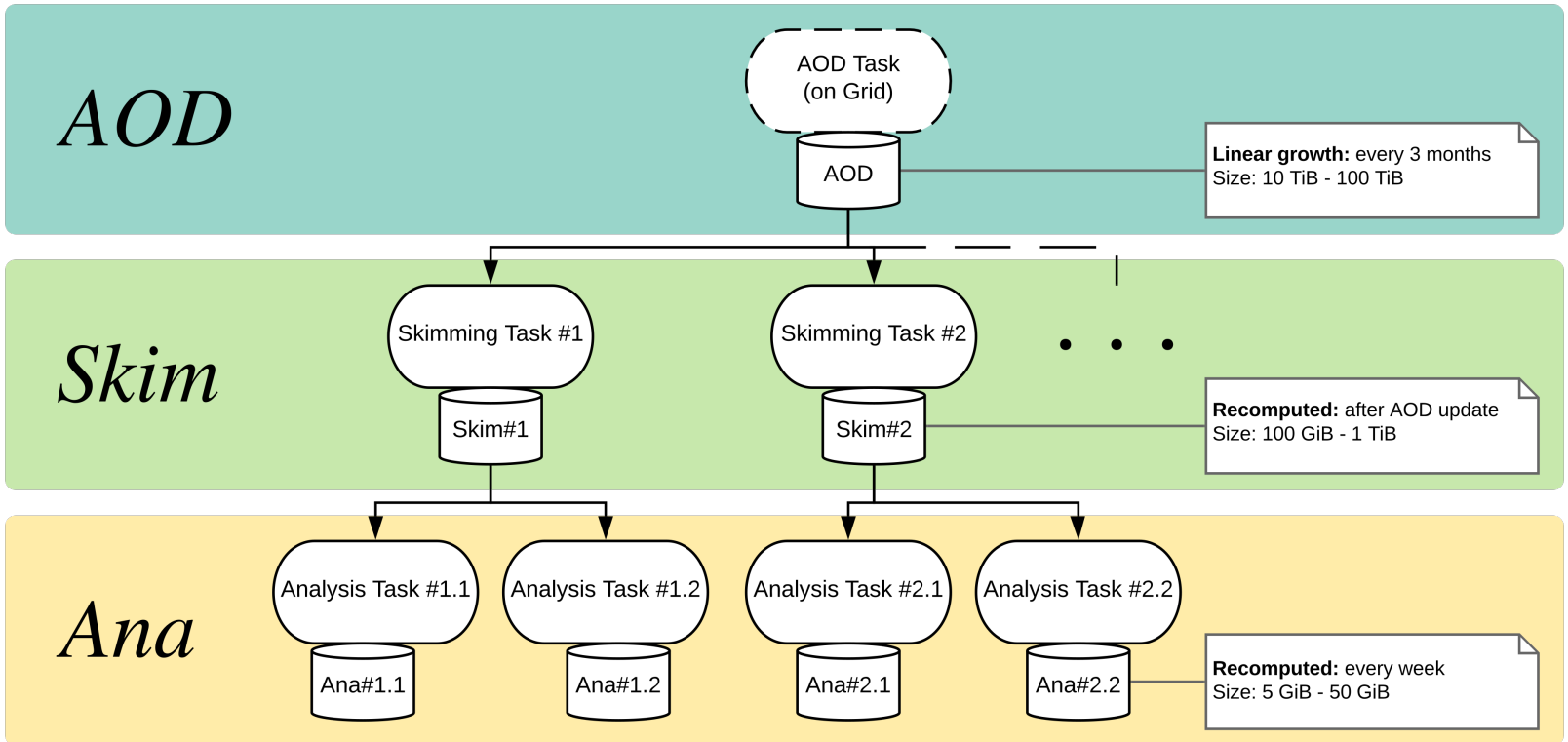
Optimizing Coord. Distributed Caching

- Three construction sites: *Job/Workflow Management*, *Computing-Architecture* and *Scheduler* very challenging due to
  - Complexity of each
  - Back couplings / convolutions / interplay

Caching adds layers of complexity to complex optimization problem!

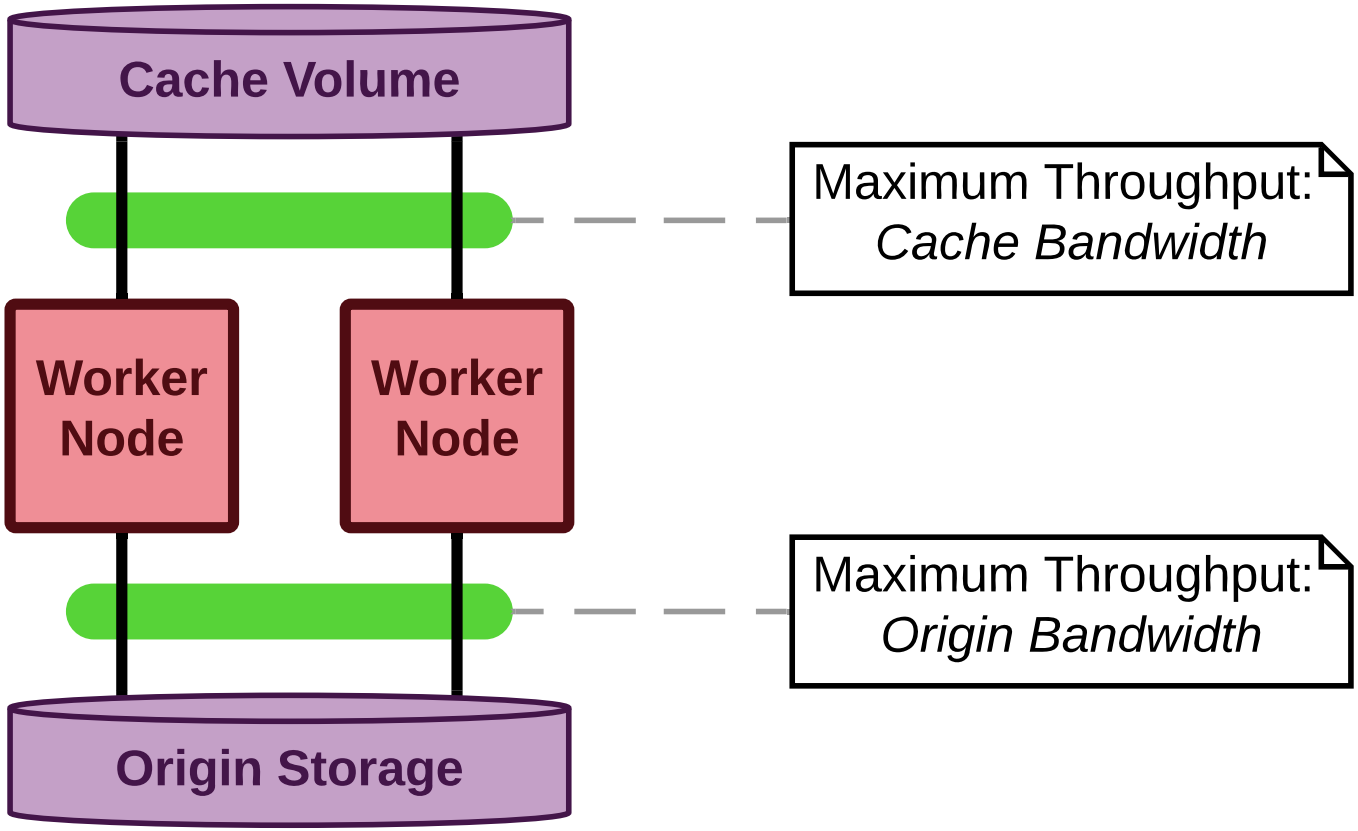
1. Decouple and isolate single components and study separately to answer specific questions
2. Study the whole combined picture

Cache Logic - Workload Generation



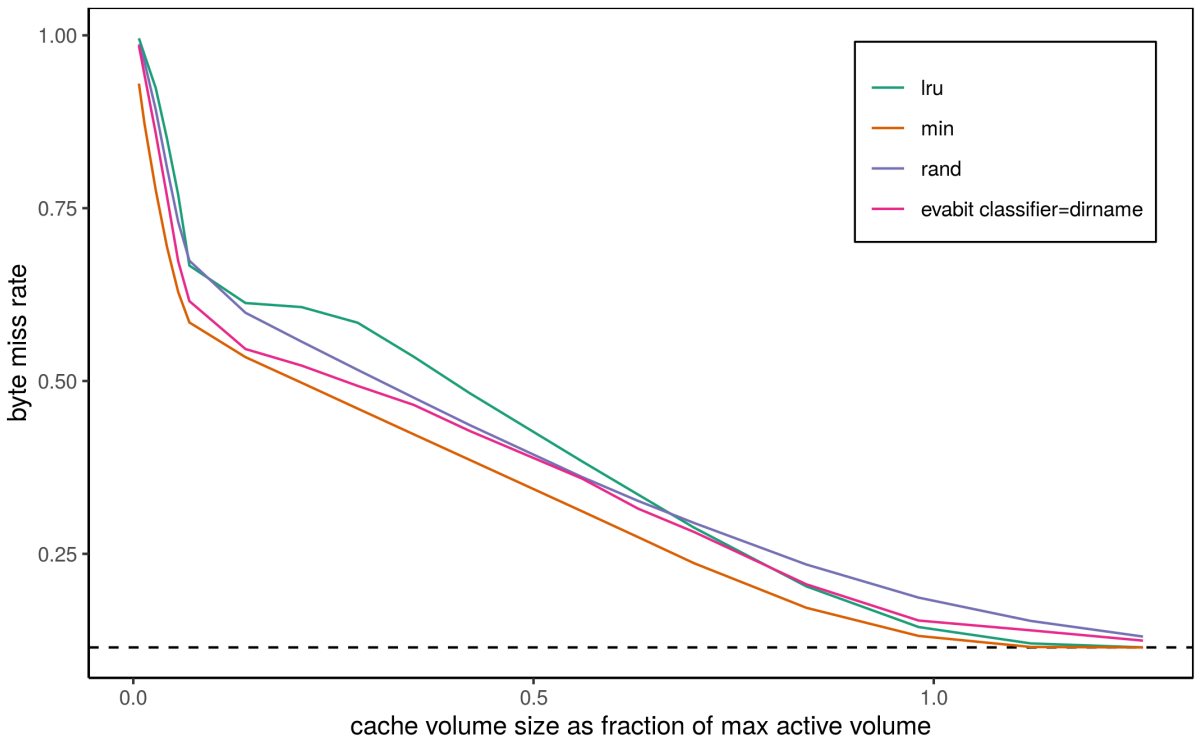
P. Skopnik, *Reactive Caching to Accelerate High-Throughput Computing Workloads*

Cache Logic - Access Generation



P. Skopnik, *Reactive Caching to Accelerate High-Throughput Computing Workloads*

Cache Logic - Impact of Cache Algorithms

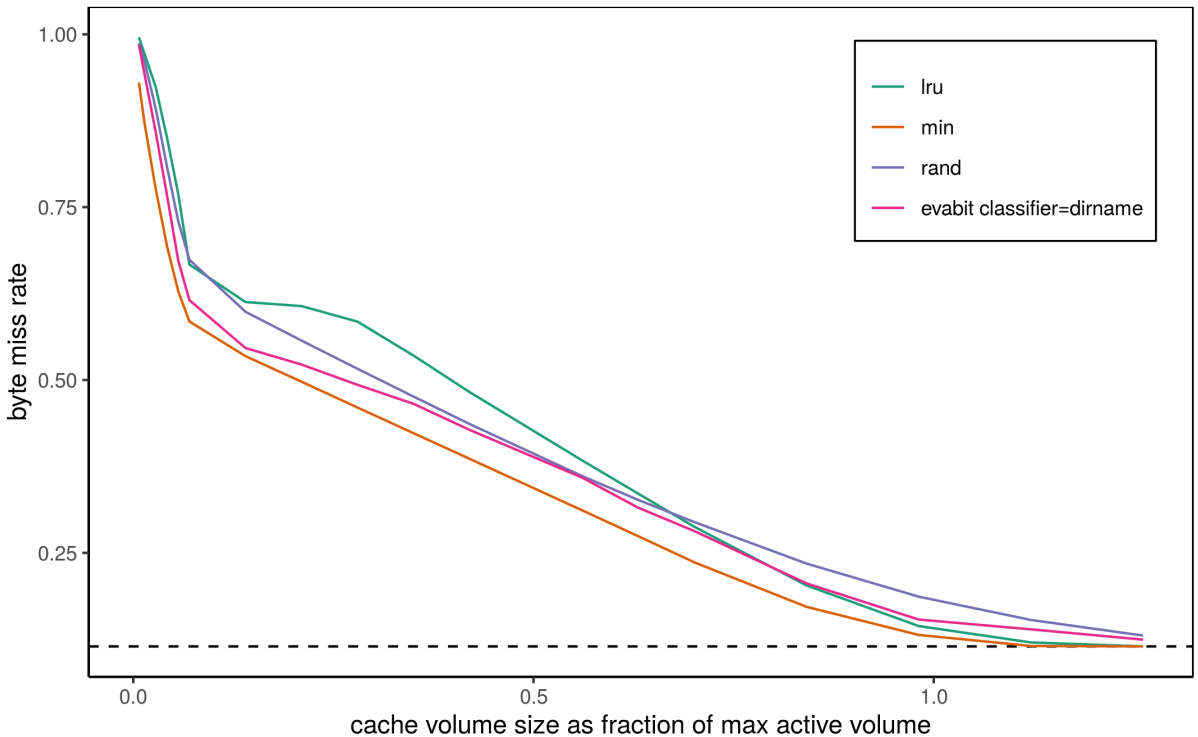


P. Skopnik

- Least Recently Used: standard in XRootD proxy file caching
- MIN: Minimum based on full knowledge
- RAND: random
- EVABIT: probabilistic algorithm

$$\text{missrate} = 1 - \text{hitrate} = 1 - \frac{\text{input-files}_{\text{cache}}}{\text{input-files}_{\text{remote}} + \text{input-files}_{\text{cache}}}$$

Drawing Conclusions from Simulations

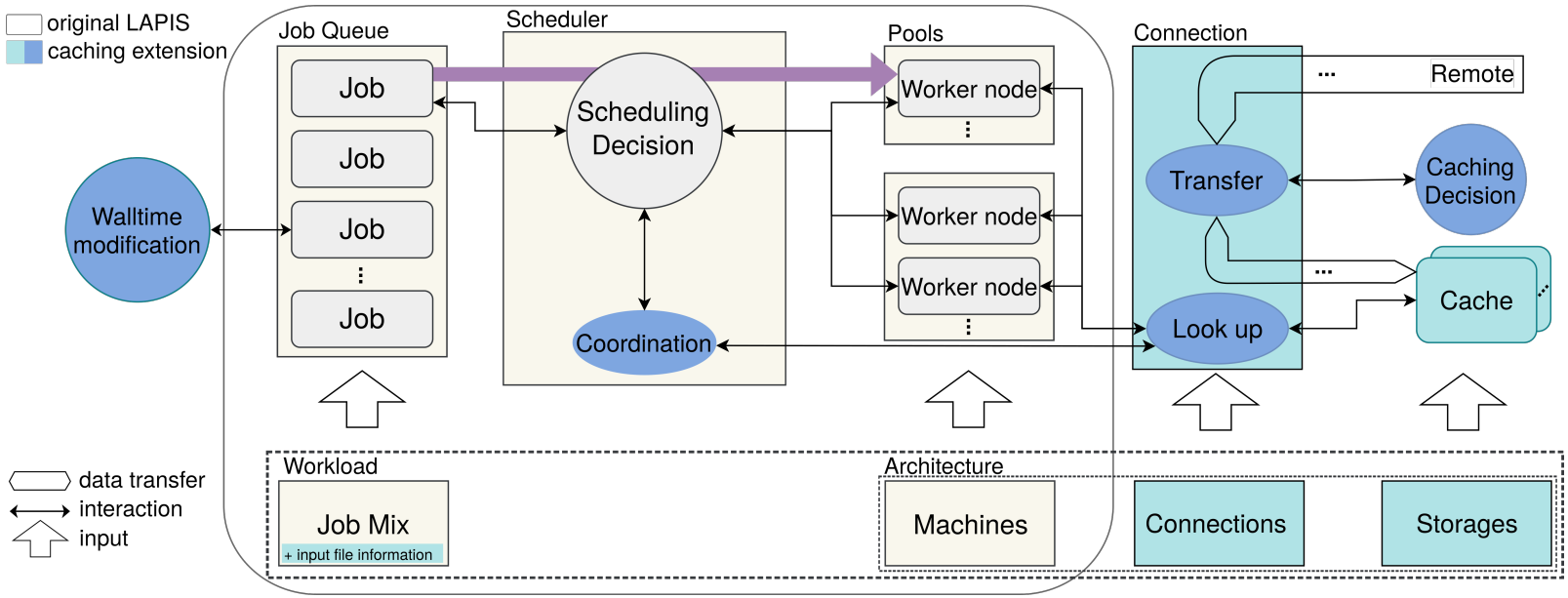


P. Skopnik

- Choice of cache algorithm for general improvement negligible (at least for this type of workflow)

Simulation answers fundamental questions in the caching context with full control over the boundary conditions on a large phase space!

Simulator LAPIS

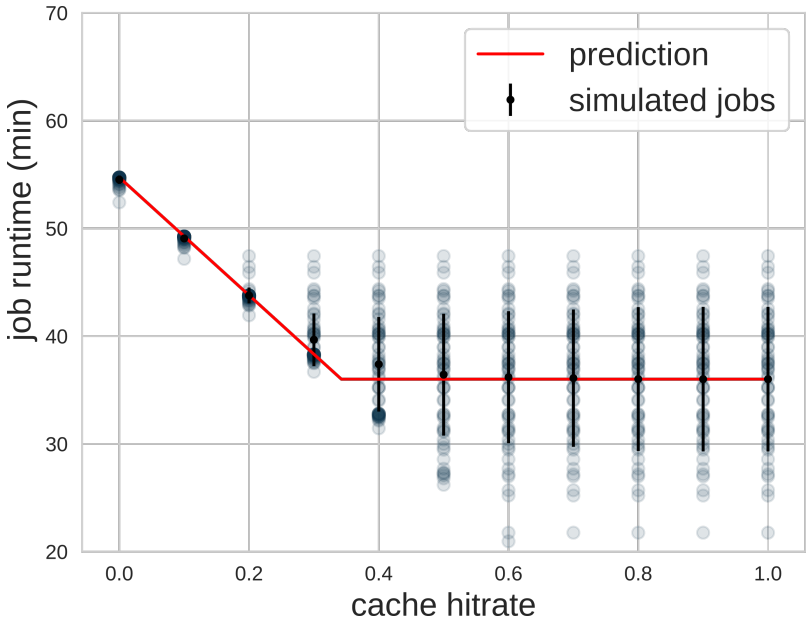


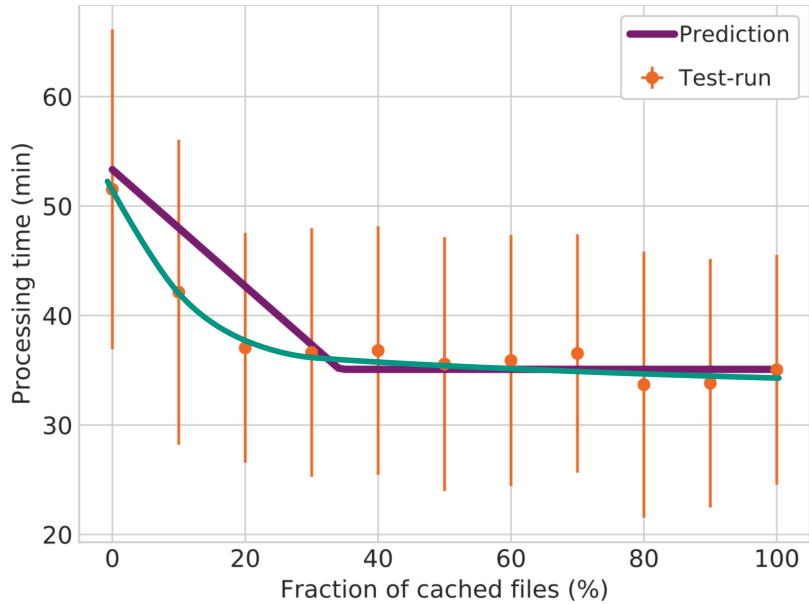
T. Feßenbecker, *Modeling of distributed coordinated caching for LHC data analyses*

- Central metric for job characterization: job walltime

$$t_{\text{wall}} = \max\left(t_{\text{calculation}}, t_{\text{transfer}}\right) = \max\left(\frac{t_{\text{CPU}}}{\varepsilon_{\text{instr}}}, \frac{V \cdot (1-h)}{b_{\text{remote}}}, \frac{V \cdot h}{b_{\text{cache}}}\right)$$

Simulating Coord. Caching





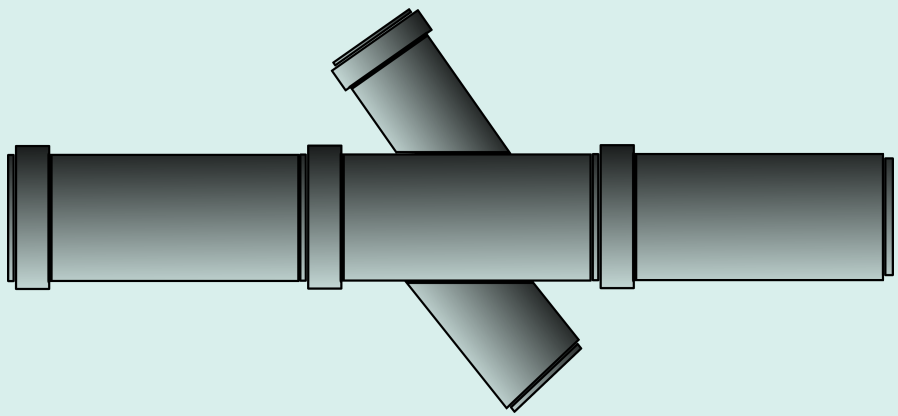
T. Feßenbecker, *Modeling of distributed coordinated caching for LHC data analyses*

- Good starting point for deeper studies!
- Increase freedom to be able to answer more complex questions

Generalizing the Simulation Ansatz

- One possible improvement: model . . .
  - . . . that separates between architectures and workloads and avoids a convoluted basis
  - . . . is able to define and handle arbitrarily complex architectures and workloads
  - . . . is able to simulate at arbitrary precision

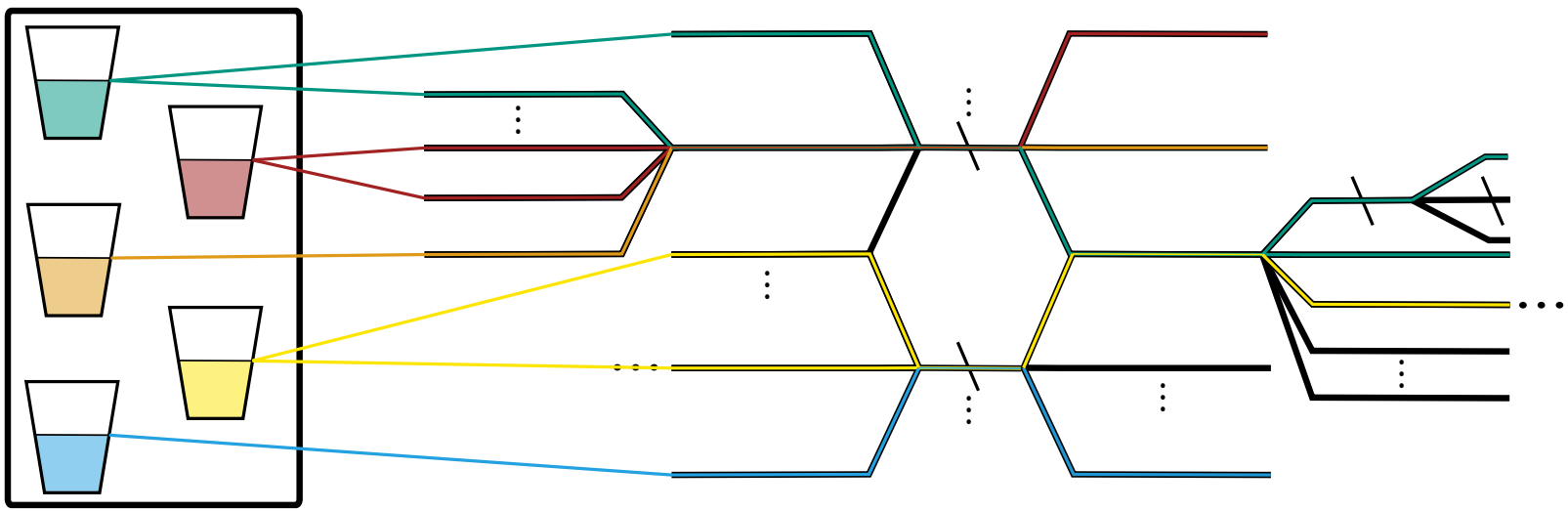
Information Flux Model



- Trade-off between level of generalization and expense due to increasing complexity!

Full Simulation

- Macroscopically three components: *Workload*, *Architecture* and *Scheduler* already covered by LAPIS



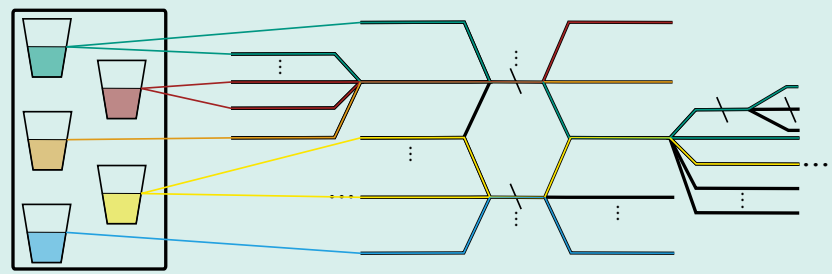
- Use simulation to answer urging questions

Summary

- Efficiency improvement through (redundant) caching is self-evident, but complex

- Simulation provides a
  - scalable
  - flexibly modifiable
  - adaptable
  - universally usable

  ansatz to solve arbitrarily complex optimization problems with many possible configurations!



- LAPIS provides the foundation for getting answers

Backup!

Economic Value Added (EVA)

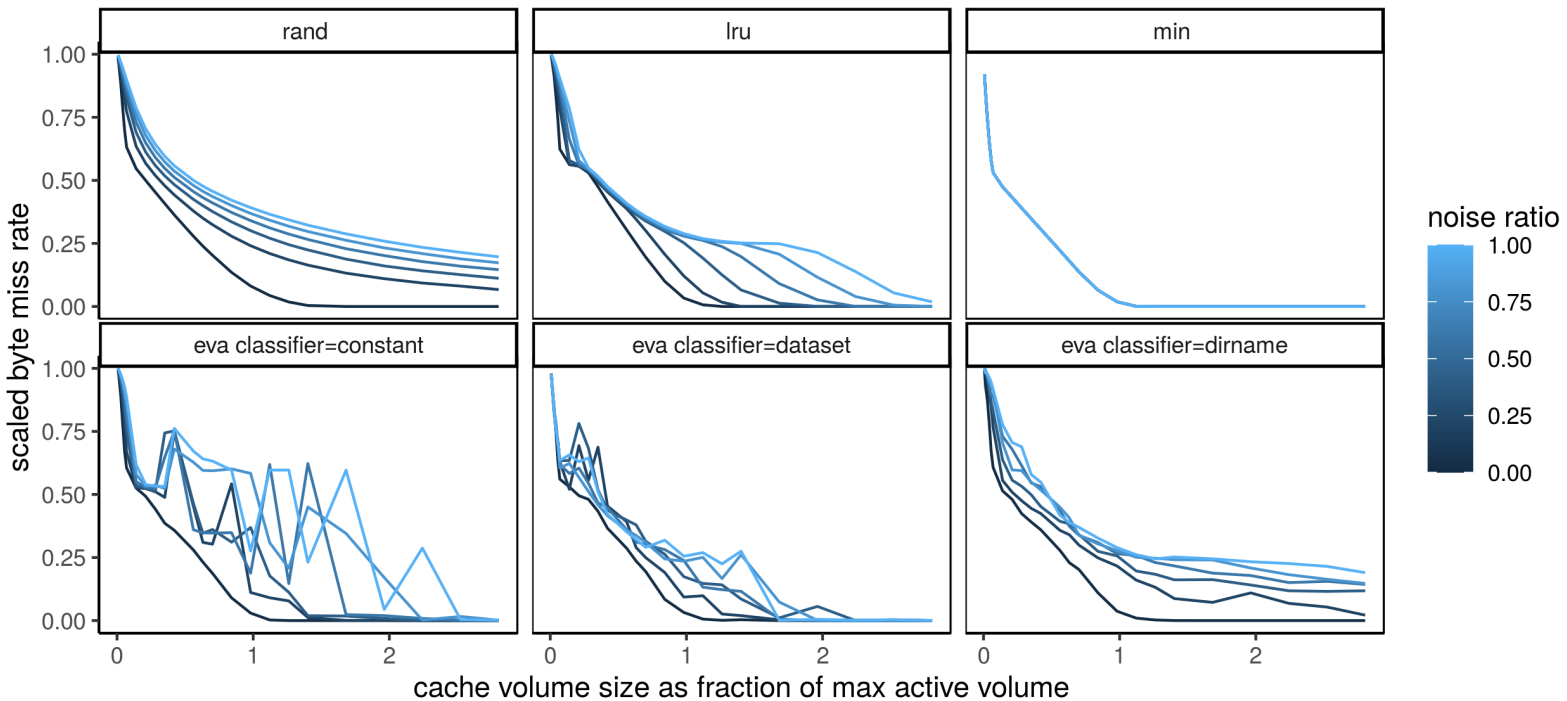
- Assign a value to each element f

$$EVA(f) = \text{expected hits of } f - \frac{\text{cache hit rate}}{\text{cache size}} \cdot \text{expected remaining lifetime of } f$$

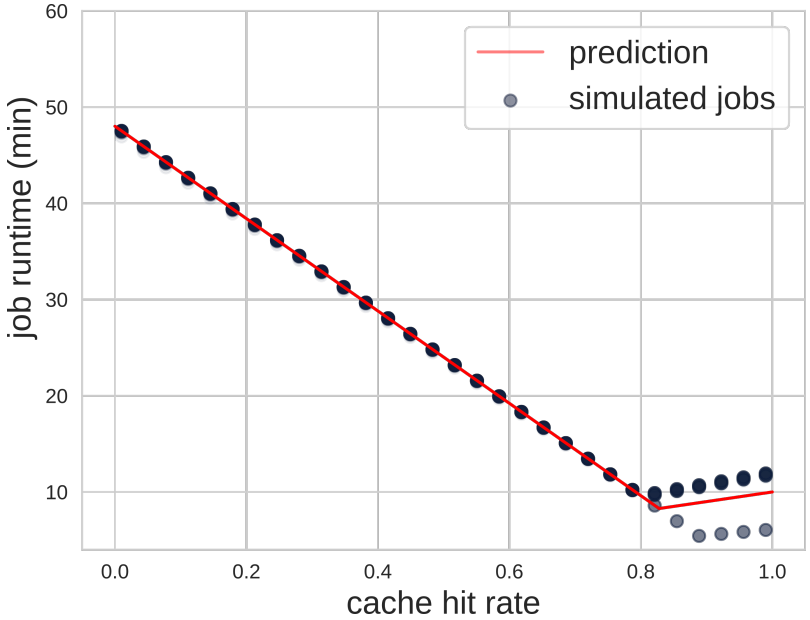
- Evict element with smallest value
- All elements are sorted into predefined classes
  - expected hits of f and expected remaining lifetime of f randomly drawn from class distribution

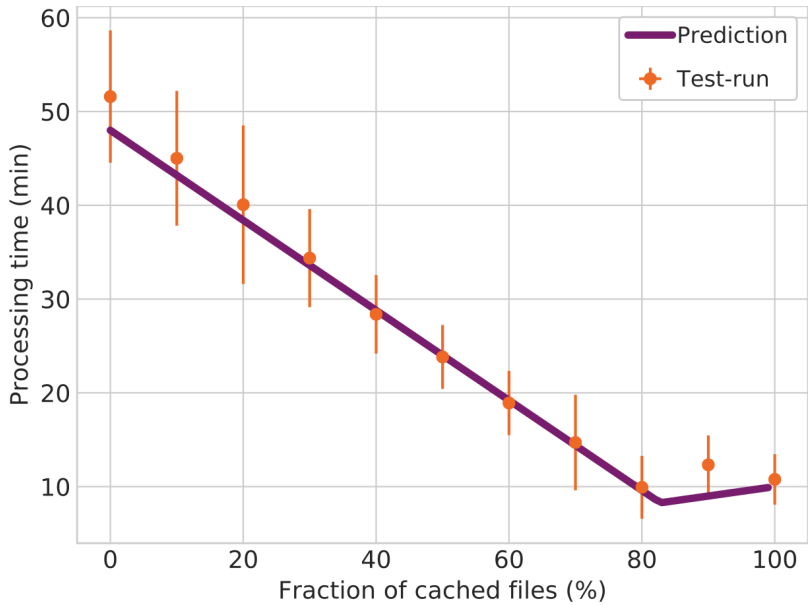
Cache logic - Noise stability

- Adding single accesses to files belonging to an external dataset at a given rate
- Since single accesses, no benefit from caching



Simulating Simplified Coord. Caching

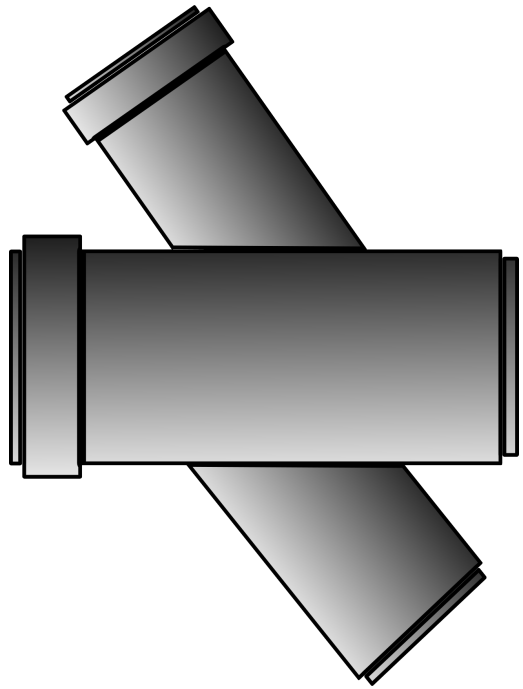




$$\text{hitrate} = \frac{\text{input-files}_{\text{cache}}}{\text{input-files}_{\text{remote}} + \text{input-files}_{\text{cache}}}$$

Information Flux Model – Basics I

- Two (three) assumptions:
  - Causality
  - Local information conservation
  - (Continuity of information)
- Idea:
  - A job is a volume of information, which needs to be transferred
  - The transferring agent are pipes with limited throughput, which carry the information from one point to another
  - The interplay of job collections (workflows) with arbitrary pipe structures (computing architectures) is able to picture real systems



Information Flux Model – Basics II

Definition: Information

- On a very elementary level: digital signals on arbitrary digital-electronic elements
- But: Resolution property of the model can be used to absorb this into an effective abstraction by making the right assumptions

- Current is limited by maximum throughput of the pipe

$$\mathcal{T}_{\text{pipe}} = w_{\text{pipe}} \cdot \nu_{\text{pipe}} \geq I_{\text{pipe}}$$

- Information in general not a conserved quantity $\rightarrow$ Entry point for gain and drain currents needed

Information Flux Model – Gain and Drain I

- Information in general not a conserved quantity
  $\rightarrow$ Entry point for gain and drain currents I^{gain} and I^{drain} needed

- Local information conservation:

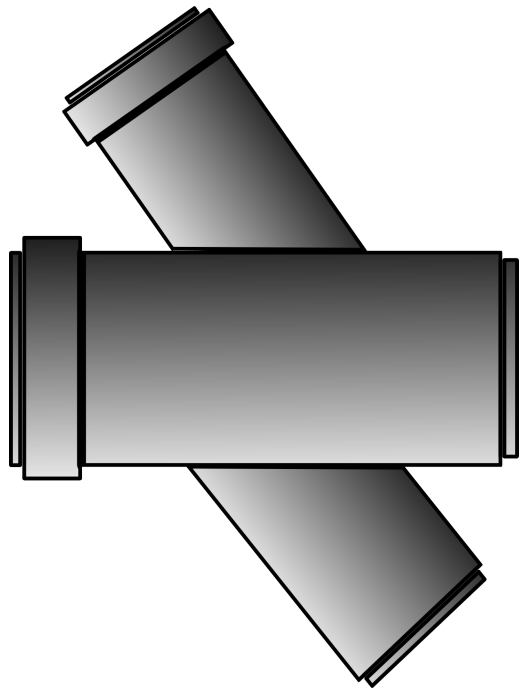
$$I^{\text{out}}_{\text{pipe}} + I^{\text{drain}}_{\text{pipe}} = I^{\text{in}}_{\text{pipe}} + I^{\text{gain}}_{\text{pipe}} \leq \mathcal{T}_{\text{pipe}}$$

- Gain is dependent on incoming current (causally connected)

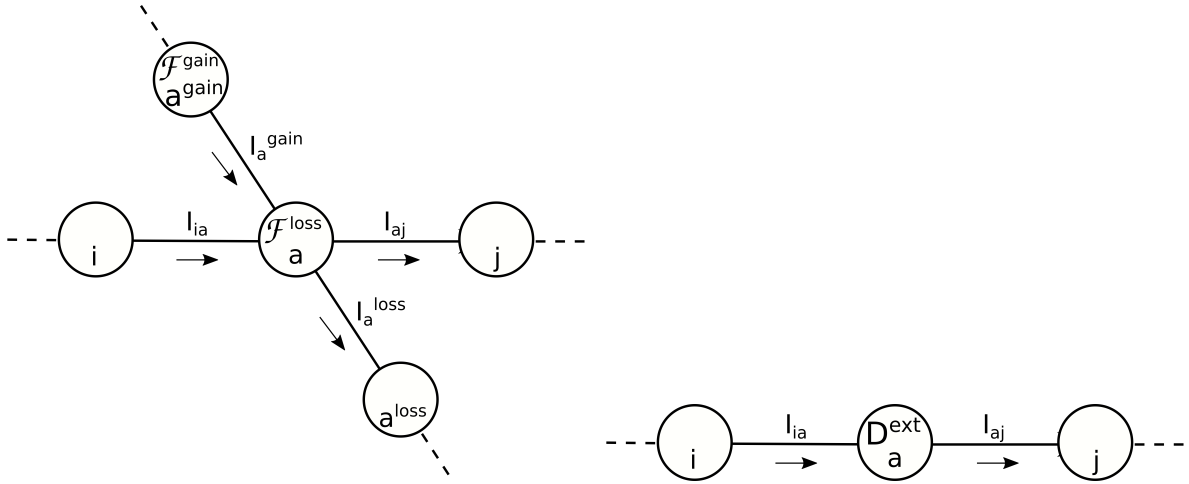
$$V^{\text{job}}_{\text{gain}} = \mathcal{F}_{\text{gain}}\left(V^{\text{job}}_{\text{in}}\right)$$

- Drain is dependent on incoming and gain currents

$$V^{\text{job}}_{\text{drain}} = \mathcal{F}_{\text{drain}}\left(V^{\text{job}}_{\text{in}}, V^{\text{job}}_{\text{gain}}\right)$$



Information Flux Model – Gain and Drain II



$$I_i^{out} = \left(1 + \frac{\partial \mathcal{F}_i^{gain}}{\partial V_i^{in}} - \frac{\partial \mathcal{F}_i^{drain}}{\partial V_i^{in}} - \frac{\partial \mathcal{F}_i^{drain}}{\partial V_i^{gain}} \frac{\partial \mathcal{F}_i^{gain}}{\partial V_i^{in}}\right) I_i^{in}$$

Information Flux Model – Pipe Chain Limit

- In a pipe chain the current can be limited by all subsequent pipes

→ For a linear pipe chain with n pipes the incoming current at pipe i:

$$I_i^{\text{in}} \leq \min\left(\left\{ \frac{\mathcal{T}_j}{D_j^{\text{int}}} \cdot \prod_{k=i}^{j-1} \frac{1}{D_k^{\text{ext}}} \;\middle|\; \text{for } i \leq j \leq n \right\}\right)$$

$$\text{with } D_i^{\text{int}} = 1 + \frac{\partial \mathcal{F}_i^{\text{gain}}}{\partial V_i^{\text{in}}} \text{ and } D_i^{\text{ext}} = 1 + \frac{\partial \mathcal{F}_i^{\text{gain}}}{\partial V_i^{\text{in}}} - \frac{\partial \mathcal{F}_i^{\text{drain}}}{\partial V_i^{\text{in}}} - \frac{\partial \mathcal{F}_i^{\text{drain}}}{\partial V_i^{\text{gain}}} \frac{\partial \mathcal{F}_i^{\text{gain}}}{\partial V_i^{\text{in}}}$$

Unreliable Connection

- Input information volume V^{in} is transferred by a single pipe
- Random loss of information with unique rate c for the specific component:

$$\mathcal{F}^{\text{gain}} = 0$$

$$d\left\langle \mathcal{F}^{\text{drain}} \right\rangle = d\left\langle \mathcal{F}^{\text{loss}} \right\rangle = c \cdot dV^{\text{in}}$$

- Output information stream:

$$I^{\text{out}} = (1 - c)\, I^{\text{in}}$$

One-pipe Duplication Job

- Input information:

$$V^{\text{in}} = \mathcal{V}^{\text{instr}} + \mathcal{V}^{\text{data}}$$

$$f^{\text{instr}} = \frac{\mathcal{V}^{\text{instr}}}{V^{\text{in}}}$$

- Duplication of data with n number of duplicates:

$$d\left\langle \mathcal{F}^{\text{gain}} \right\rangle = n \cdot \left(1 - f^{\text{instr}}\right) dV^{\text{in}}$$

$$\mathcal{F}^{\text{drain}} = 0$$

- Output information stream:

$$I^{\text{out}} = \left(1 + n\left(1 - f^{\text{instr}}\right)\right) I^{\text{in}} \approx (1 + n)\, I^{\text{in}}$$

One-pipe Simulation Job

- Input information:

$$V^{\text{in}} = \mathcal{V}^{\text{instr}} + \mathcal{V}^{\text{data}}$$

$$f^{\text{instr}} = \frac{\mathcal{V}^{\text{instr}}}{V^{\text{in}}}$$

- Creation of data from instructions with boost factor b and filtering of created data with efficiency ϵ:

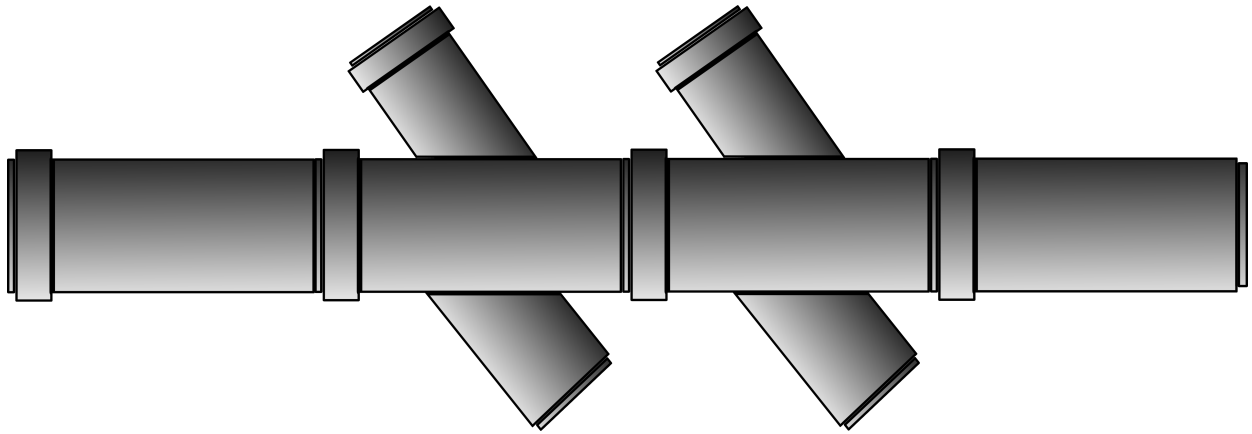
$$d\left\langle \mathcal{F}^{\text{gain}} \right\rangle = b \cdot d\mathcal{V}^{\text{instr}} = bf^{\text{instr}} dV^{\text{in}}$$

$$d\mathcal{F}^{\text{drain}} = \epsilon \cdot dV^{\text{gain}} + d\mathcal{V}^{\text{instr}} = \epsilon \cdot dV^{\text{gain}} + f^{\text{instr}} dV^{\text{in}}$$

- Output information stream:

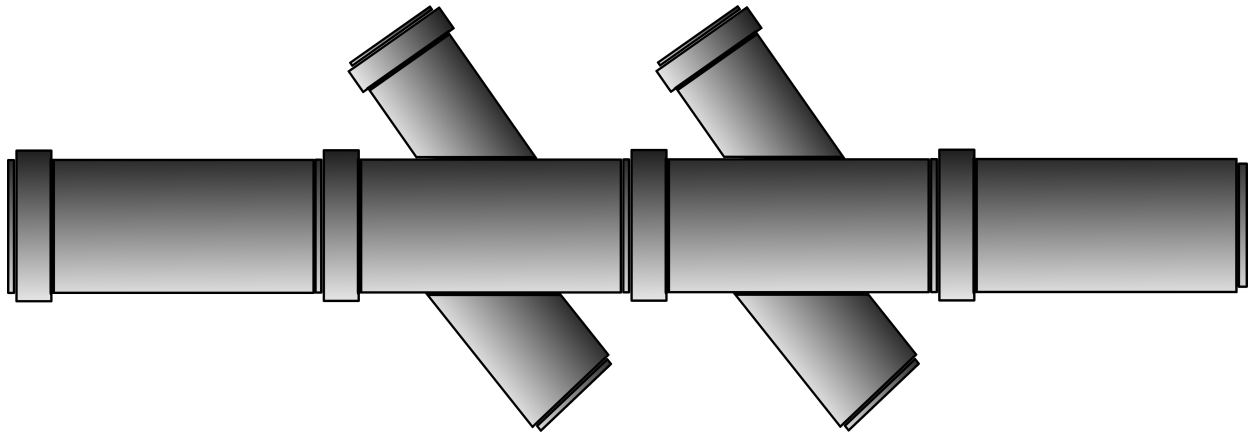
$$I^{\text{out}} = \left(1 + bf^{\text{instr}} - f^{\text{instr}} - \epsilon bf^{\text{instr}}\right) I^{\text{in}} \approx (1 - \epsilon)\, bI^{\text{in}}$$

Four-pipes Simulation Job



- Pipes labelled from left to right with counter
- Simplifying here: $f_{\text{instr}} = 1$

Four-pipes Simulation Job – Information Fluxes



- $I_1^{\text{in}} = I_1^{\text{out}}$ and $I_2^{\text{in}} = I_1^{\text{out}}$
- $d\mathcal{F}_2^{\text{drain}} = dV_2^{\text{in}}$, $d\mathcal{F}_2^{\text{gain}} = b dV_2^{\text{in}} \quad \Rightarrow \quad I_2^{\text{out}} = b I_2^{\text{in}} = b I_1^{\text{in}}$
- $I_3^{\text{in}} = I_2^{\text{out}}$
- $d\mathcal{F}_3^{\text{gain}} = 0$, $d\mathcal{F}_3^{\text{drain}} = \epsilon dV_3^{\text{in}} \quad \Rightarrow \quad I_3^{\text{out}} = (1-\epsilon)\, I_3^{\text{in}} = (1-\epsilon)\, b I_1^{\text{in}}$
- $I_4^{\text{out}} = I_4^{\text{in}} = I_3^{\text{out}} = (1-\epsilon)\, b I_1^{\text{in}}$

Four-pipes Simulation Job – Throughput Limitation

- Throughput limitation for the incoming current:

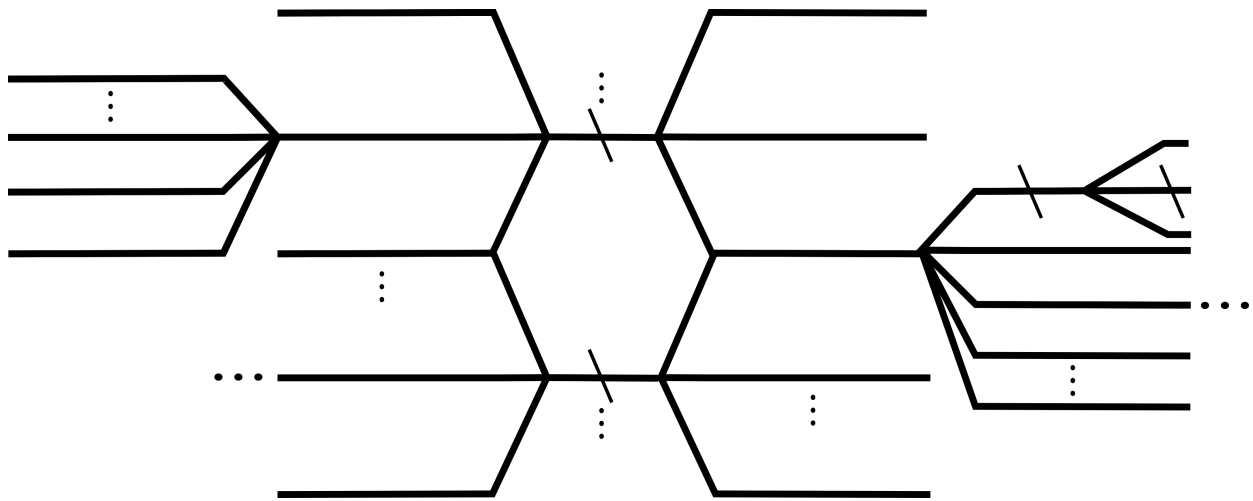
$$
\begin{aligned}
I_1^{\text{in}} &\leq \min\left(\frac{\mathcal{T}_1}{D_1^{\text{int}}}, \frac{\mathcal{T}_2}{D_2^{\text{int}}}\frac{1}{D_1^{\text{ext}}}, \frac{\mathcal{T}_3}{D_3^{\text{int}}}\frac{1}{D_2^{\text{ext}}}\frac{1}{D_1^{\text{ext}}}, \frac{\mathcal{T}_4}{D_4^{\text{int}}}\frac{1}{D_3^{\text{ext}}}\frac{1}{D_2^{\text{ext}}}\frac{1}{D_1^{\text{ext}}}\right) \\
&\leq \min\left(\mathcal{T}_1, \frac{\mathcal{T}_2}{1+b}, \frac{\mathcal{T}_3}{b}, \frac{\mathcal{T}_4}{b(1-\epsilon)}\right)
\end{aligned}
$$

- Different to the one pipe example:

$$
I_{\text{pipe}}^{\text{in}} \leq \frac{\mathcal{T}_{\text{pipe}}}{1+b}
$$

- However, if $\frac{\mathcal{T}_2}{1+b}$ limiting factor and $\mathcal{T}_{\text{pipe}} = \mathcal{T}_2$ both descriptions are equivalent

Non-Linear Architectures

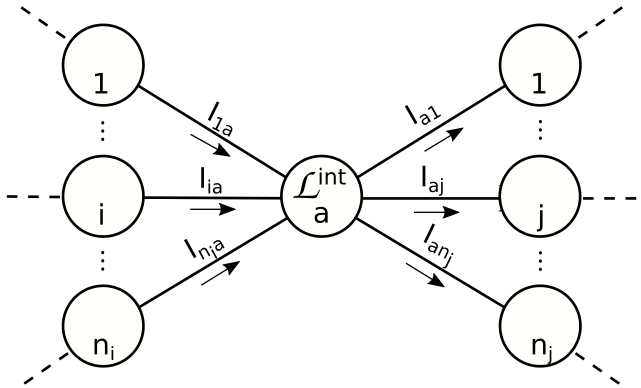


Internal Node-logic

- Many possible paths for a single job to choose

$\rightarrow$ Some internal logic $\mathcal{L}^{\text{int}}$ needed to guide the jobs across certain paths

$\Rightarrow$ Scheduler module matches jobs to paths



- Multiple jobs might share same paths

$\rightarrow$ Internal logic needed to share available throughputs among jobs

$\Rightarrow$ Implementation of transfer protocols in architecture components