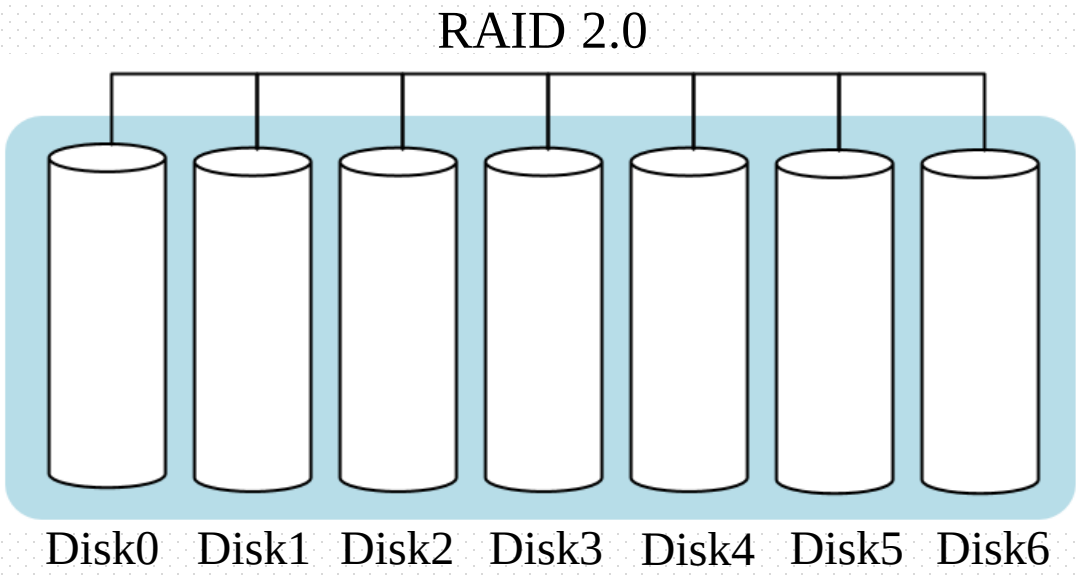
RAID2.0++: Fast Reconstruction for Large Disk Enclosures Based on RAID2.0

Qiliang Li, Yinlong Xu and Min Lyu

University of Science and Technology of China

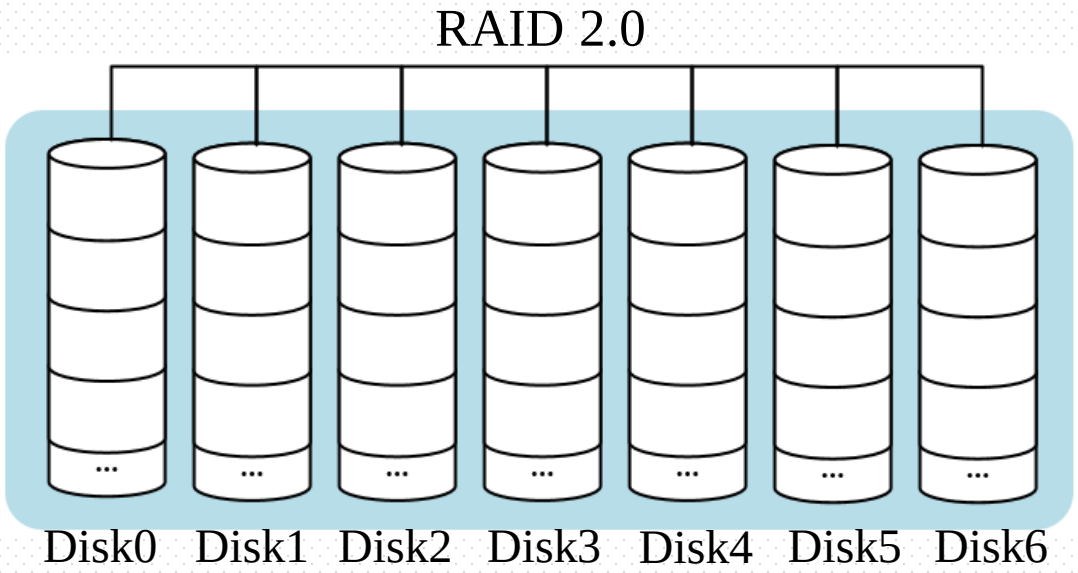
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)



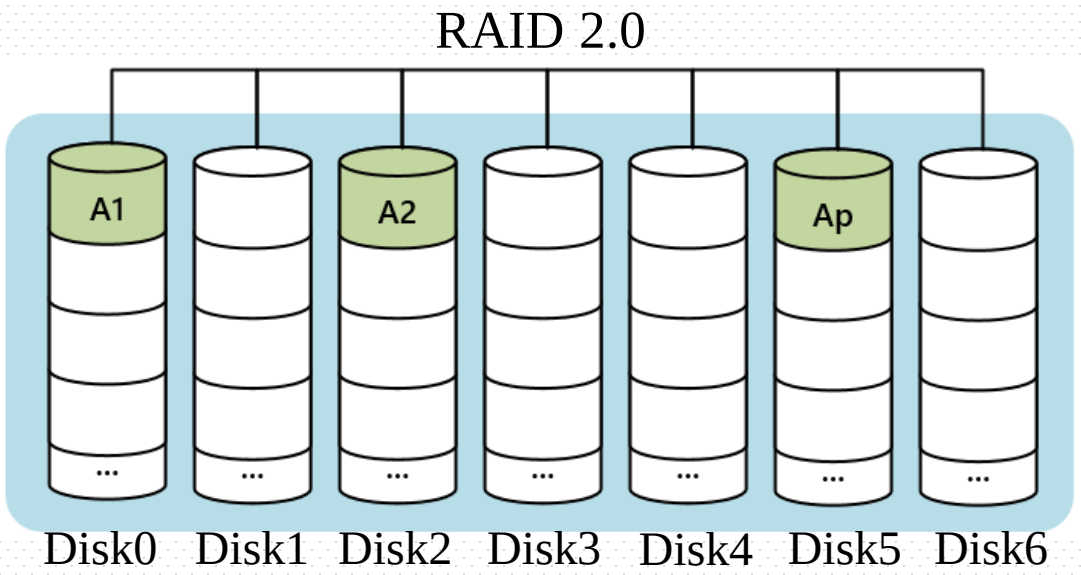
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)



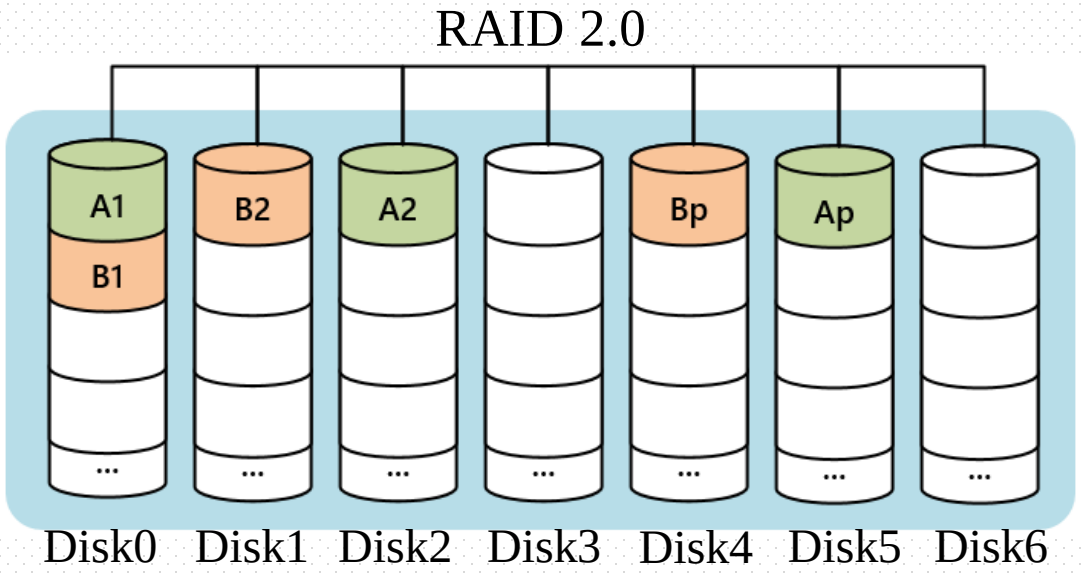
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)
  - Construct RAID by randomly selected chunks
    - RAID5 groups: A1, A2, Ap



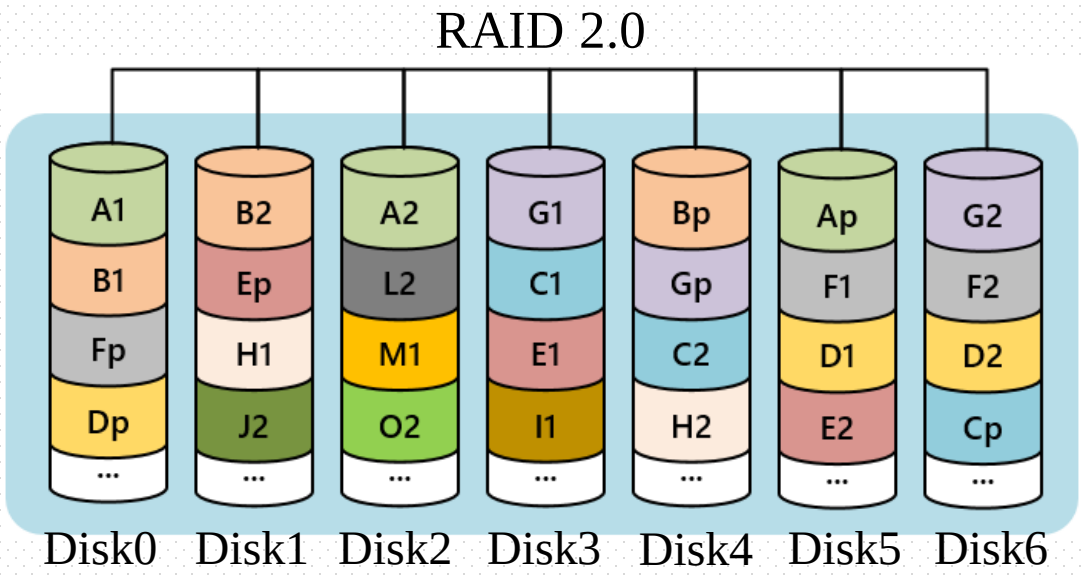
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)
  - Construct RAID by randomly selected chunks
    - RAID5 groups: A1, A2, Ap
      B1, B2, Bp



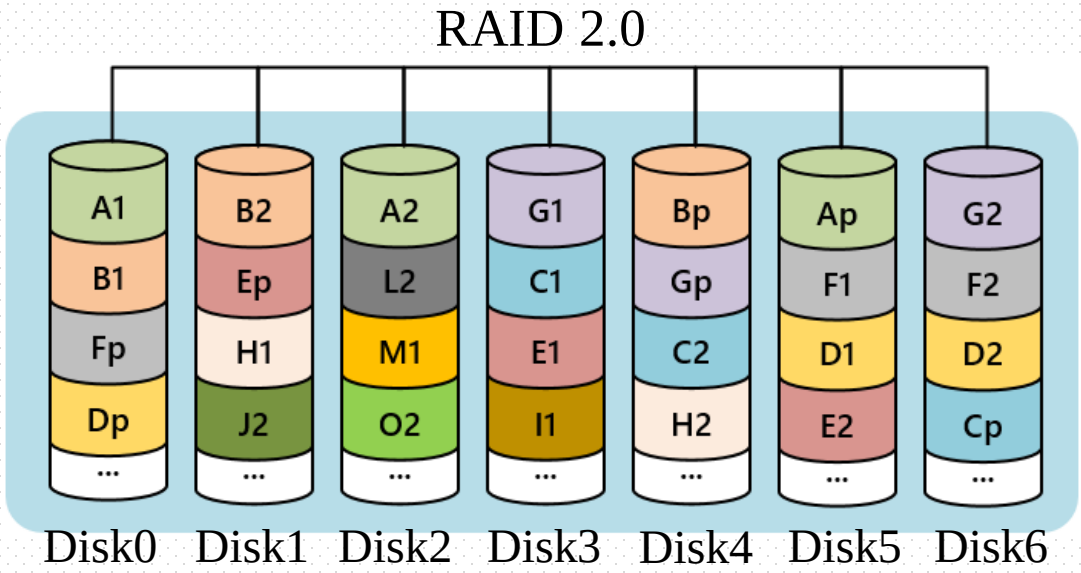
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)
  - Construct RAID by randomly selected chunks
    - RAID5 groups: A1, A2, Ap
      B1, B2, Bp
      …



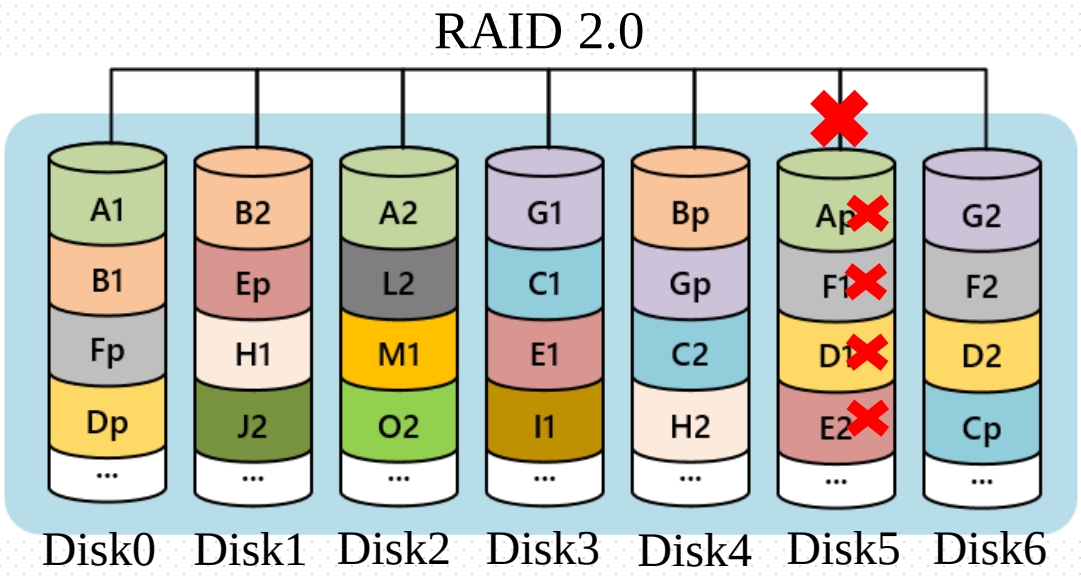
Background

- RAID 2.0
  - Divide each disk into chunks (usually not smaller than 64MB)
  - Construct RAID by randomly selected chunks
- Features
  - Flexible resource scheduling
  - Use hot spare space for reconstruction



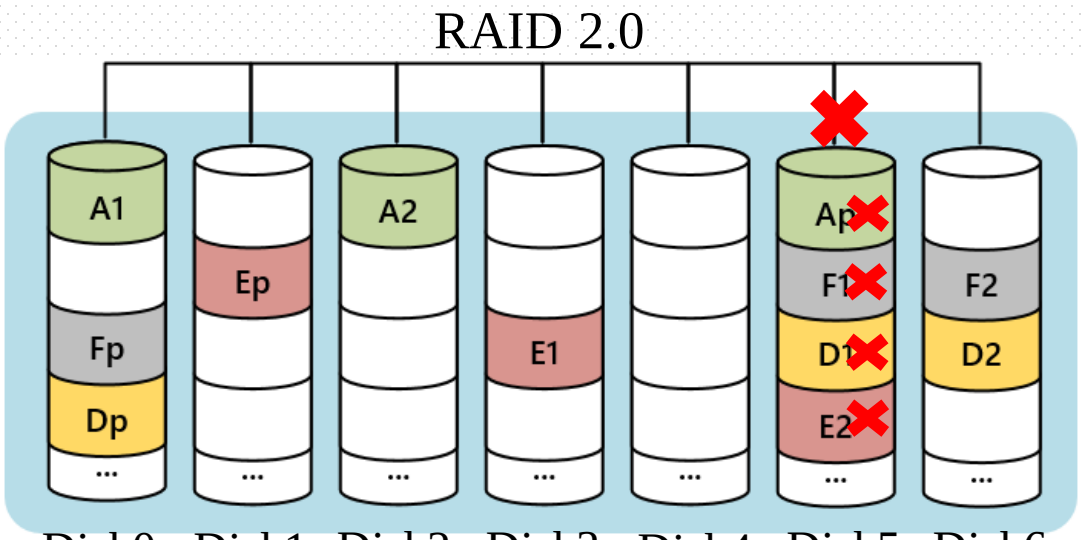
Background

- Reconstruction
  - Perform in batches
  - Reconstruct a certain number of chunks in each batch



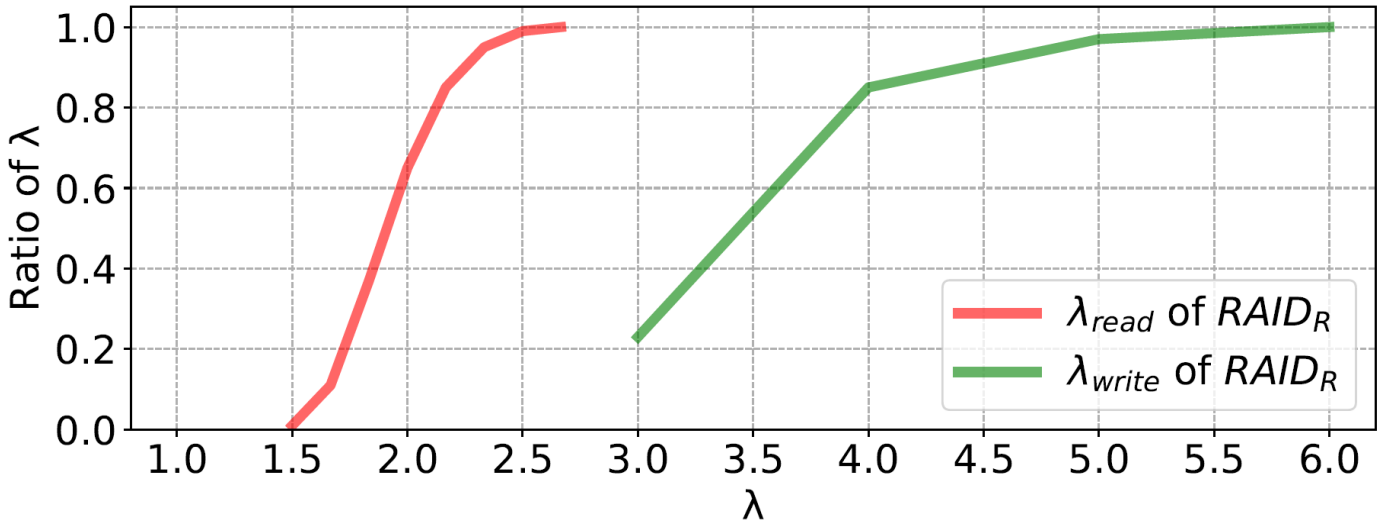
Background

- Reconstruction
  - Perform in batches
  - Reconstruct a certain number of chunks in each batch
    - Local load imbalance in a batch



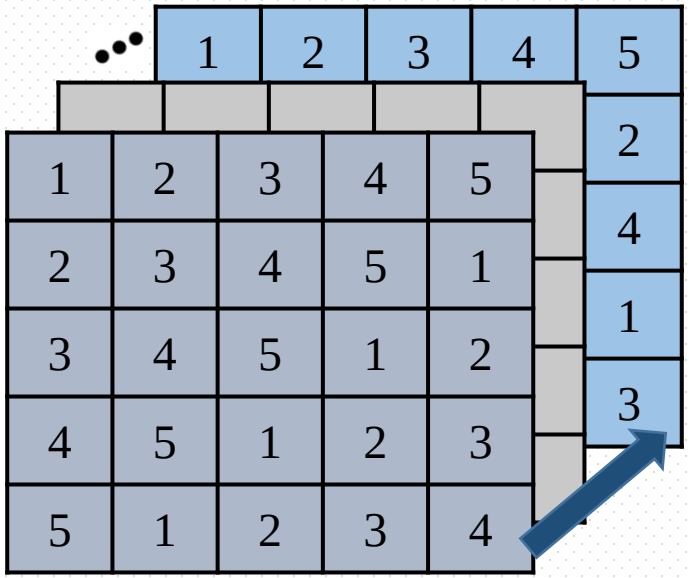
Motivation

- Evaluate local load imbalance in a batch
  - Define $\lambda_{read/write}$: ratio of the maximum number of read/write (in terms of chunks) to the average
  - Example: 59 disks, (6,1) RAID5, batch size=58, CDF of 100 batches
    - Read: the maximum load is 1.5x to 2.7x the average
    - Write: the maximum load is 3x to 6x the average



Existing Approaches

- Designing a dedicated data layout
  - Cost of relocating data is extremely heavy
    - a long-used status to a well-designed data layout
    - the interim to the normal



RAID+ [1]

c	a	a	a	b
d	d	b	b	c
e	e	e	c	d
g	f	f	g	f
i	h	g	h	h
j	j	i	j	i
i	k	l	k	k
m	m	n	m	l
o	n	o	o	n
q	r	p	p	p
r	s	s	q	q
s	t	t	t	r

Normal layout

c	a	a	a	b
d	d	b	b	c
e	e	e	c	d
g	f	f	g	f
i	h	g	h	h
j	j	i	j	i
i	k	l	k	k
m	m	n	m	l
o	n	o	o	n
q	r	p	p	p
r	s	s	q	q
s	t	t	t	r
	c	d	e	g
	i	j	l	m
	o	q	r	s

Interim layout

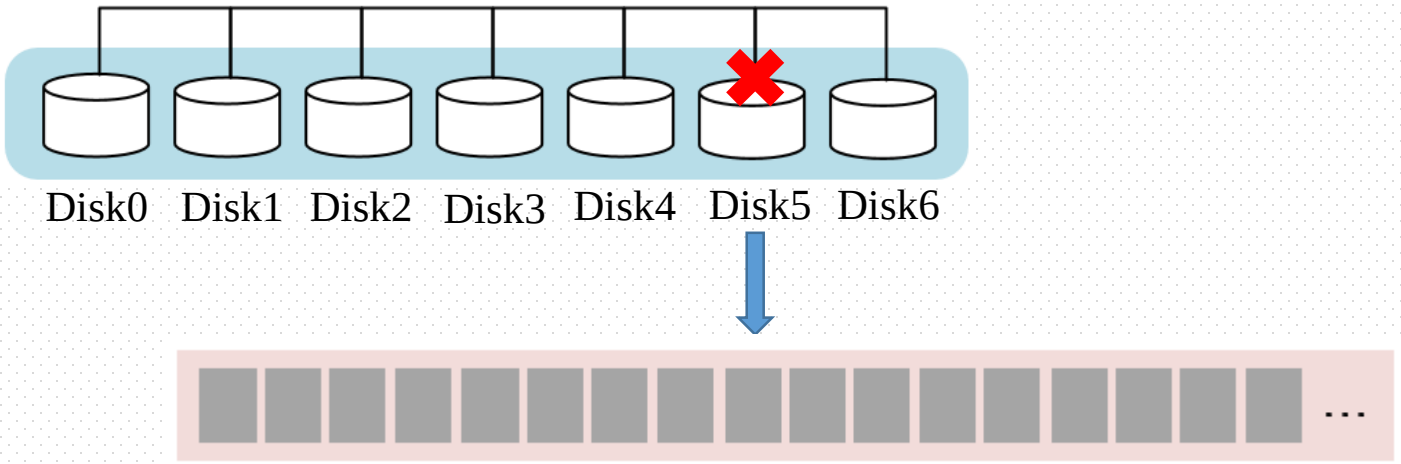
[1] RAID+: Deterministic and balanced data distribution for large disk enclosures - Zhang et al., at FAST'18

Existing Approaches

- Designing a dedicated data layout
  - Cost of relocating data is extremely heavy
    - a long-used status to a well-designed data layout
    - the interim to the normal
  - Batch size only depends on the mathematical properties

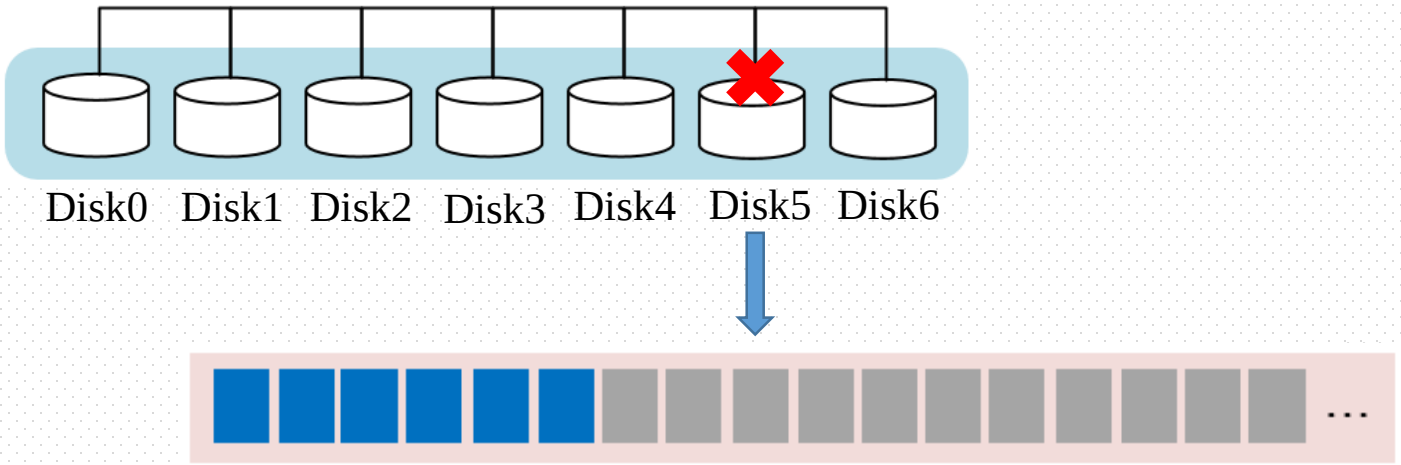
RAID2.0++

- Balance local reconstruction workloads without relocating costs



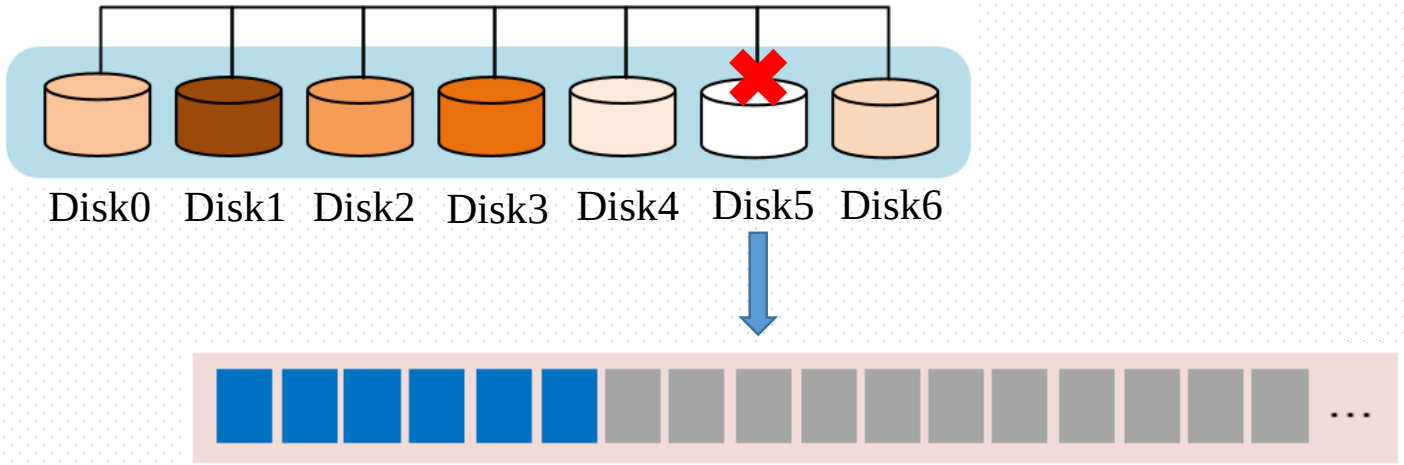
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks



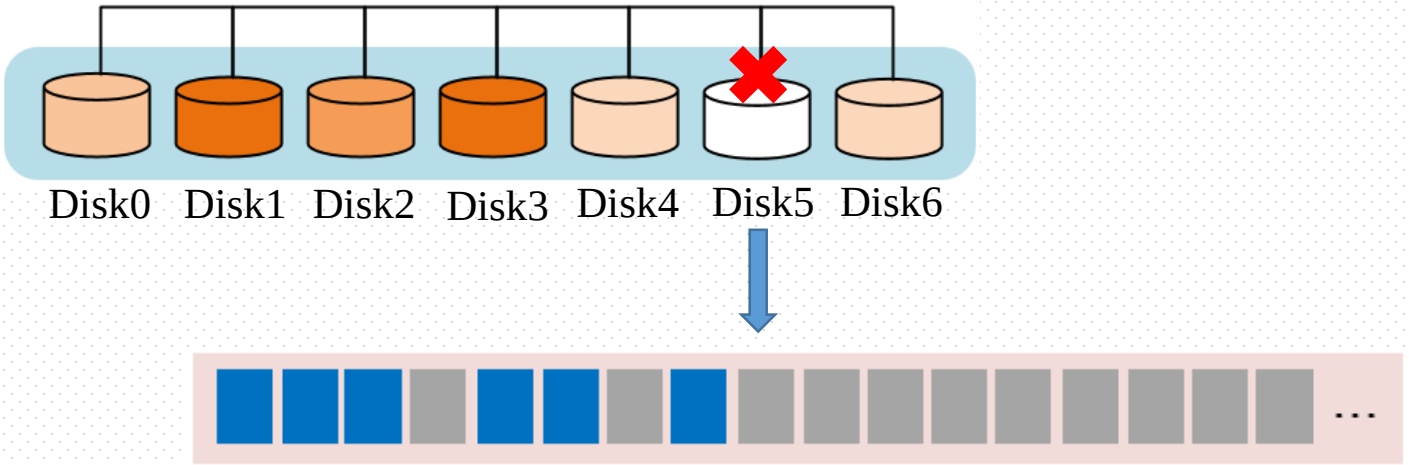
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks
  - Obtain the read load



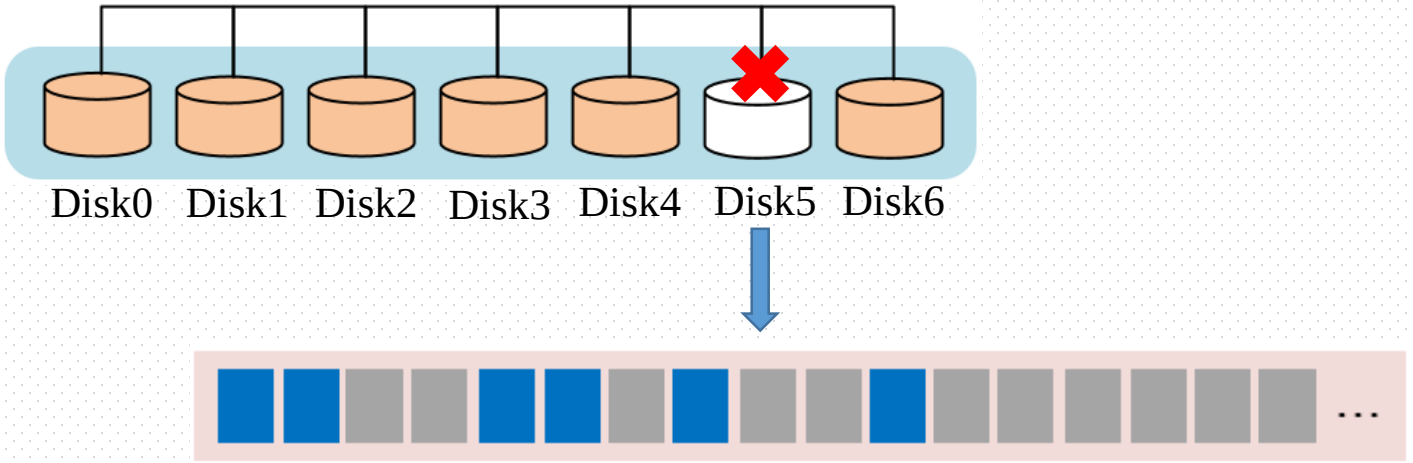
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks
  - Obtain the read load
  - Replace improper tasks with tasks reading from lighter loaded disks



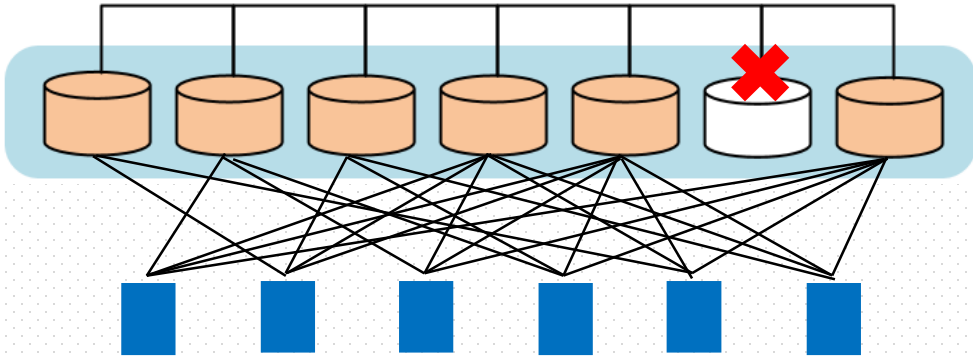
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks
  - Obtain the read load
  - Replace improper tasks with tasks reading from lighter loaded disks



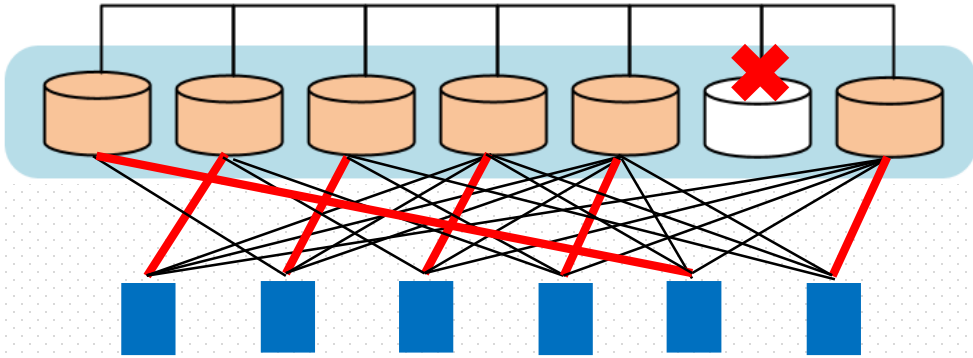
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks
  - Obtain the read load
  - Replace improper tasks with tasks reading from lighter loaded disks
  - Distribute reconstructed chunks based on matching theory



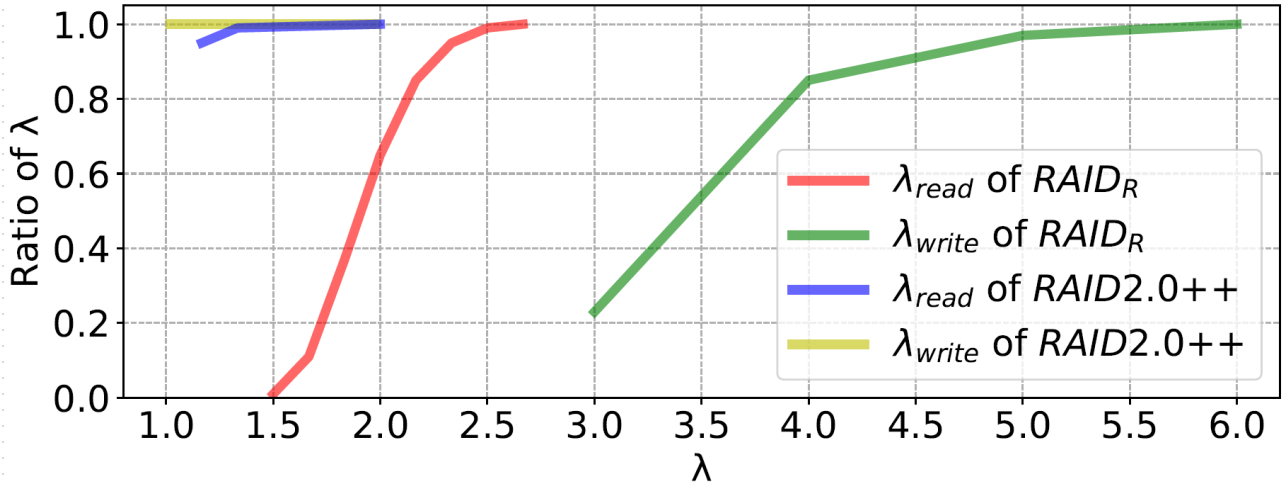
RAID2.0++

- Balance local reconstruction workloads without relocating costs
  - Determine the batch size and initialize a batch of tasks
  - Obtain the read load
  - Replace improper tasks with tasks reading from lighter loaded disks
  - Distribute reconstructed chunks based on matching theory



RAID2.0++

- Preliminary results
  - 59 disks, (6,1) RAID5, batch size=58, CDF of 100 batches
  - λ_{read}
    - RAID2.0++: around 1.167
    - $RAID_R$: 1.5 to 2.7
    - Improved by 41%
  - λ_{write}
    - RAID2.0++: 1 (i.e. complete balance)
    - $RAID_R$: 3 to 6
    - Improved by 74%



Following work

- Cope with dynamic workloads to determine the batch size
- Implementation
  - Isomorphic/Heterogeneous disks
  - Workloads with different features

That’s all! Thank you!