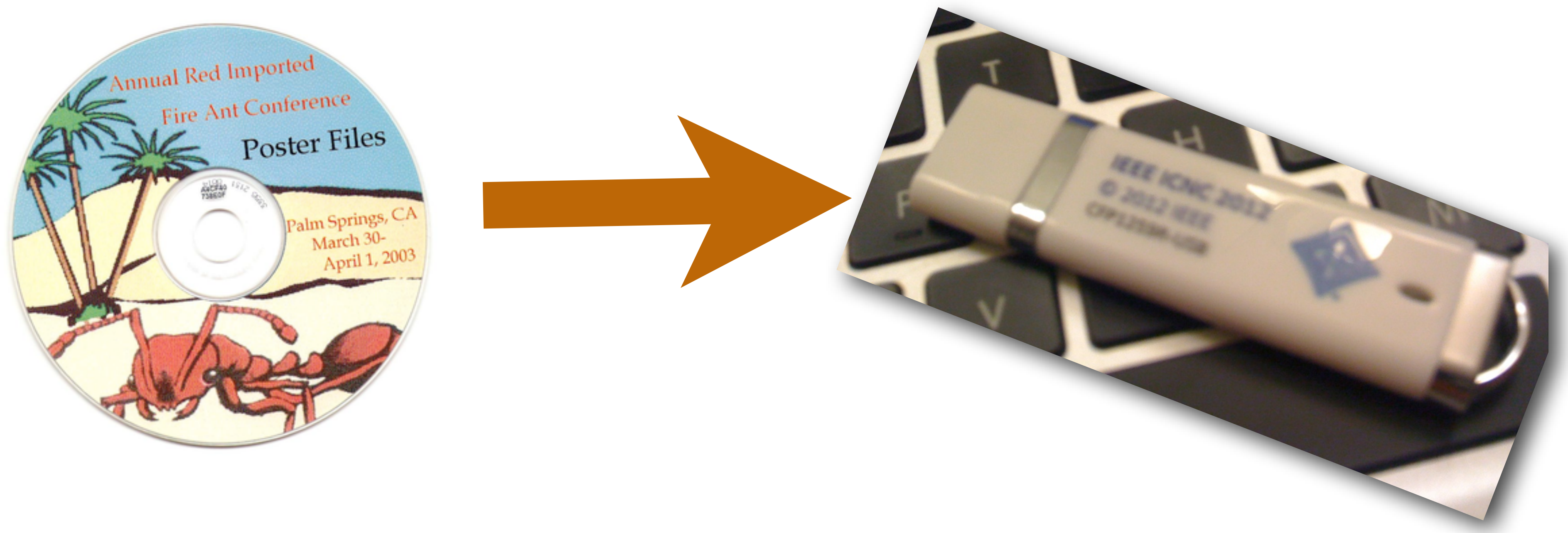
# An Improved Analytic Expression for Write Amplification in Flash Memory

Loujie Xiang
University of Electronic Science and Technology of China

Brian M. Kurkoski
University of Electro-Communications
Tokyo, Japan
kurkoski@ice.uec.ac.jp

**ICNC 2012**
**Lahaina, Hawaii, USA**
**February 1, 2012**

# Flash Memory is Quickly Becoming Ubiquitous




mobile devices

Embedded

USB memory


SSD

# Background

**Write amplification** is a problem in flash memory systems:

- More physical writes than logical writes, due to flash memory block-erase
  - Unneeded write operations is a system-level problem
  - Excessive writes cause failure of flash memory device

**Overprovisioning** mitigates this problem

- increase the physical memory without increasing logical memory

Numerous empirical studies

Agarwal and Marrow [Globecom 2010]

- an analytical expression for write amplification
  - analysis of a system using “greedy garbage collection”

# Contribution

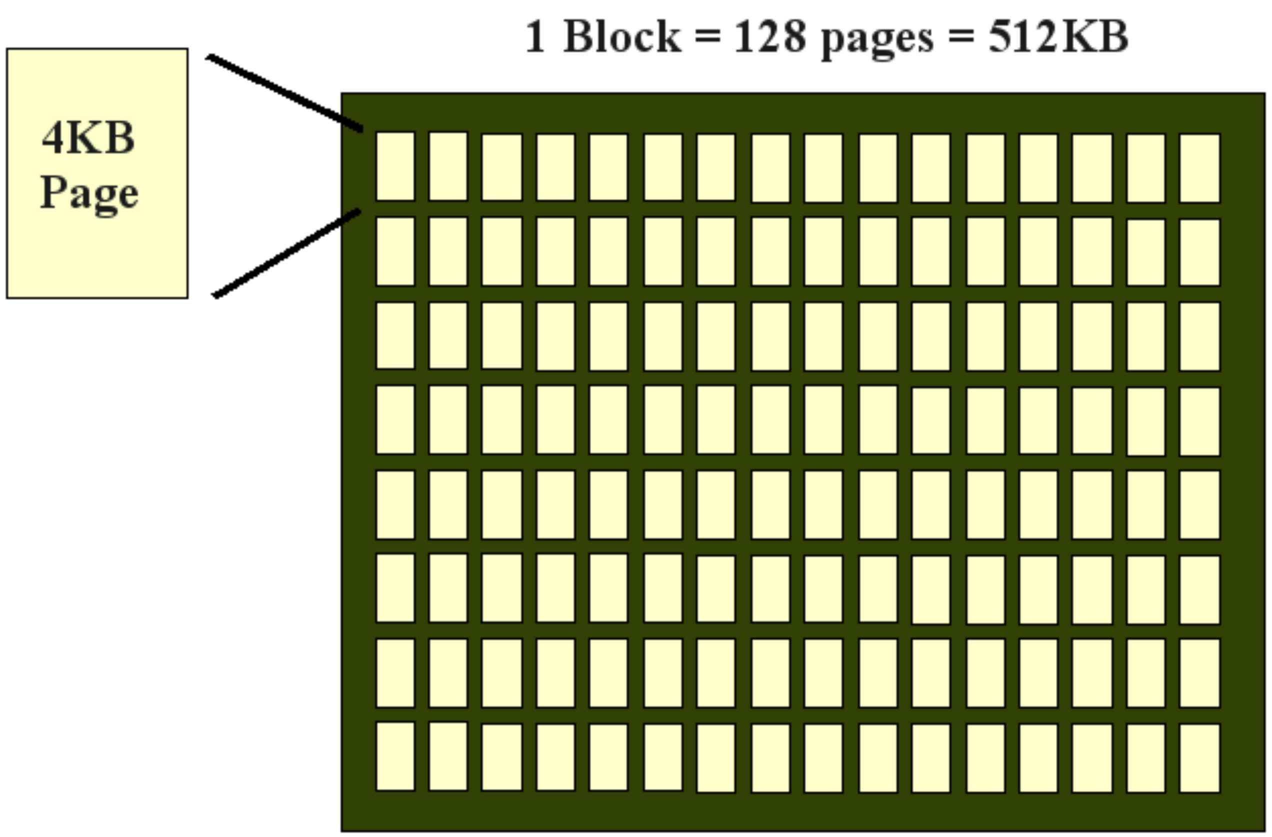
- A new analytic expression for write amplification
- More precise than previous expression

# Outline

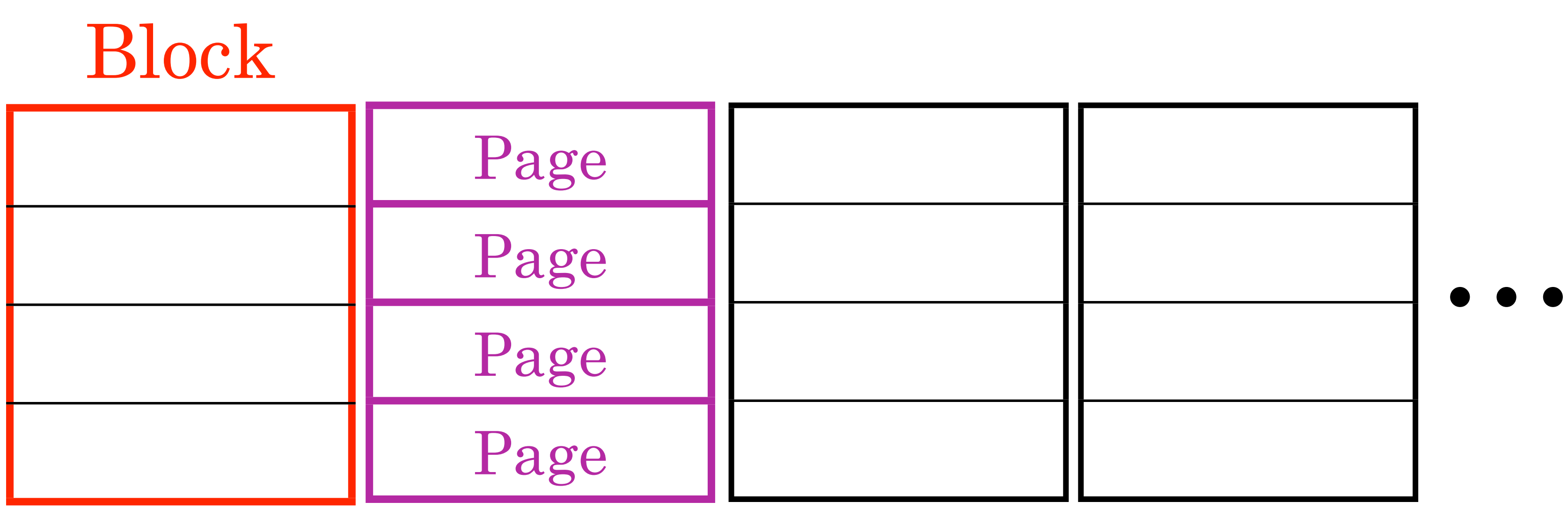
1. Flash memory system
2. Agarwal & Marrow analysis
3. Improved analytic expression

# Organization of flash memory

- Flash is a re-writable semiconductor memory
- Organization of flash memory
  - Contains thousands of blocks
  - A block contains typically 64 pages
  - A page is typically 4 KB, smallest unit
- Operations on flash memory
  - Page-level write operations
  - Can write only to empty blocks
  - Block-level erase operations

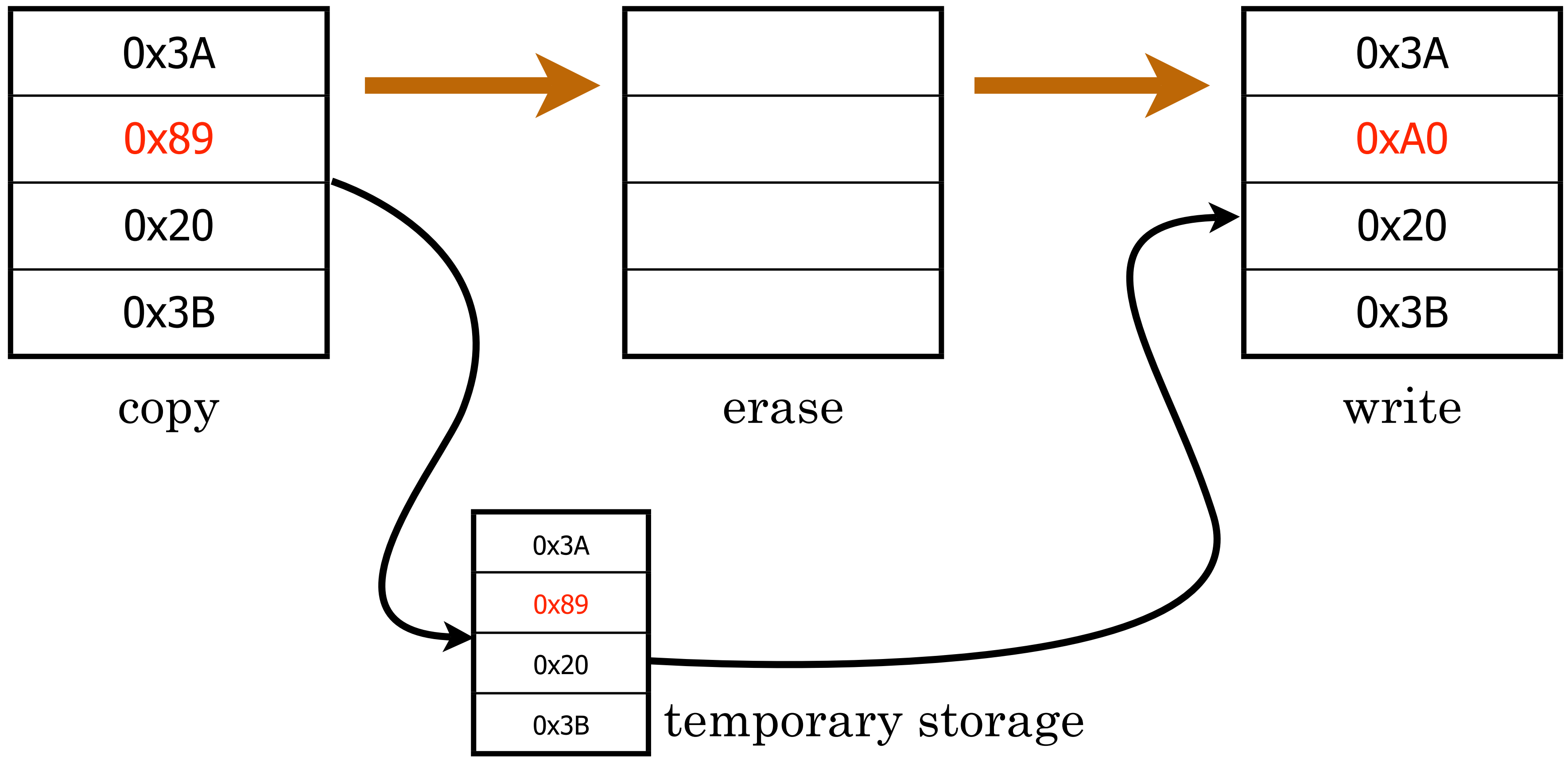



http://www.linux-mag.com/id/7590/



# Flash memory: Write Amplification

- To change one page, must copy-erase-write
- "Write amplification" Changing one page requires 64 page writes!
- Undesirable:
  - reduces system performance
  - reduces flash memory device longevity

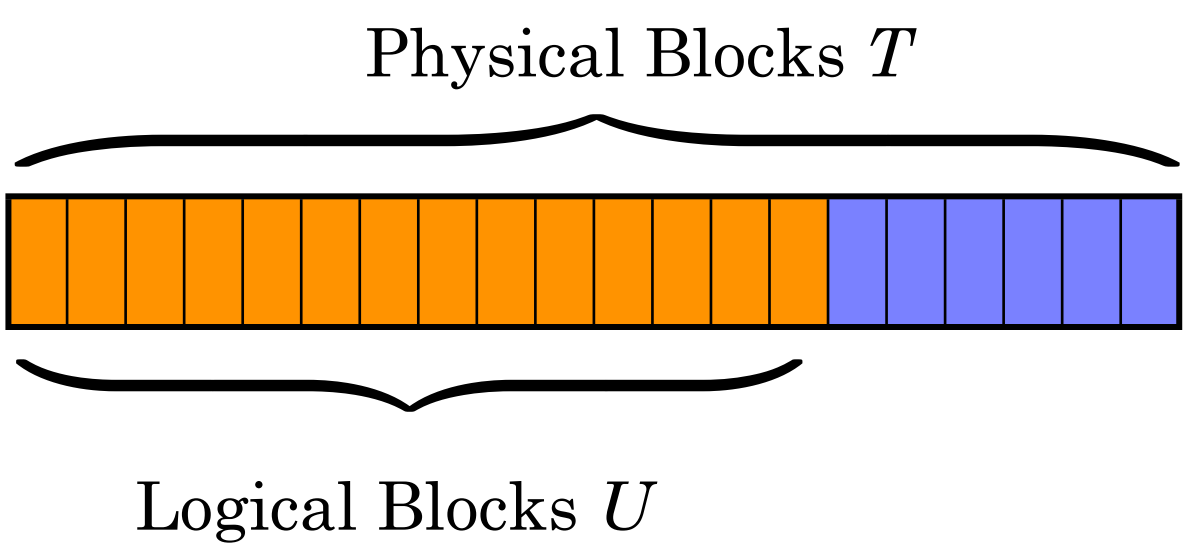


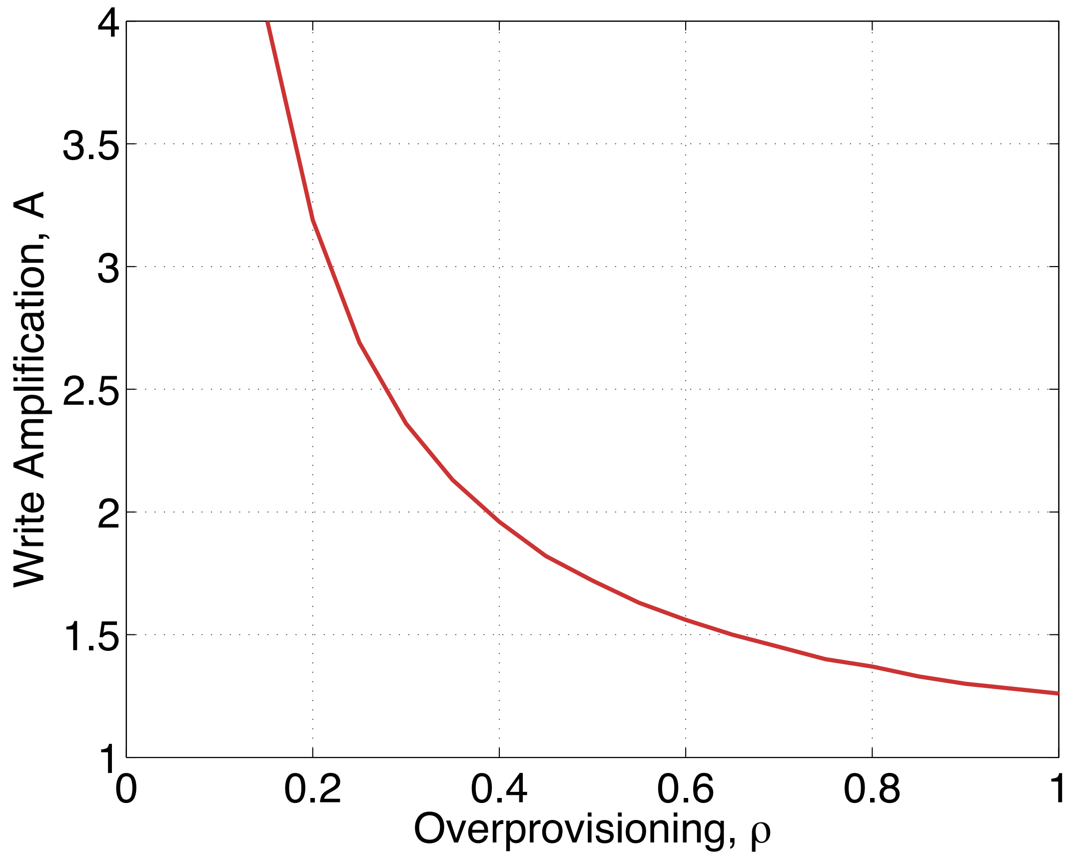
# Write Amplification and Overprovisioning

- **Problem:** Write Amplification

$$\text{Write Amplification } A = \frac{\text{Number of Physical Writes}}{\text{Number of Logical Writes}}$$

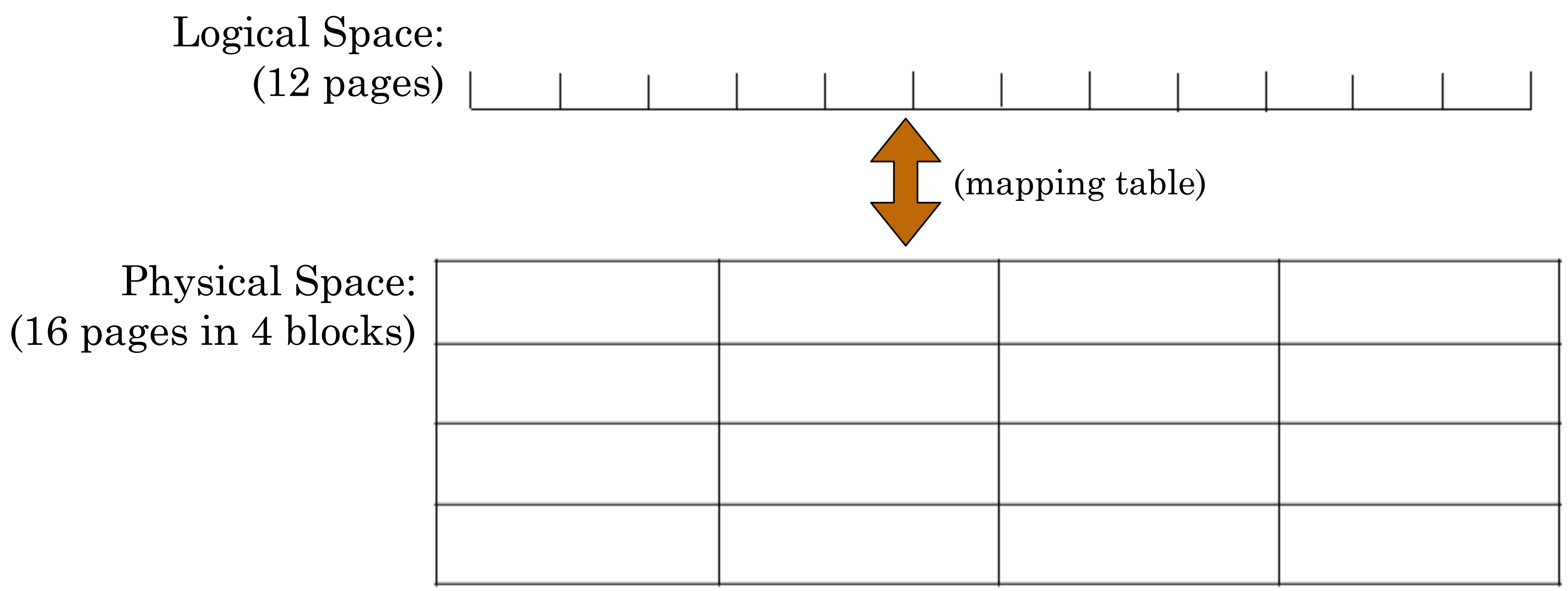
- **Solution:** Overprovisioning
  - More physical memory than logical memory
  - (some physical memory the user cannot see)




Overprovisioning factor $\rho = \frac{T-U}{U}$

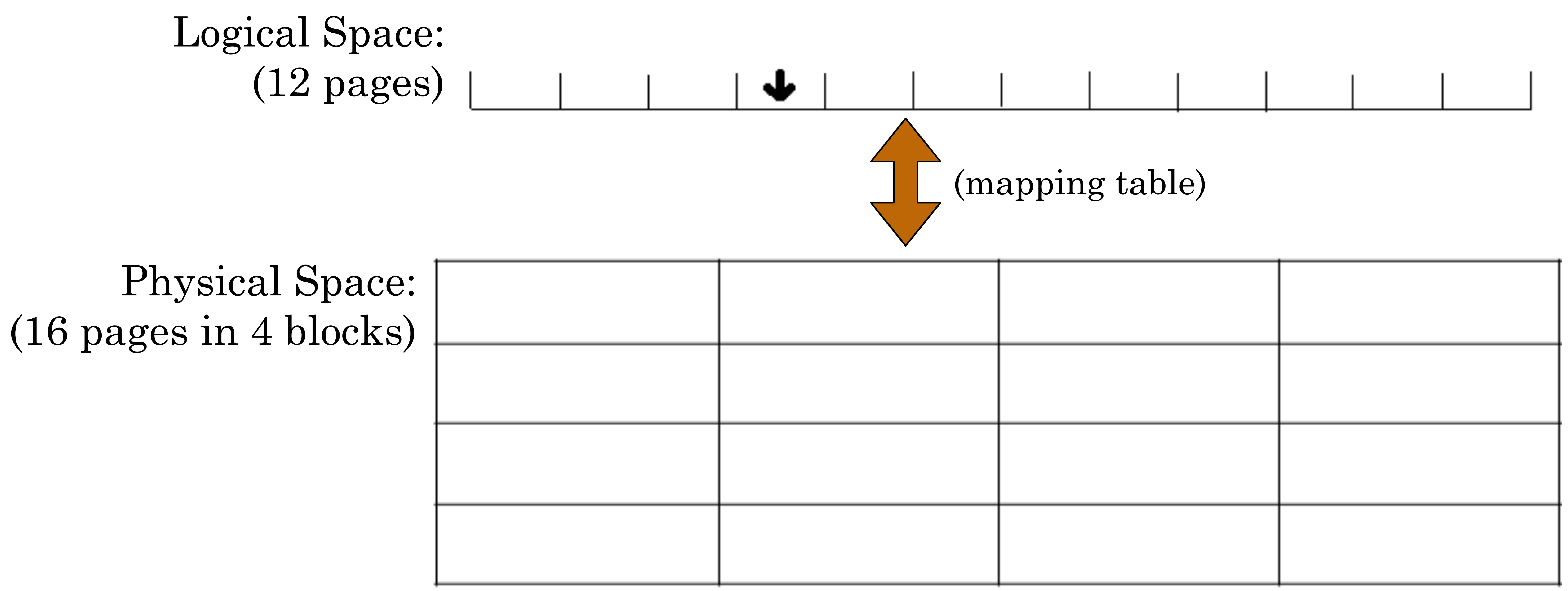
# Example of Writing Flash Memory




Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

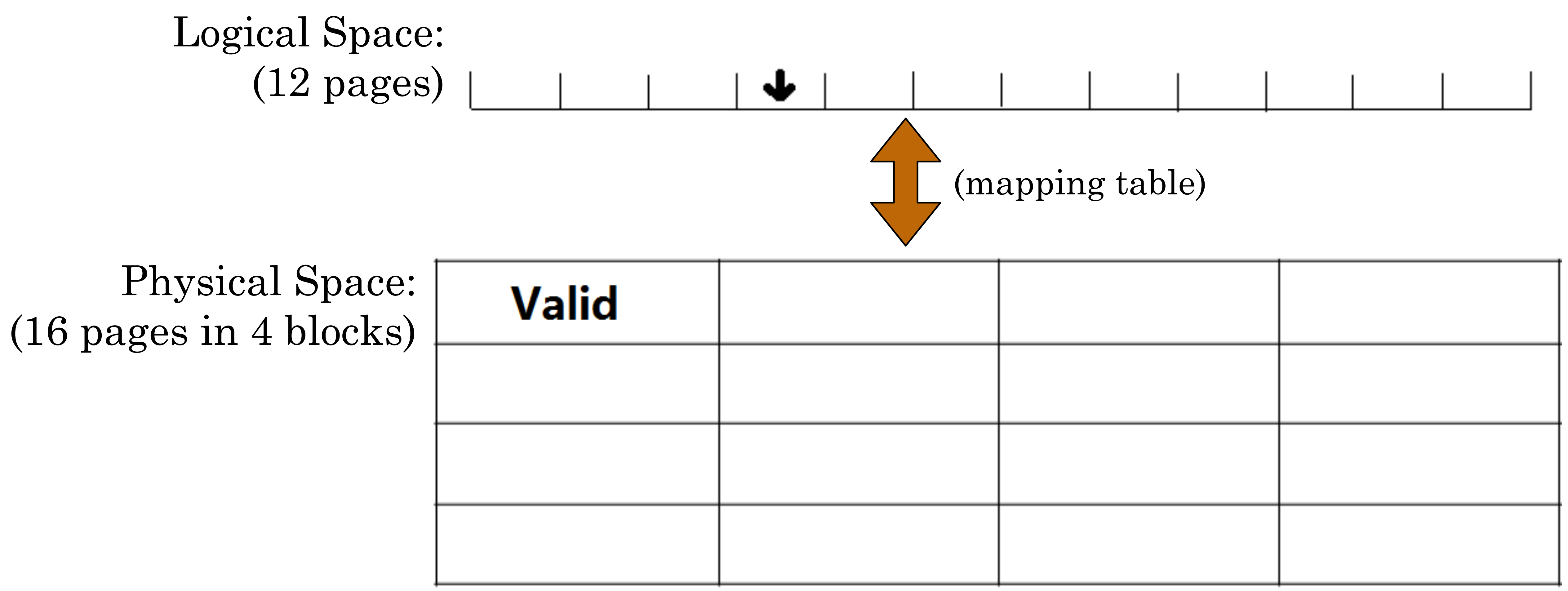
System

# Example of Writing Flash Memory




Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Example of Writing Flash Memory

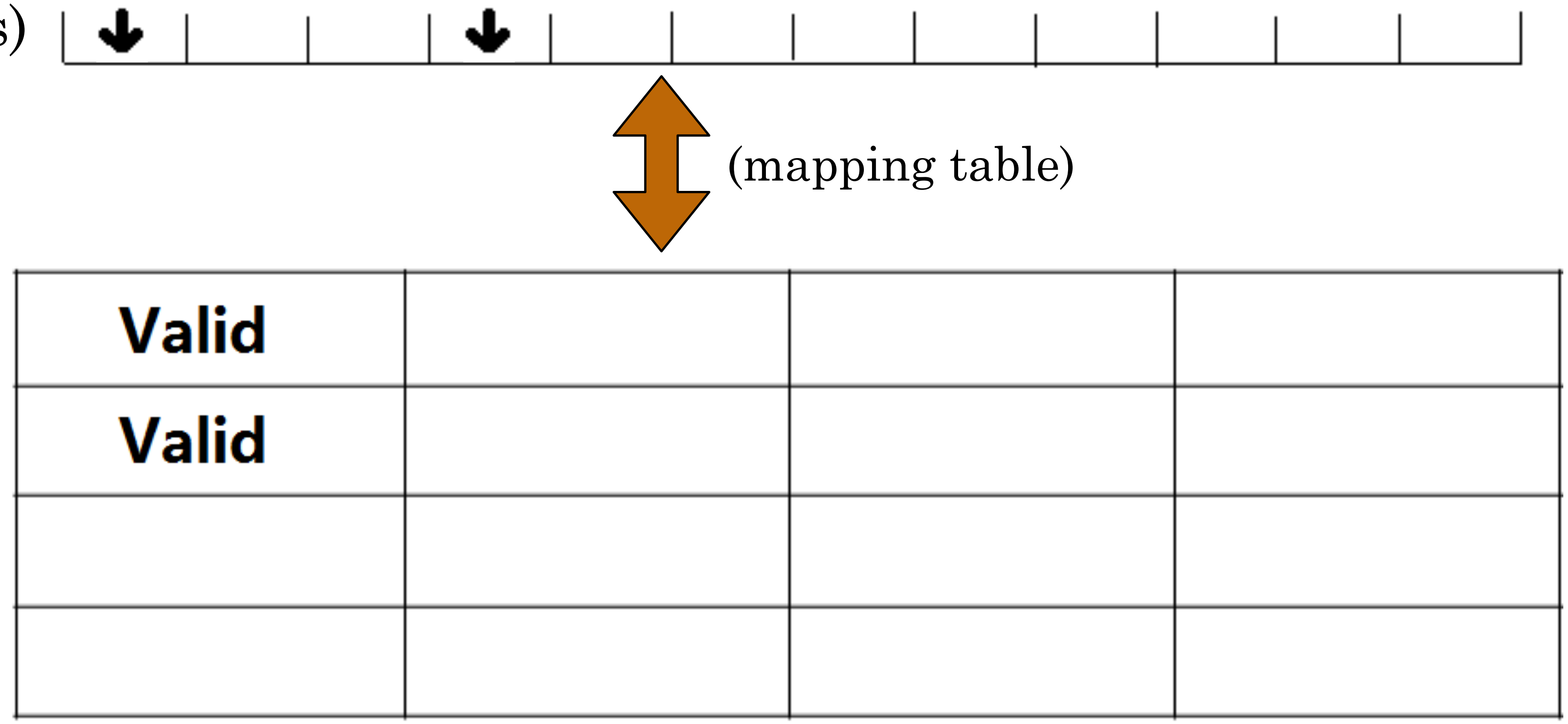



Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Example of Writing Flash Memory

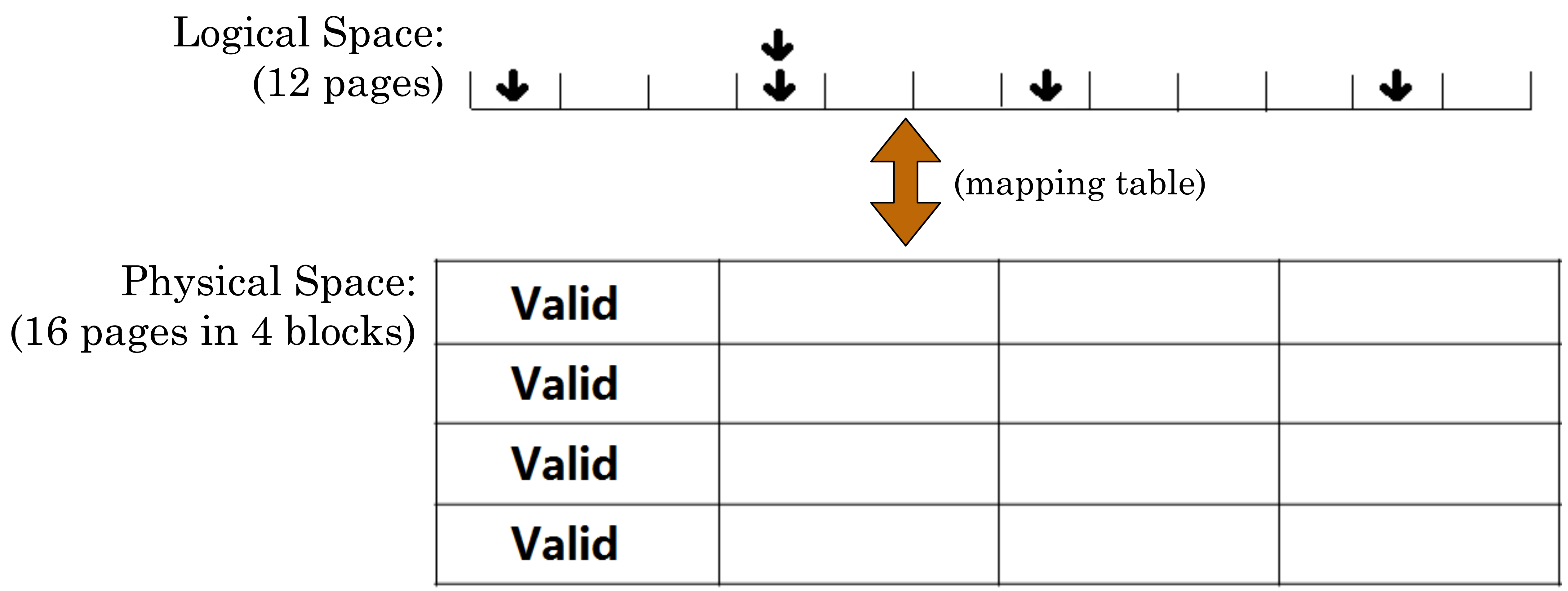
Logical Space:
(12 pages)

Physical Space:
(16 pages in 4 blocks)



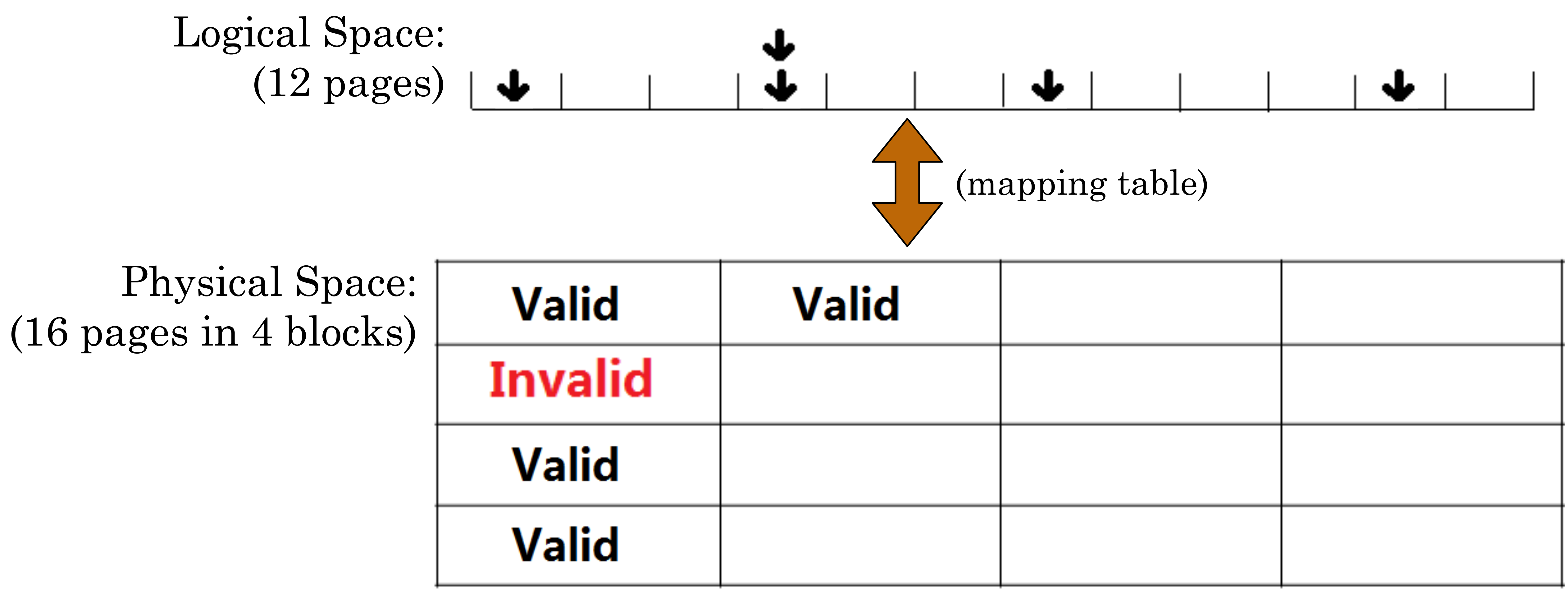

Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Example of Writing Flash Memory



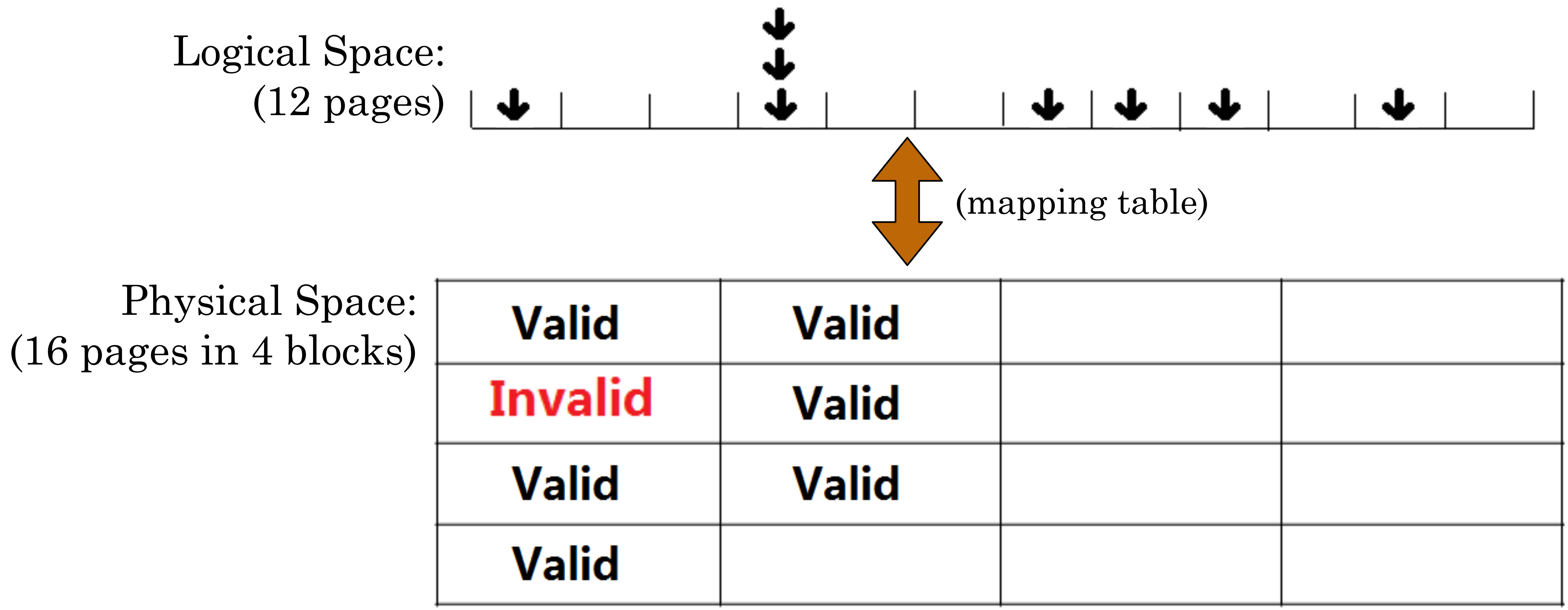

Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Example of Writing Flash Memory




Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

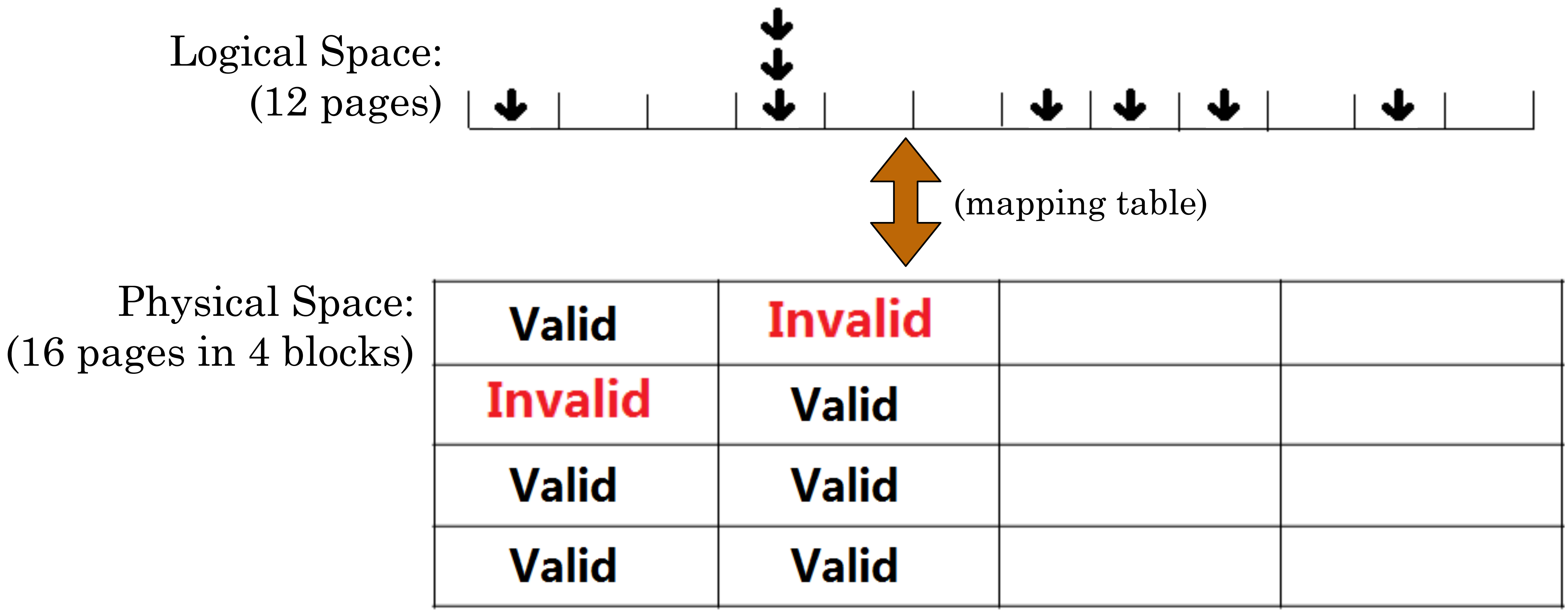
# Example of Writing Flash Memory




Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

## System

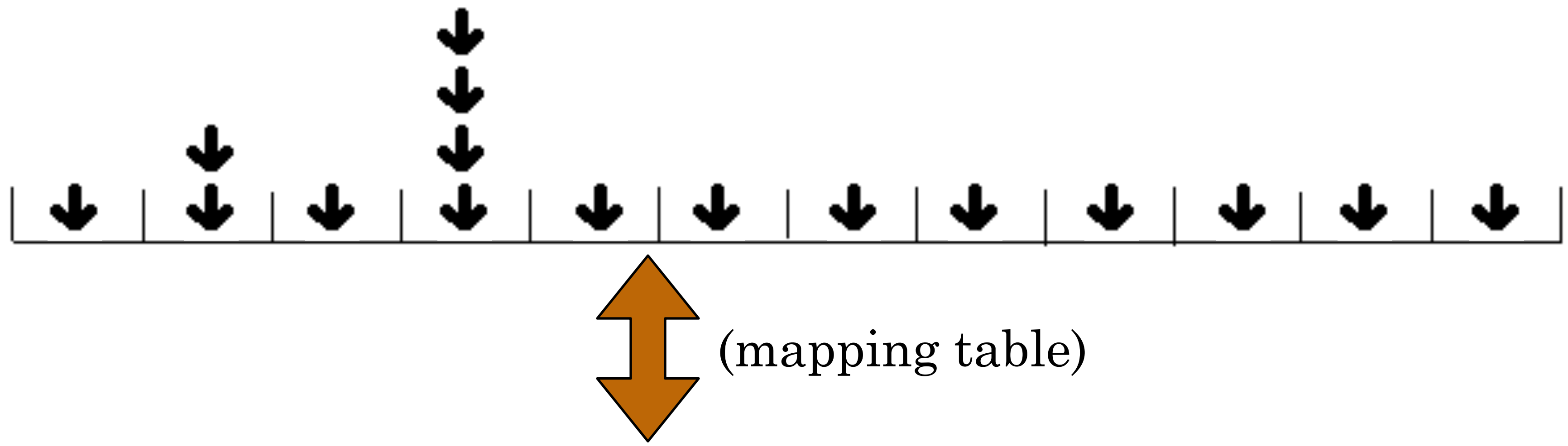
# Example of Writing Flash Memory




Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Example of Writing Flash Memory

Logical Space:
(12 pages)




Physical Space:
(16 pages in 4 blocks)

| | | | |
|---|---|---|---|
| Valid | Invalid | Valid | Valid |
| Invalid | Valid | Valid | Valid |
| Valid | Valid | Invalid | Valid |
| Valid | Invalid | Valid | Valid |

Initial condition: Start with an empty memory
User writes uniformly and randomly distributed on user space
stationary condition: Logical memory is always full (worst case)

# Garbage Collection

Logical Space:
(12 pages)

Physical Space:
(16 pages in 4 blocks)

| | | | |
|---|---|---|---|
| Valid | Invalid | Valid | Valid |
| Invalid | Valid | Valid | Valid |
| Valid | Valid | Invalid | Valid |
| Valid | Invalid | Valid | Valid |

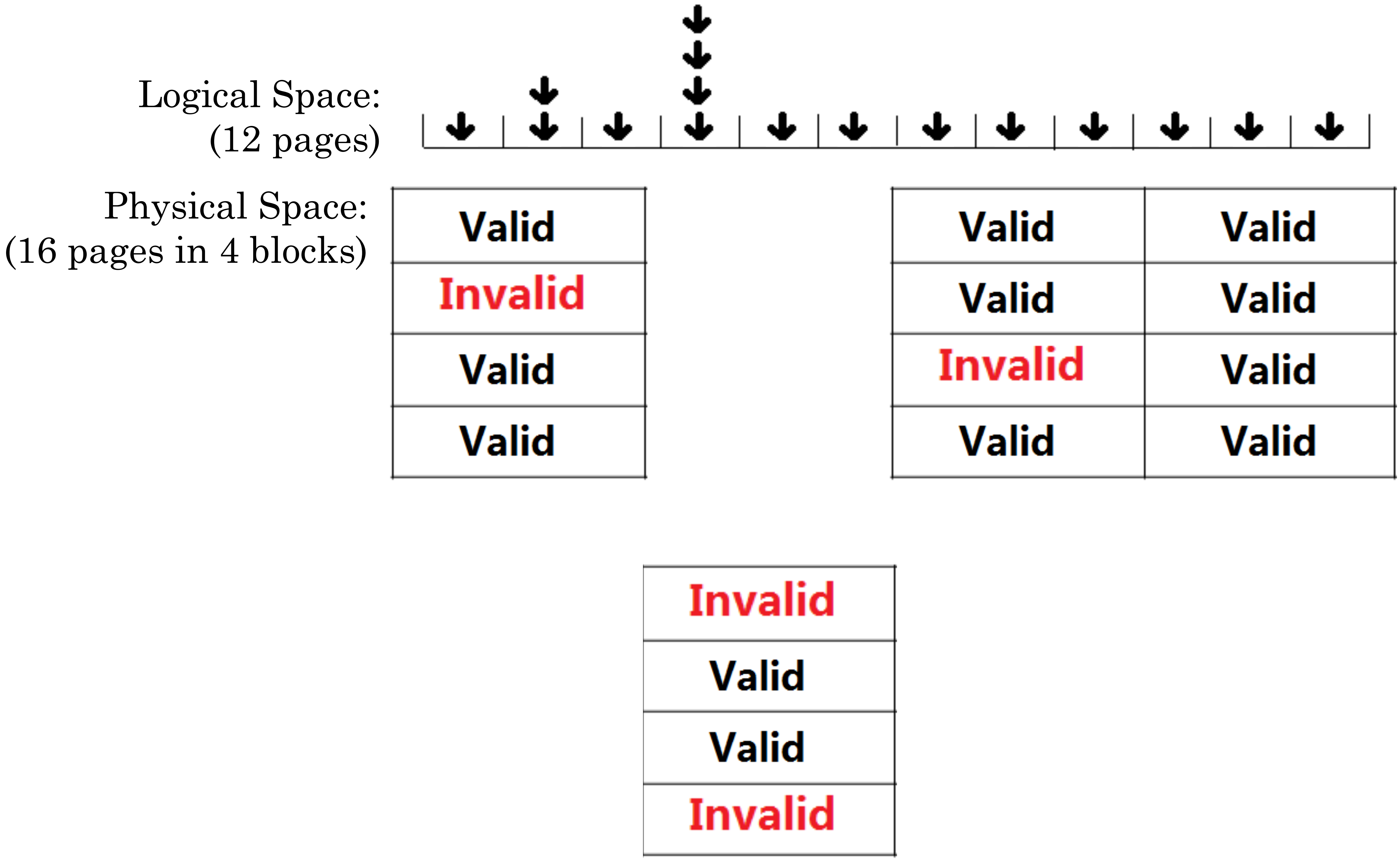
Time to erase

Greedy Garbage collection:

- Block with most invalid pages

Only two writes needed

# Garbage Collection



System

# Garbage Collection

Logical Space:
(12 pages)

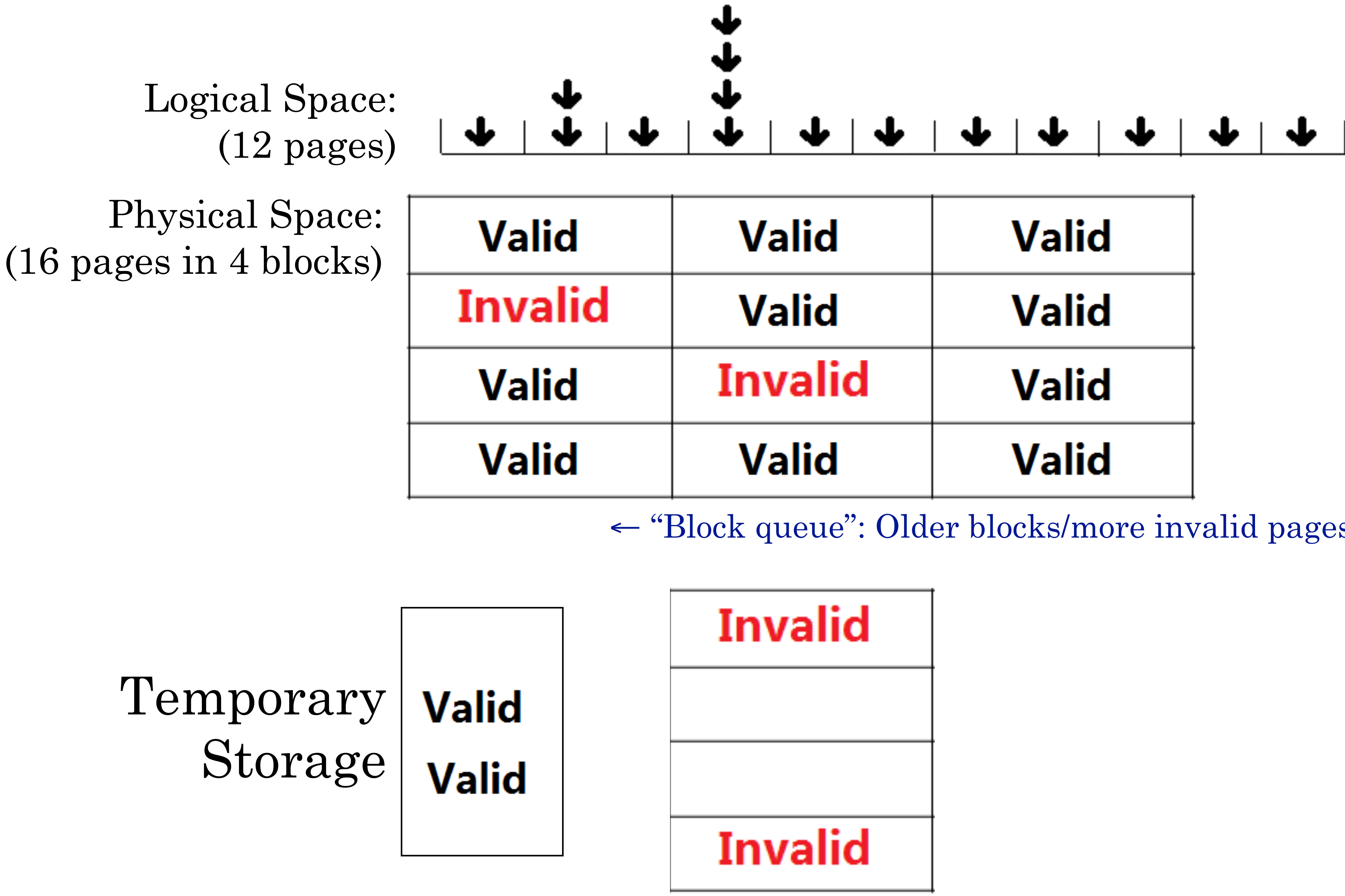
Physical Space:
(16 pages in 4 blocks)

| | | |
|---|---|---|
| Valid | Valid | Valid |
| Invalid | Valid | Valid |
| Valid | Invalid | Valid |
| Valid | Valid | Valid |

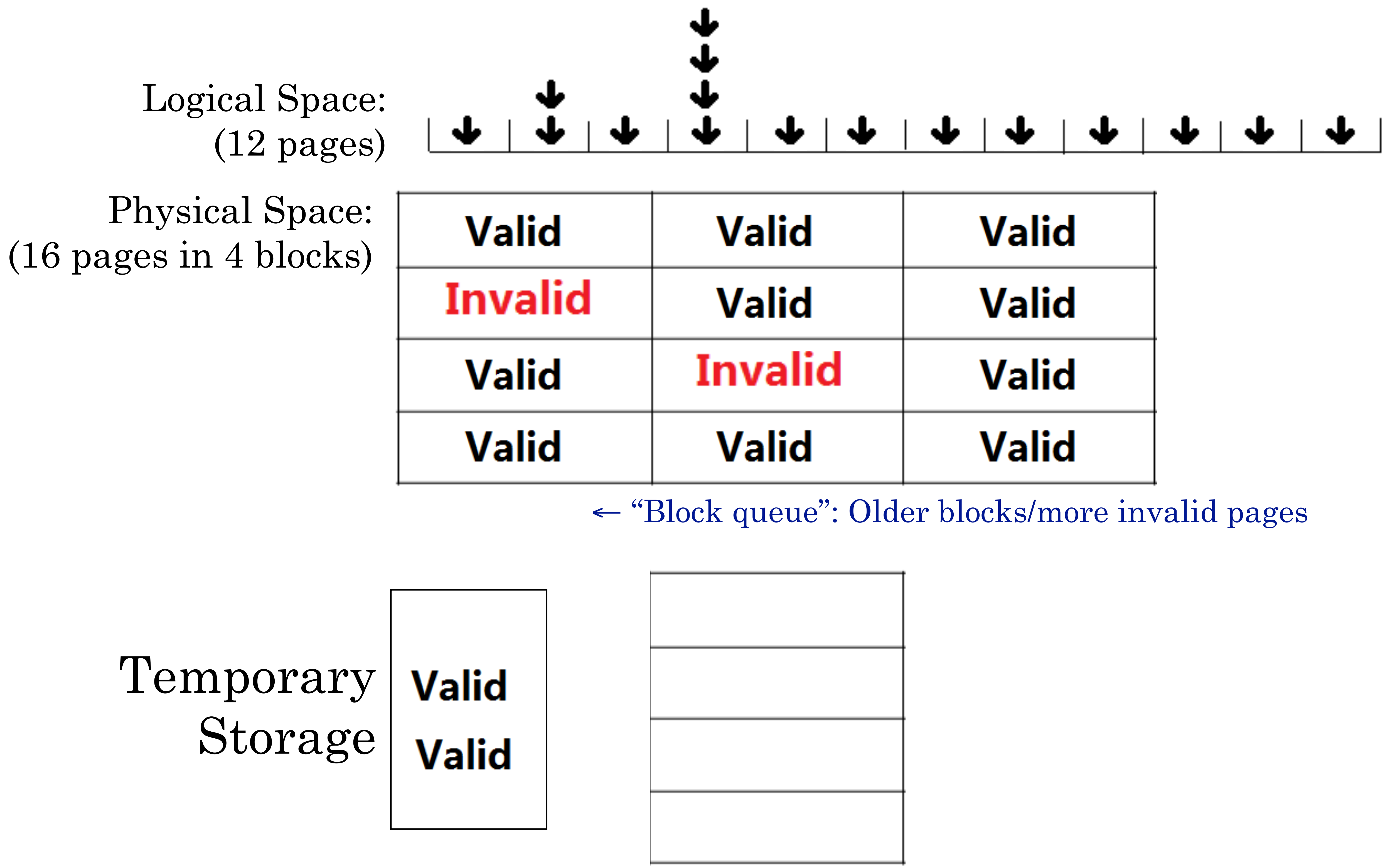
← “Block queue”: Older blocks/more invalid pages

| |
|---|
| Invalid |
| Valid |
| Valid |
| Invalid |

# Garbage Collection



# Garbage Collection



## System

# Garbage Collection

Logical Space:
(12 pages)

Physical Space:
(16 pages in 4 blocks)

| | | | |
|---|---|---|---|
| Valid | Valid | Valid | |
| Invalid | Valid | Valid | |
| Valid | Invalid | Valid | |
| Valid | Valid | Valid | |

← "Block queue": Older blocks/more invalid pages

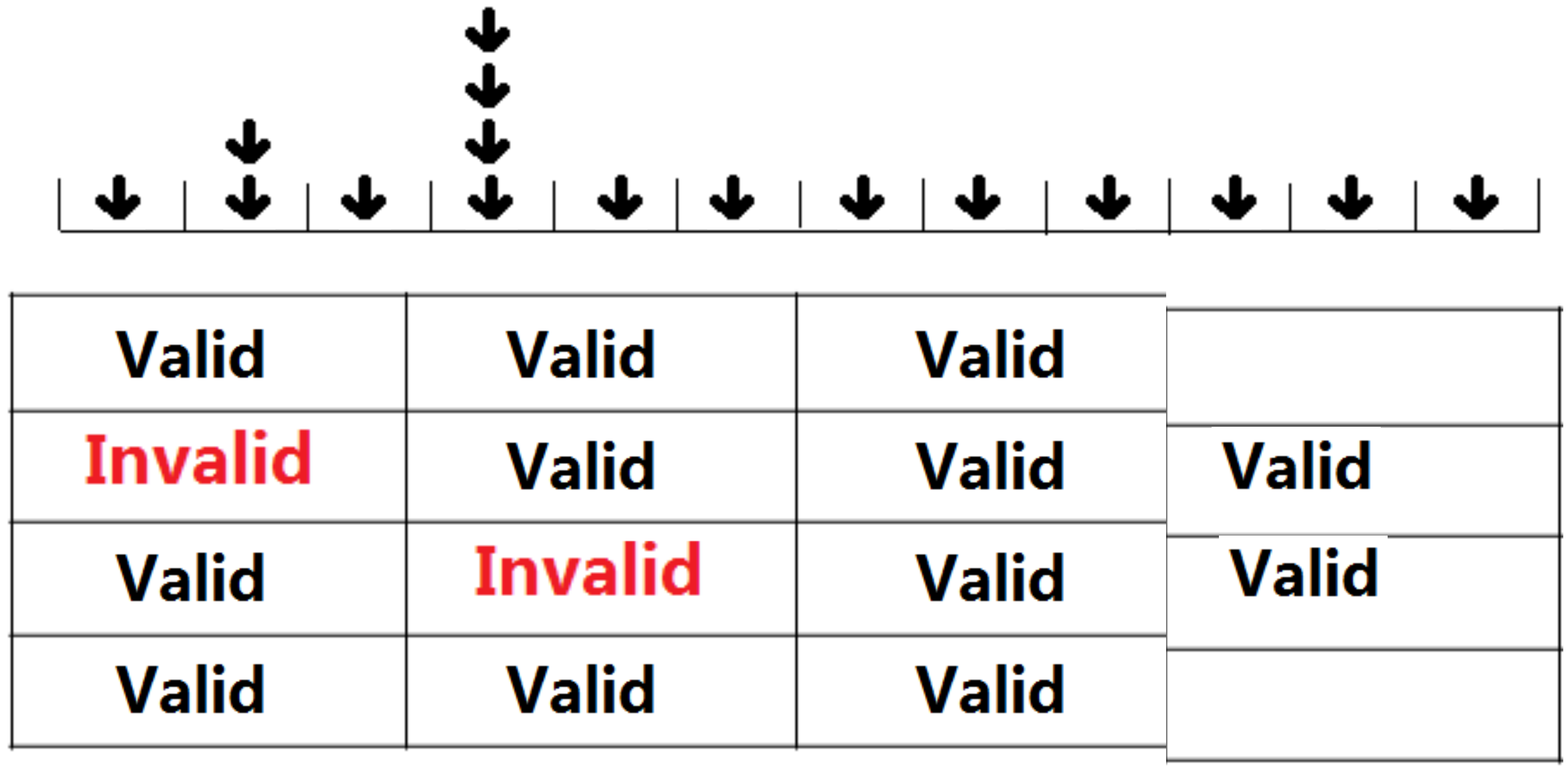
Temporary Storage

| |
|---|
| Valid |
| Valid |

# Garbage Collection

Logical Space:
(12 pages)

Physical Space:
(16 pages in 4 blocks)




Temporary Storage

Time to erase

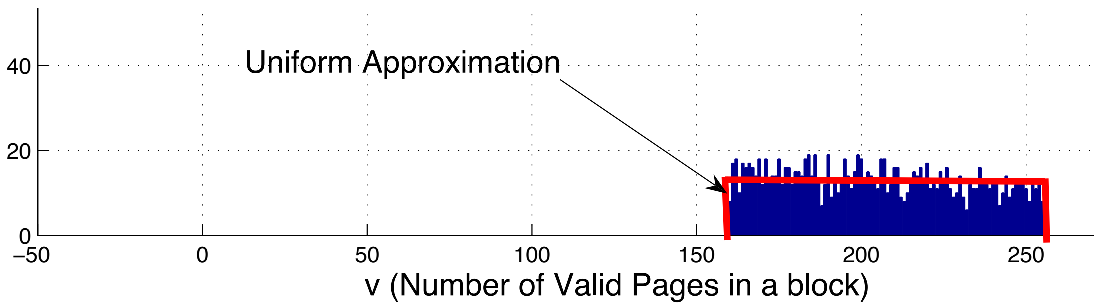
Greedy Garbage collection:

- Block with most invalid pages

Only two writes needed

## Analysis

# Agarwal & Marrow Approach




The number of valid pages **in a block** (over all blocks)

- Assumed uniform distribution

The number of valid pages **per block** (over one block)

- Assumed binomial distribution

Two different ways to find the **expected number of valid blocks**

- set them equal
- found an analytic expression for write amplification $A$:

$$A = \frac{1+\rho}{2\rho}$$

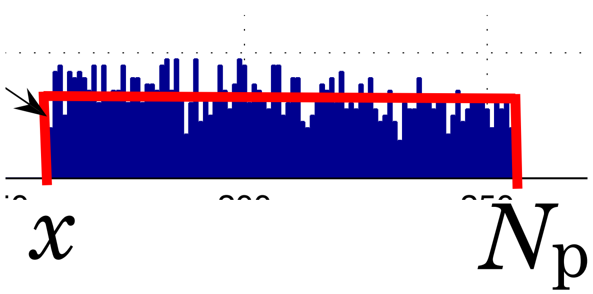
- Simple expression

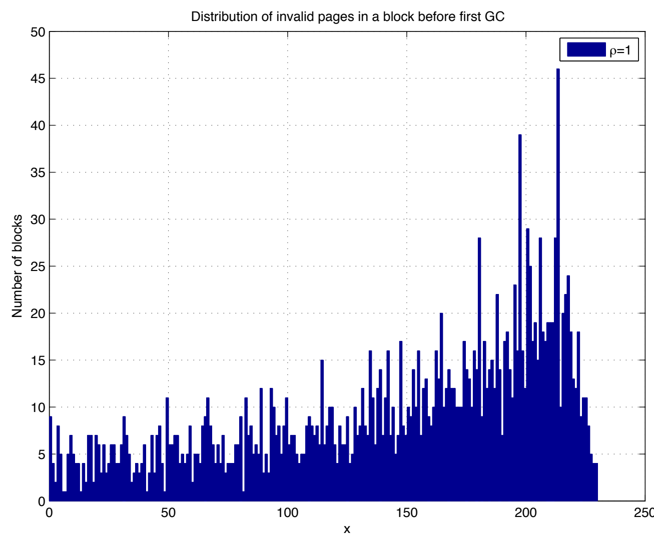
Overprovisioning factor $\rho$

# Analysis
# Weakness in Previous Work

Technique: Solve for the an "average block"

$$
\begin{aligned}
\text{Expected number of valid pages in a block} &= \text{Expected number of valid pages in a block} \\
\text{Mean of the uniform distribution} &= \text{mean of a binomial distribution} \\
\frac{N_{\mathrm{p}} + x}{2} &= pN_{\mathrm{p}}
\end{aligned}
$$




Weaknesses:

- the binomial probability is not $p = U/T$
- The binomial distribution is not truly binomial
- The uniform distribution is not truly uniform

# Our Approach

Technique: Solve for the block selected for garbage collection

- Each garage collection, $x$ invalid blocks are freed

A “new” block only has valid pages

- Consider a block that journeys from “new” to “old”
- For each logical write: a page has some small probability of being hit

For an “old” block ready for garbage collection

- There were $Tx$ writes ($Tx$ chances to become invalid)
- # of invalid pages = # blocks per page × probability of being invalid

$$x = N \times \big(1 - (1-p)\big)^{Tx}$$

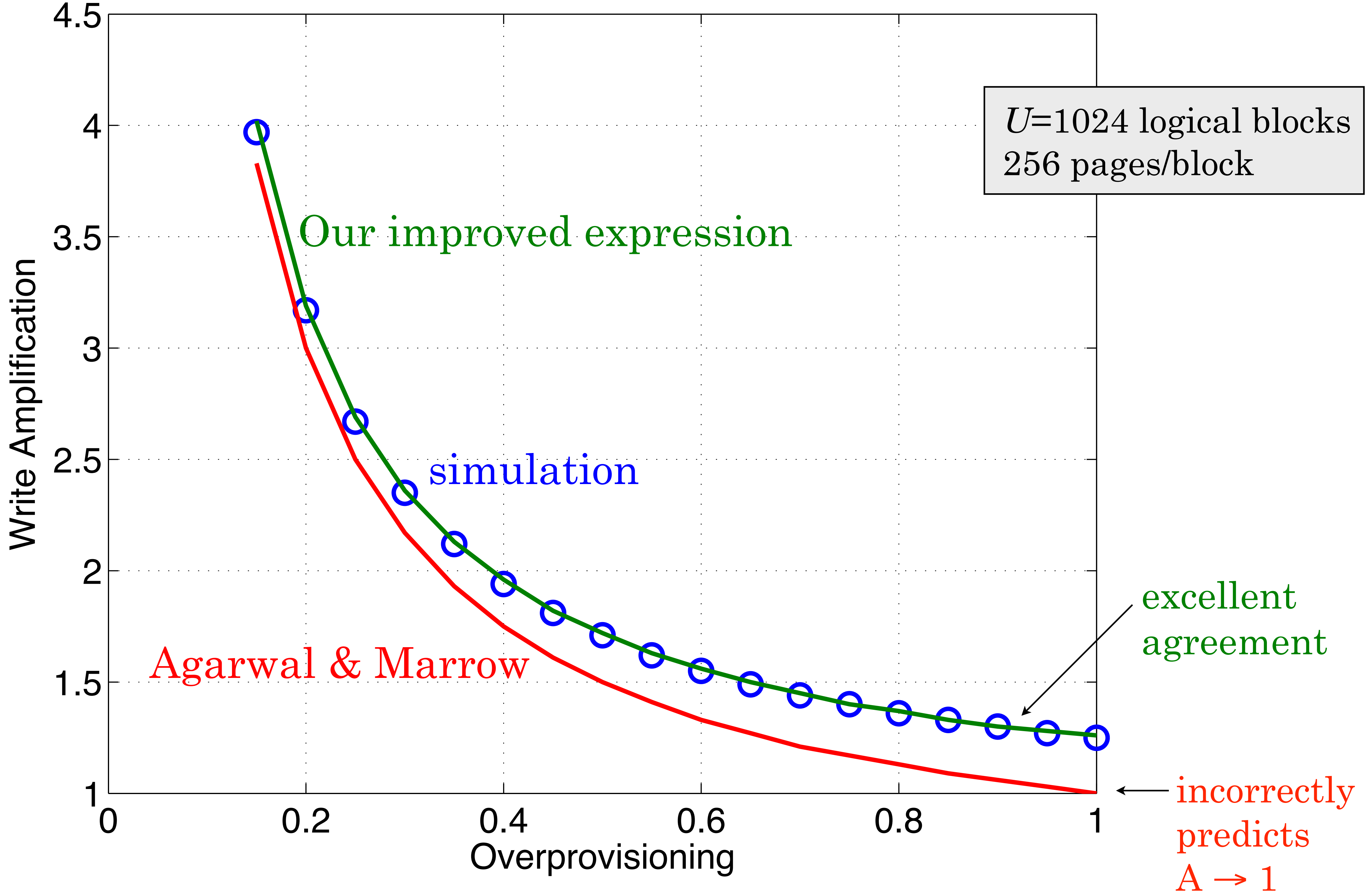
... Main Result — Obtain write amplification:

$$A = \frac{1+\rho}{1+\rho+W\big(-(1+\rho)e^{-(1+\rho)}\big)}$$

$W(.)$ is the Lambert W function. The solution to $c = xe^x$ is $W(c)$.

Let the number of pages $\to\infty$. Reasonable, since flash memories are huge.

# Improved Prediction of Write Amplification



## Discussion & Conclusion

**Write amplification** is a problem in flash memory systems:

- More physical writes than logical writes, due to flash memory block-erase
  - Unneeded write operations is a system-level problem
  - Excessive writes cause failure of flash memory device

**Overprovisioning** mitigates this problem

**Contribution**: Gave an improved analytic description of write amplification

$$A = \frac{1+\rho}{2\rho} \qquad\qquad A = \frac{1+\rho}{1+\rho+W\big(-(1+\rho)e^{-(1+\rho)}\big)}$$

Agarwal & Marrow

- Simple
- Used "loose" assumptions
- Analyzed an average block
- Not always accurate

Our improved expression

- Closed form (with W func.)
- Used tighter assumptions
- Analyzed garbage collected block
- Much more accurate