

Speeding up Analysis of Archived Sensor Data with Columnar Layout for Tape

Ken Iizawa¹, Seiji Motoe², Yuji Yazawa², and Masahisa Tamura¹

¹Fujitsu Laboratories Ltd.

²TOYOTA MOTOR CORPORATION

Prevalence of IoT

- Devices are increasingly generating a lot of sensor data
 - e.g. vehicle, surveillance camera, home appliance
- Analyzing archived sensor data provides valuable insights
 - e.g. forensic purpose



- Sensor data generated by connected cars
- Useful for gaining various insights
 - e.g. driver identification, failure prediction
- Structured data consisting of timestamp and sensor values
 - commonly accessed by applications via SQL

<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>...</i>	<i>column j</i>
t_1	v_{11}	v_{12}		v_{1j}
t_2	v_{21}	v_{22}		v_{2j}
t_3	v_{31}	v_{32}		v_{3j}
t_4	v_{41}	v_{42}		v_{4j}
t_5	v_{51}	v_{52}		v_{5j}
t_i	v_{i1}	v_{i2}		v_{ij}

Need for Cost-effective Archival Storage



- Large amount of data generated by a single car
 - 100s+MB/month^[1]
- Rapid increase in the number of connected cars
 - 10Ms cars in 2025^[1]

[1] Envisioning Smart Mobility Society in the Connected Future
https://www.itu.int/en/ITU-T/Workshops-and-Seminars/20190123/Documents/Shinichi%20Taniguchi_Presentation.pdf

Tape is not Dead

- Most cost-efficient of all storage devices
 - 1/15 of HDD's TCO^[2]
- Inexpensive because of
 - Separation between expensive drives and inexpensive cartridges
 - Allows for low ratio of drives to cartridges
 - low \$/GB
 - Expected to keep decreasing

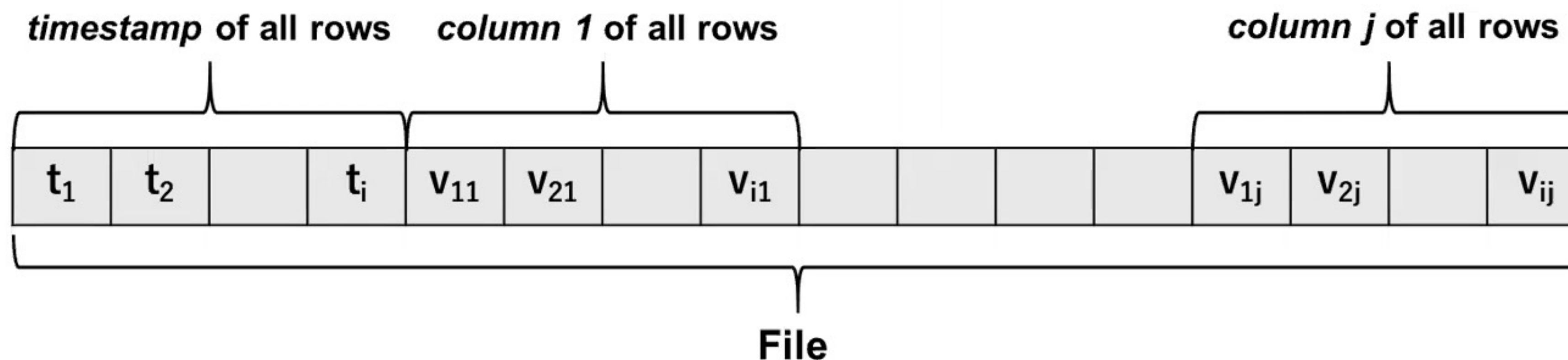


https://www.fujitsu.com/jp/Images/lt140_lineup_tcm102-4101728.jpg

[2] Tape Storage Is 'Still Here'
<https://www.itprotoday.com/backup/tape-storage-still-here>

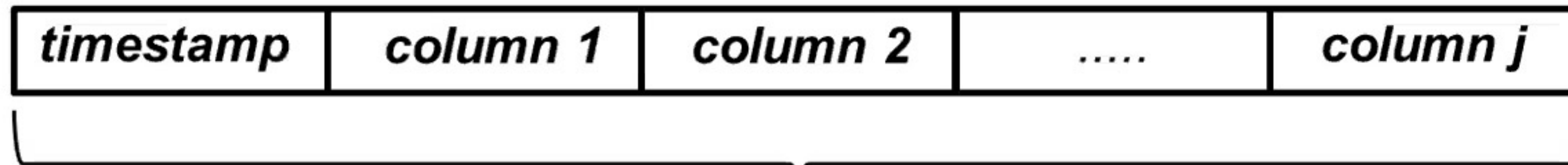
Analytical Workloads

- Read a few columns of many rows
 - instead of reading all the columns of a few rows
- Work well with column-oriented format
 - each column can be read sequentially



Challenges of Storing Data in Column-oriented Format on Tape

■ Storing a single column-oriented format file in a cartridge



Tape Cartridge

■ Pros

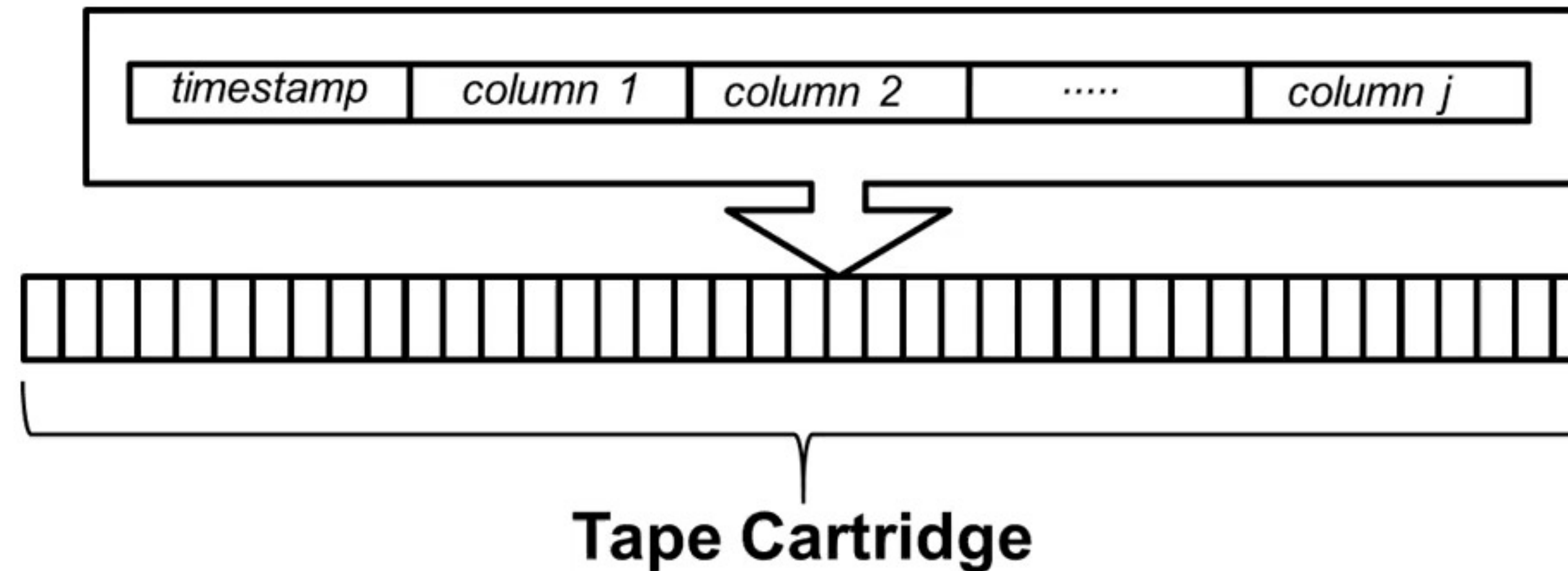
- achieves best read performance
 - 360 MB/s (LTO-8)

■ Cons

- requires the buffer of the same size as an entire cartridge
 - 12 TB (LTO-8) => 192 TB (LTO-12)

Challenges of Storing Data in Column-oriented Format on Tape (cont'd)

■ Storing many column-oriented format files in a cartridge



■ Pros

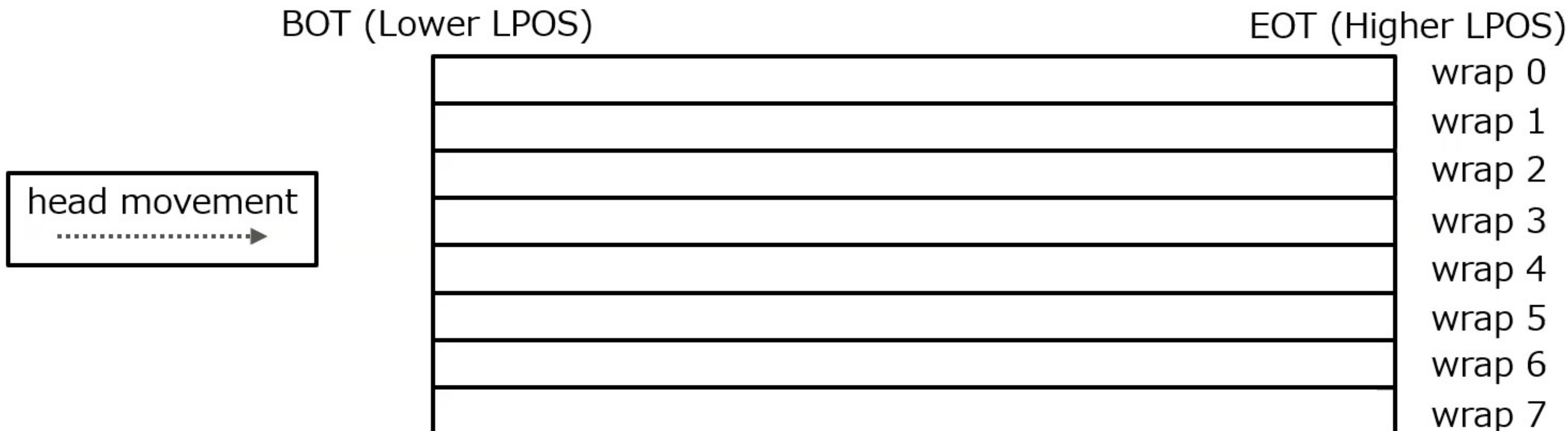
- makes the best use of a small buffer

■ Cons

- does not provide any performance improvement
 - All files in the cartridge need to be read, which takes almost the same time as scanning the entire cartridge

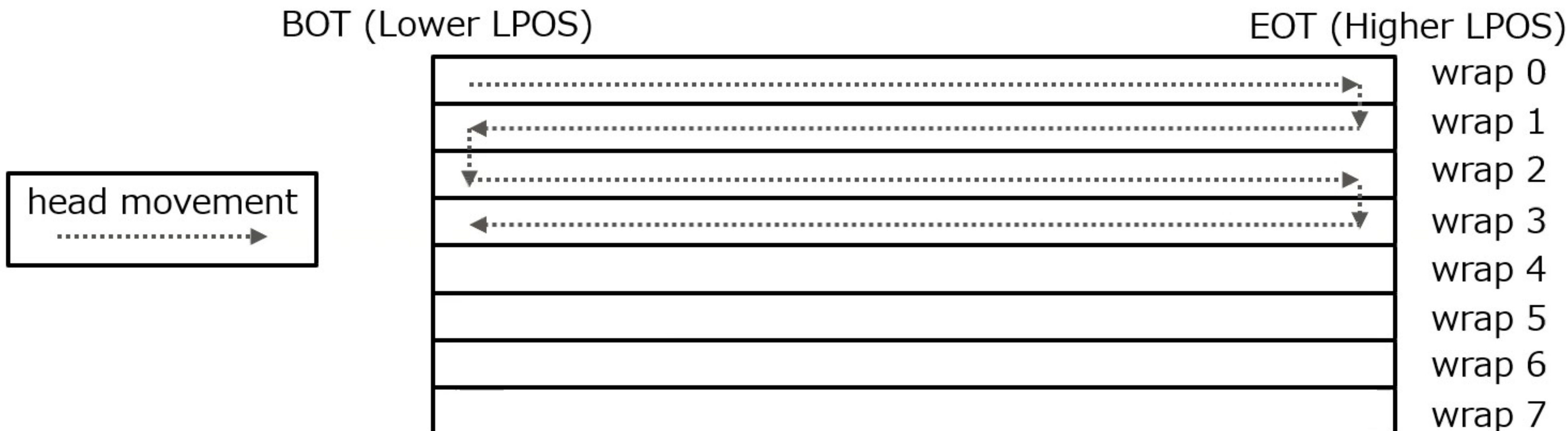
Physical Structure of Tape

- Tape cartridge consists of recording areas called wraps
 - Address within a wrap is specified using longitudinal position (LPOS)
- Tape head writes data in serpentine manner
 1. Tape head starts writing from BOT of a wrap toward EOT
 2. Tape head switches to next wrap and resumes writing toward BOT



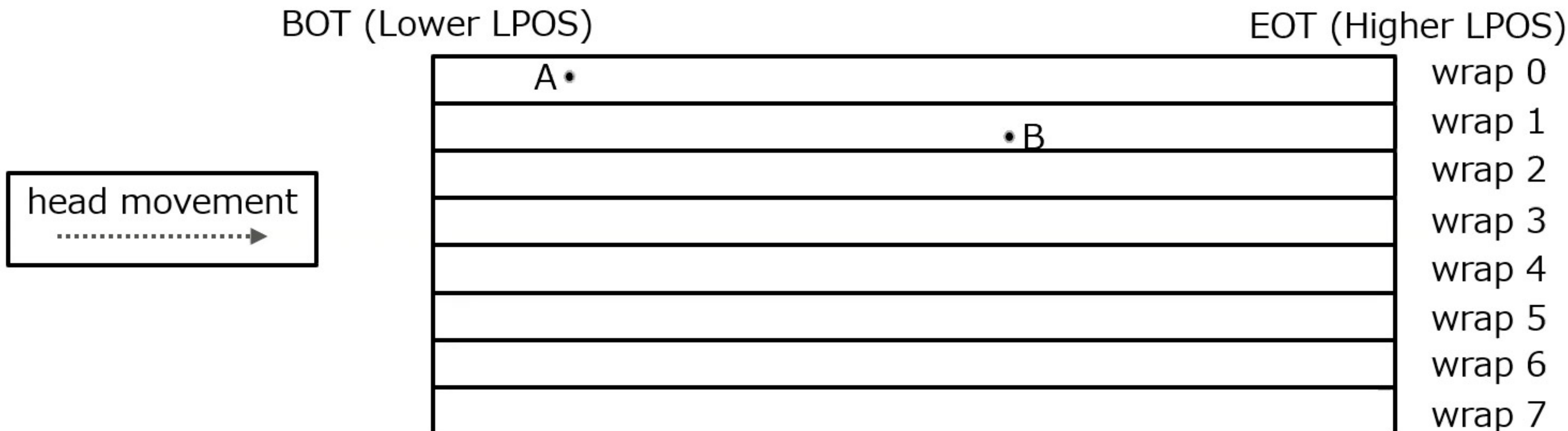
Physical Structure of Tape

- Tape cartridge consists of recording areas called wraps
 - Address within a wrap is specified using longitudinal position (LPOS)
- Tape head writes data in serpentine manner
 1. Tape head starts writing from BOT of a wrap toward EOT
 2. Tape head switches to next wrap and resumes writing toward BOT



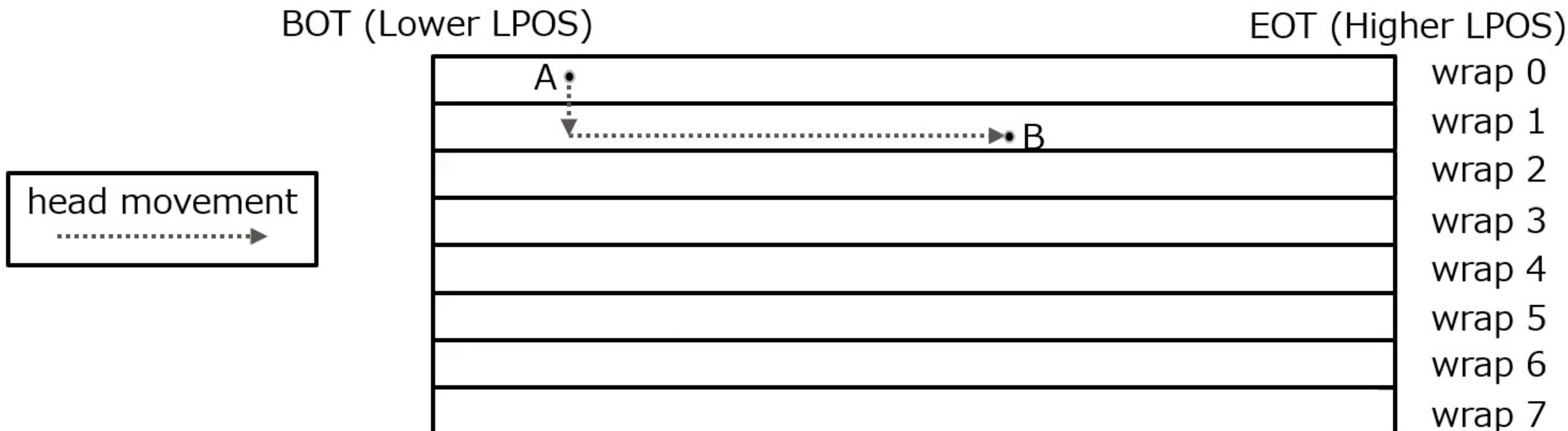
Physical Structure of Tape (cont'd)

- Switching between wraps can be used as a shortcut
 - When tape head moves from point A on wrap 0 to point B on wrap 1, tape head does not need to move to the end of wrap 0
 - Instead, it can move the shortest distance between the two points

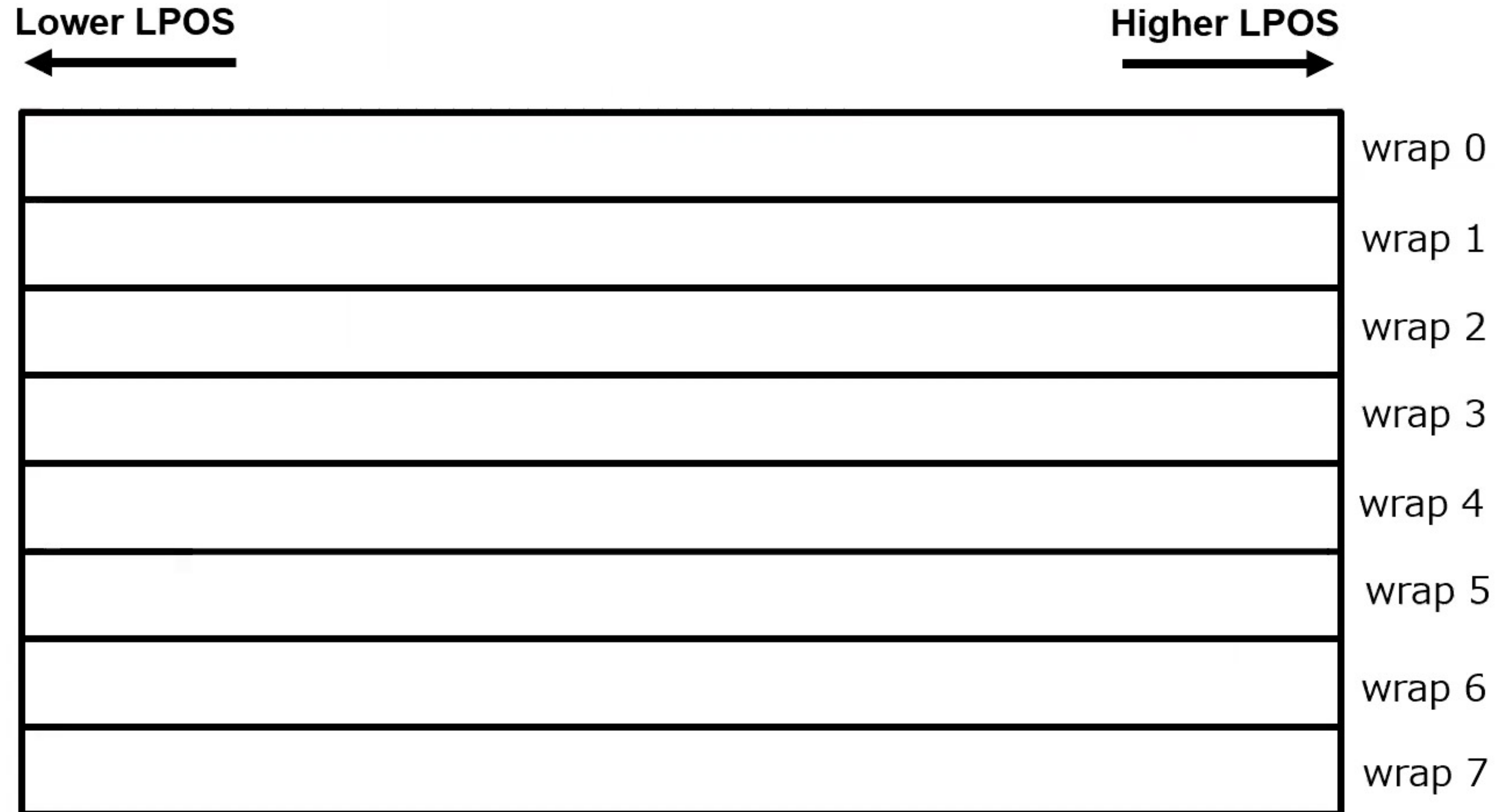


Physical Structure of Tape (cont'd)

- Switching between wraps can be used as a shortcut
 - When tape head moves from point A on wrap 0 to point B on wrap 1, tape head does not need to move to the end of wrap 0
 - Instead, it can move the shortest distance between the two points

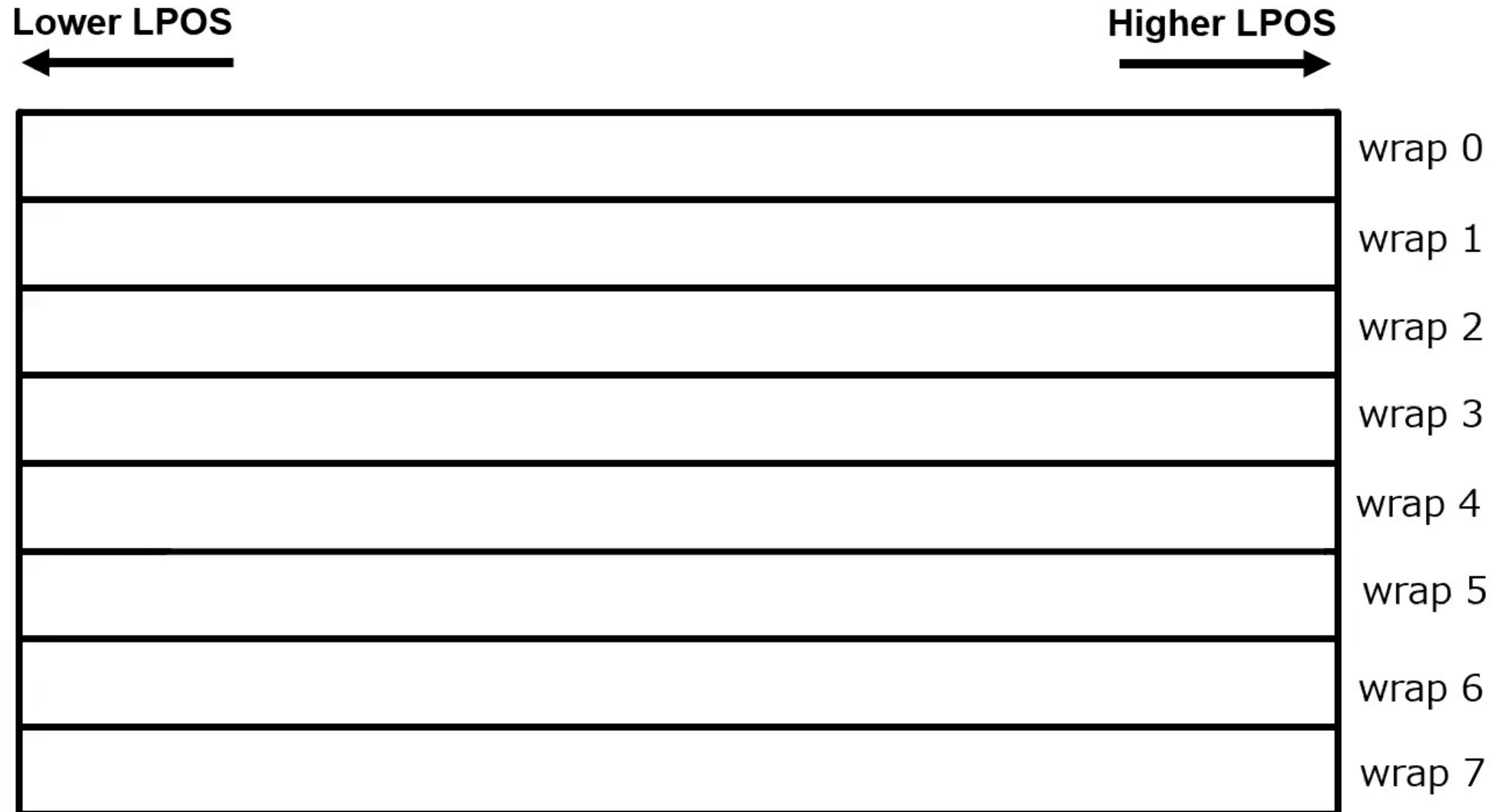


Wrap-Aware Columnar Layout (Write)



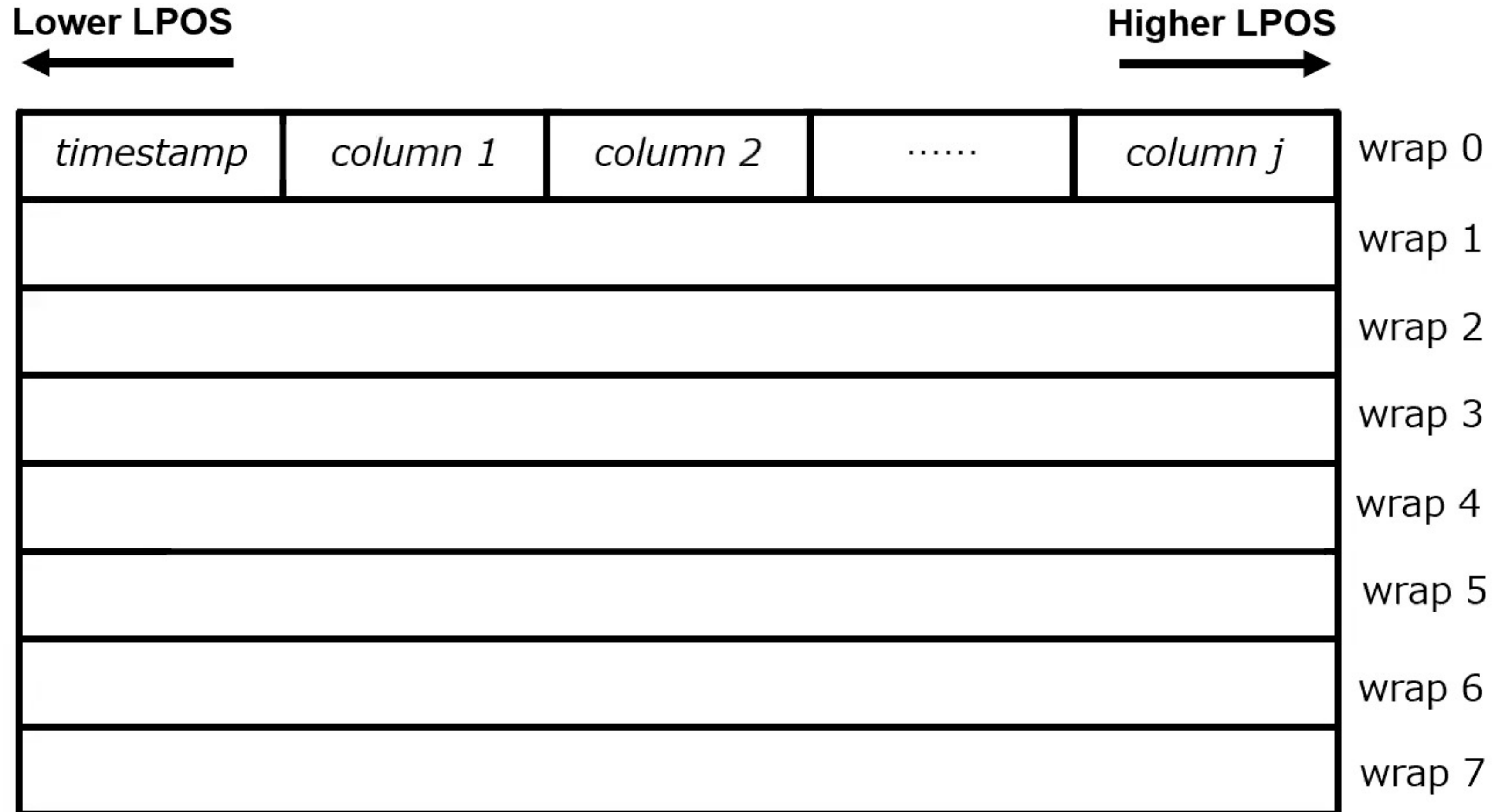
Wrap-Aware Columnar Layout (Write)

0. store stream data in a buffer of wrap size, not cartridge size



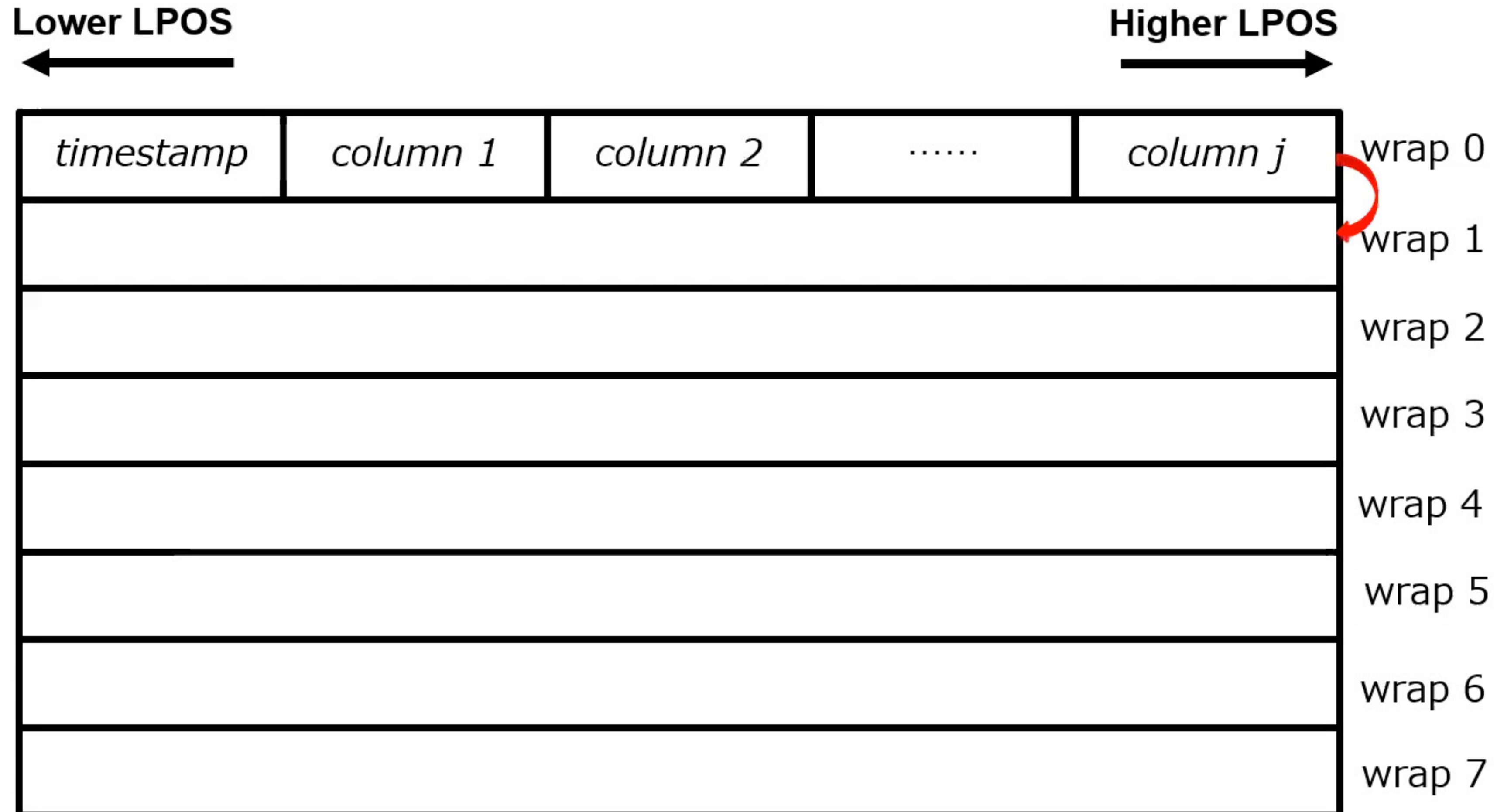
Wrap-Aware Columnar Layout (Write)

1. group data by columns, and write them as one column-oriented file



Wrap-Aware Columnar Layout (Write)

2. zero-pad and switch tape head to next wrap

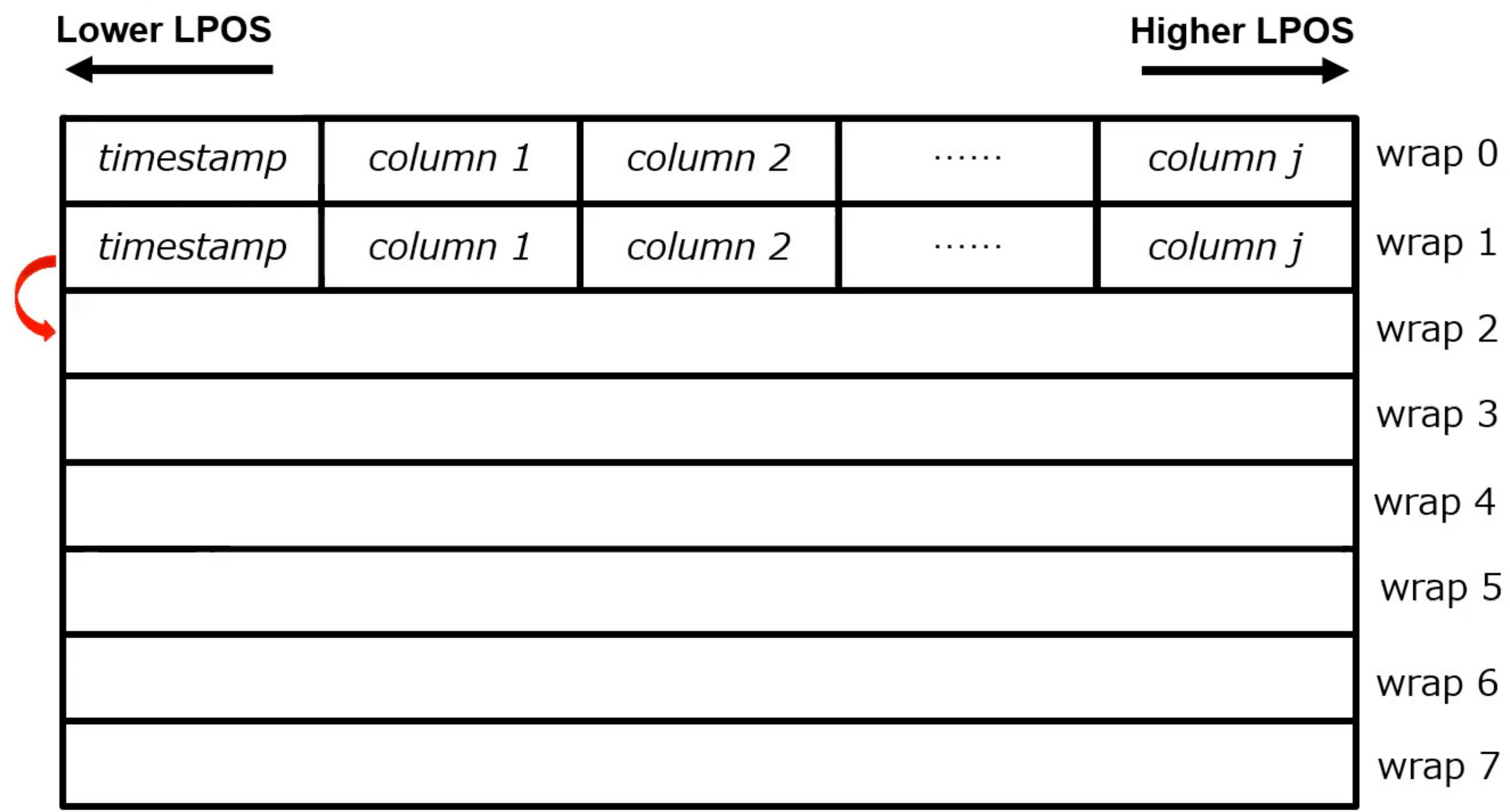


The diagram illustrates a 2D array layout with 8 rows and 5 columns. The columns are labeled from left to right as: *timestamp*, *column 1*, *column 2*, *.....*, and *column j*. The rows are labeled on the right as: wrap 0, wrap 1, wrap 2, wrap 3, wrap 4, wrap 5, wrap 6, and wrap 7. Above the array, two arrows indicate the direction of wrap-around: a left-pointing arrow labeled "Lower LPOS" and a right-pointing arrow labeled "Higher LPOS". The first two rows (wrap 0 and wrap 1) contain the labels in the columns. The subsequent rows (wrap 2 through wrap 7) are empty, representing the continuation of the array after a wrap-around.

Wrap-Aware Columnar Layout (Write)



4. zero-pad and switch tape head to next wrap



Wrap-Aware Columnar Layout (Write)

5. repeat steps 1-4 until cartridge gets full

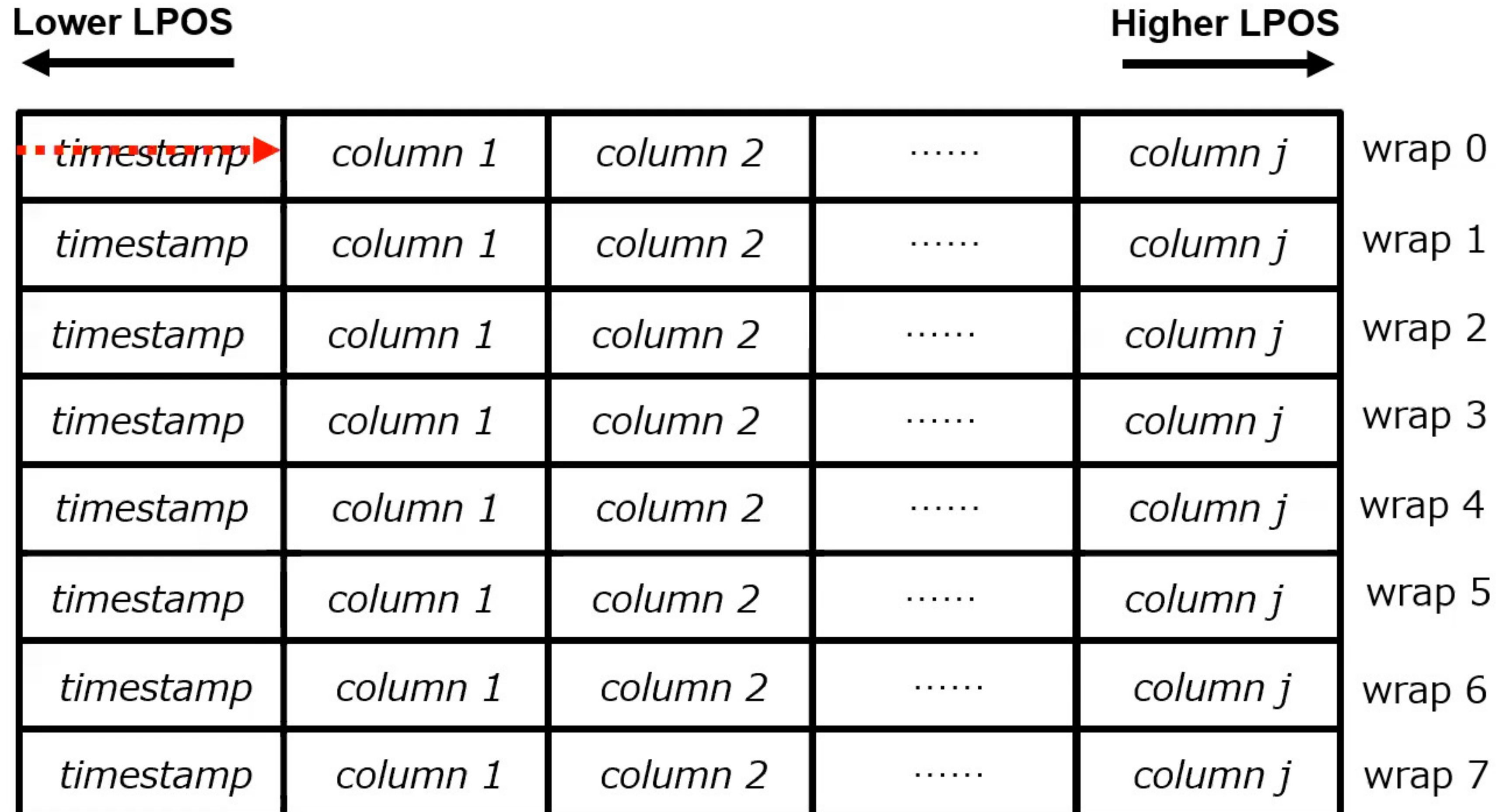
Lower LPOS		Higher LPOS			
←		→			
timestamp	column 1	column 2		column j	wrap 0
timestamp	column 1	column 2		column j	wrap 1
timestamp	column 1	column 2		column j	wrap 2
timestamp	column 1	column 2		column j	wrap 3
timestamp	column 1	column 2		column j	wrap 4
timestamp	column 1	column 2		column j	wrap 5
timestamp	column 1	column 2		column j	wrap 6
timestamp	column 1	column 2		column j	wrap 7

Wrap-Aware Columnar Layout (Read)

Lower LPOS ←					Higher LPOS →
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 0
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 1
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 2
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 3
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 4
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 5
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 6
<i>timestamp</i>	<i>column 1</i>	<i>column 2</i>	<i>.....</i>	<i>column j</i>	wrap 7

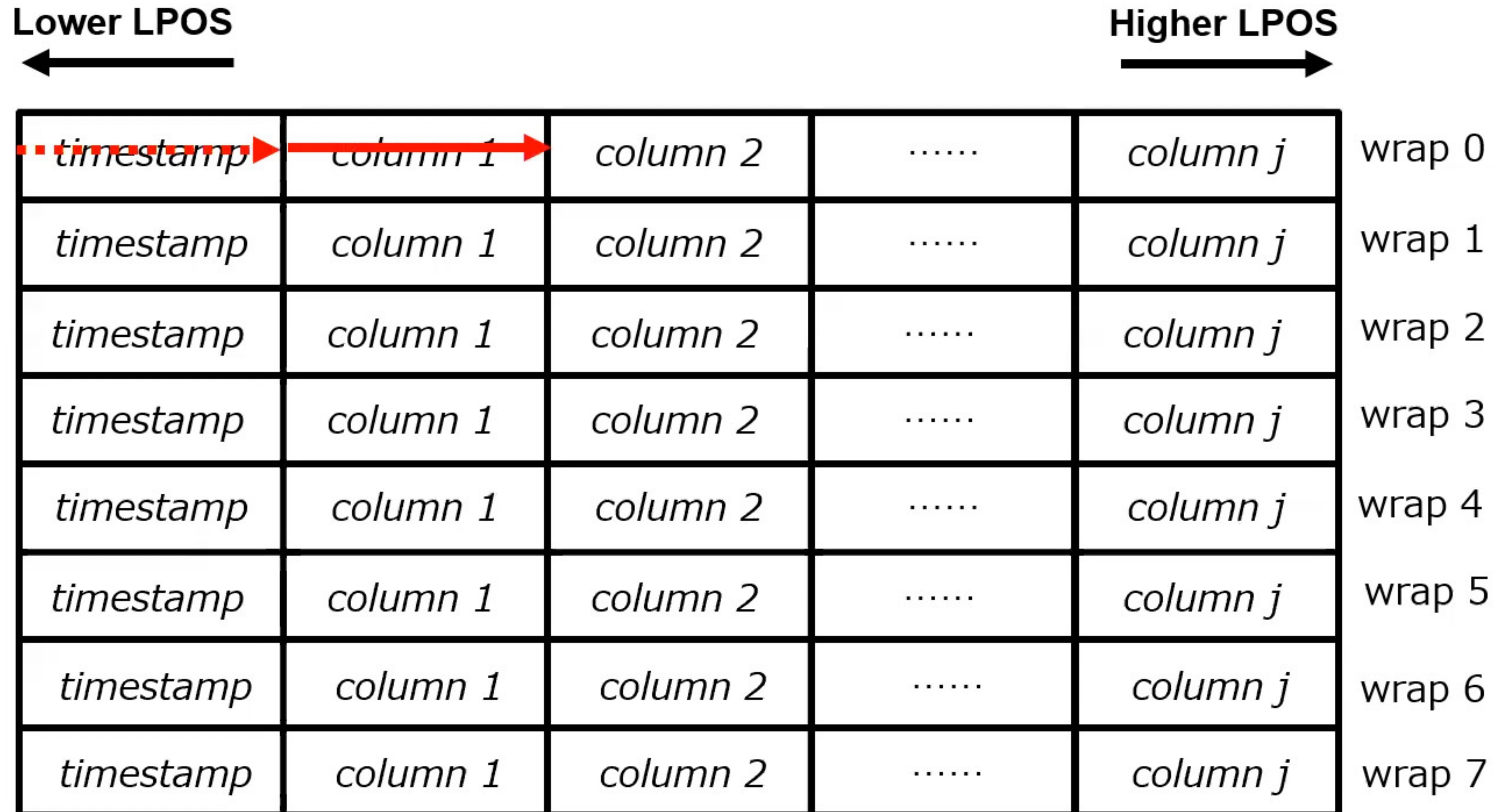
Wrap-Aware Columnar Layout (Read)

1. move head to beginning of column on first wrap



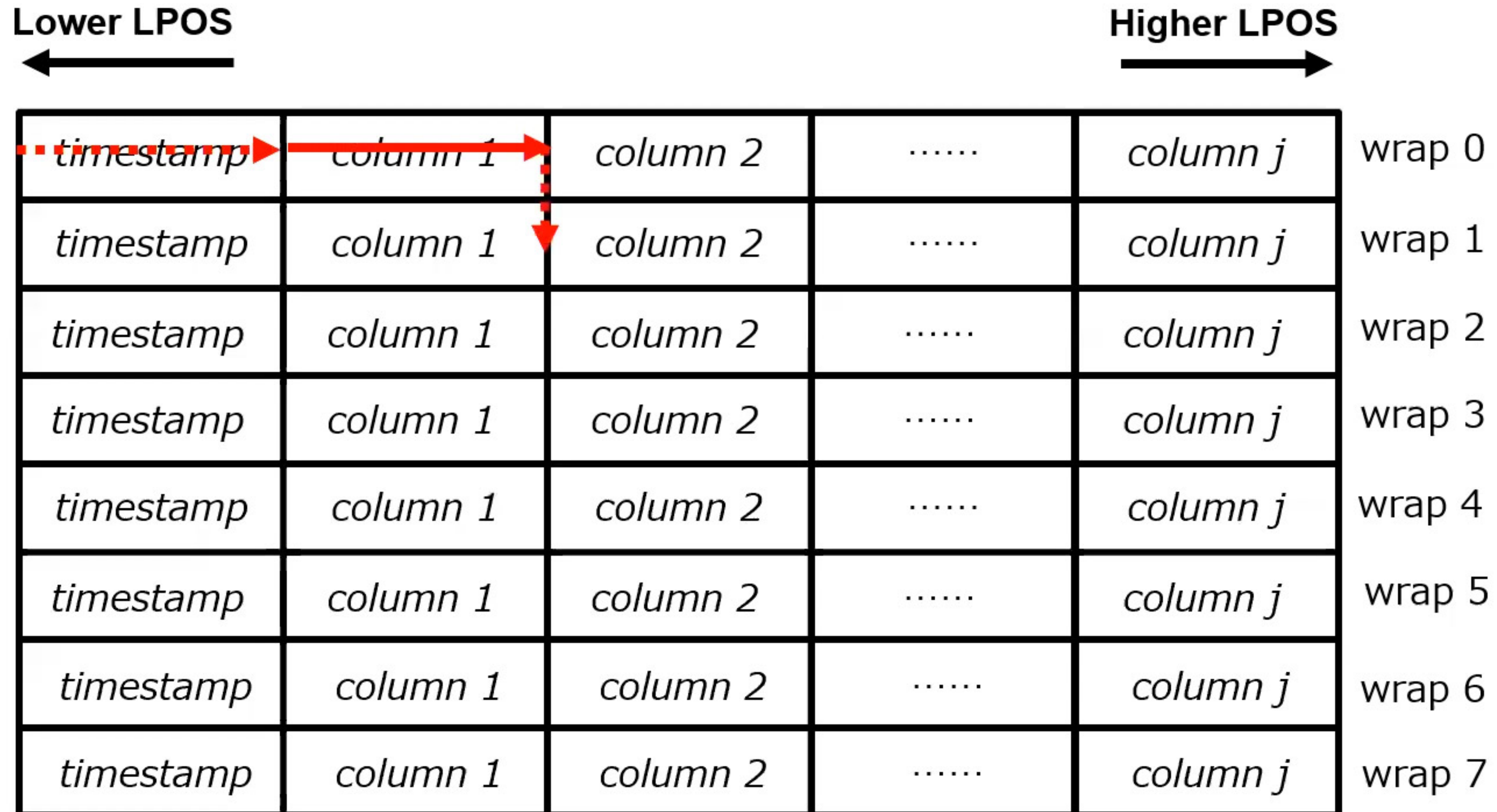
Wrap-Aware Columnar Layout (Read)

2. read column on first wrap



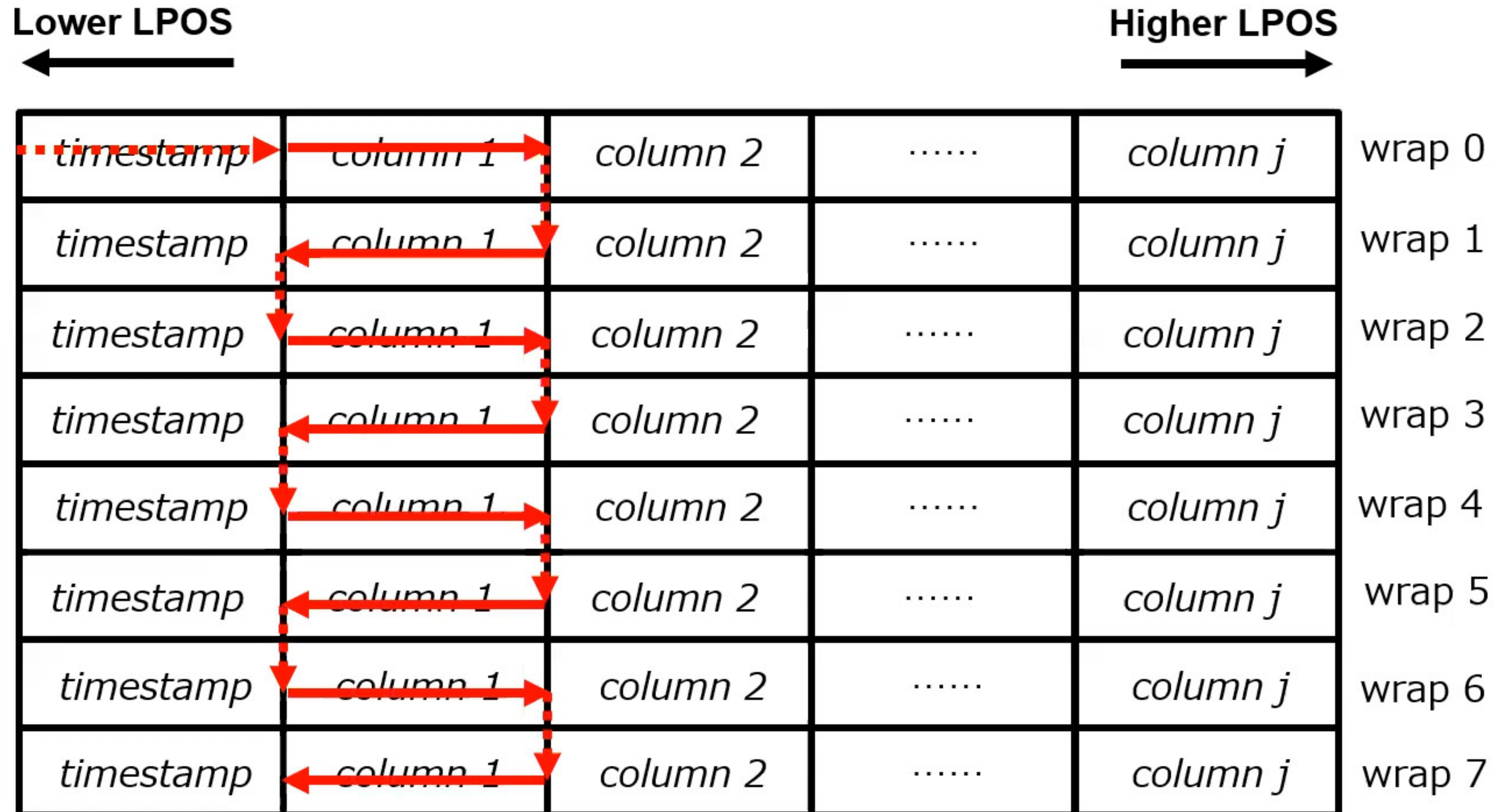
Wrap-Aware Columnar Layout (Read)

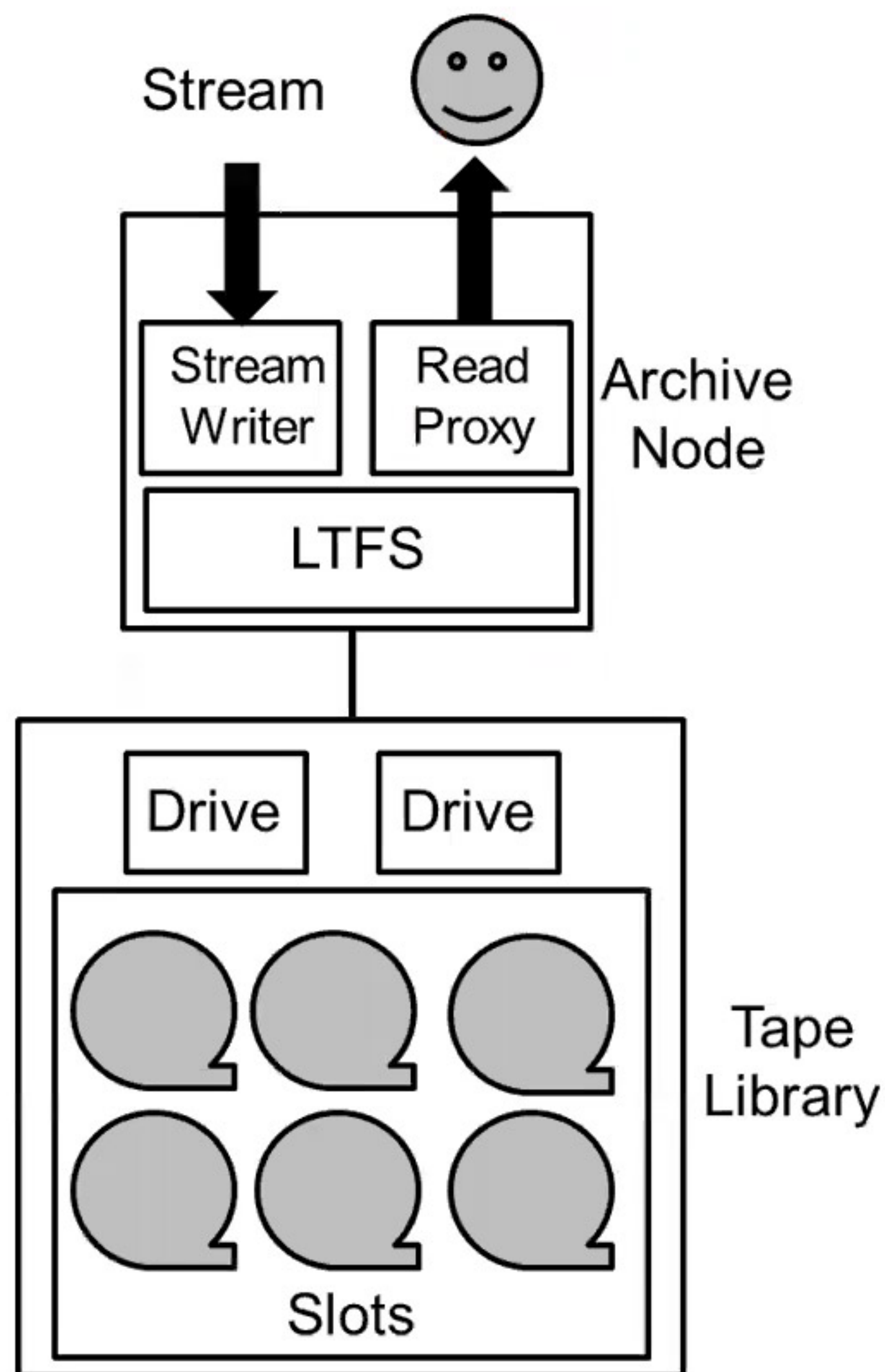
3. switch tape head to next wrap



Wrap-Aware Columnar Layout (Read)

4. repeat steps 1-3





■ Stream Writer

- buffers and converts stream data
- writes them to a cartridge in Wrap-aware Columnar Layout

■ Read Proxy

- reads and returns data from the cartridge in response to a read request from clients

■ Dataset

- CAN-bus data generated by vehicle simulator
- 966 columns including timestamp
- 3.7 billion rows
- Occupies six wraps on an LTO-7 cartridge when stored in Wrap-aware Columnar Layout

■ Layouts

- Many Files Layout (311 Parquet files of 1 GB)
- Wrap-aware Columnar Layout (WACL) (6 Parquet files of 54 GB)
- Single File Layout (1 Parquet file of 328 GB)

■ Benchmark

- Emulates I/O behavior of an application that identifies geographical locations where the car speed tends to increase by analyzing following six columns
 - VIN (vehicle identification number), timestamp, latitude, longitude, speed, acceleration

■ Measurements

- Time taken to read all the six columns and each one of the six columns was measured
 - In Many Files Layout and Single File Layout, benchmark reads data file by file
 - In Wrap-aware Columnar Layout, benchmark reads data column by column

■ Hardware

■ Archive Node

- PRIMERGY RX2540 M4 (Fujitsu)
 - Xeon Gold 6184 (Intel) 2 sockets (40 cores)
 - DRAM 192GB

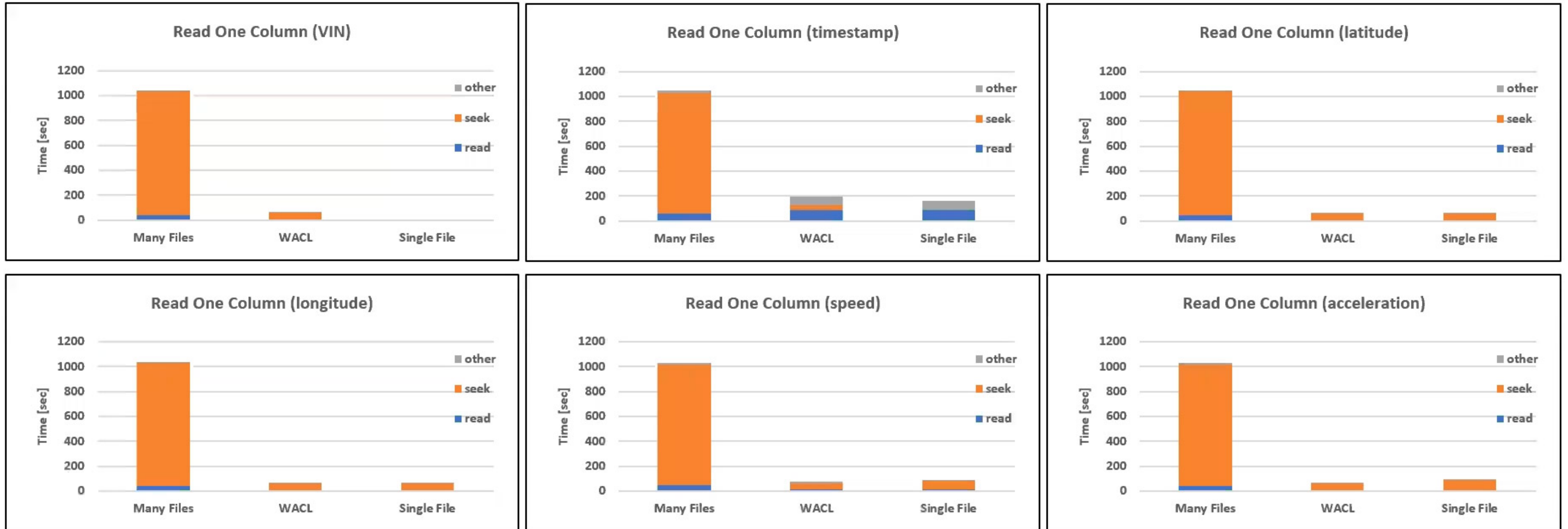
■ Tape Library

- ETERNUS LT60 S2 (Fujitsu)
 - LTO-7 (300MB/s)
 - Hardware compression disabled

■ Software

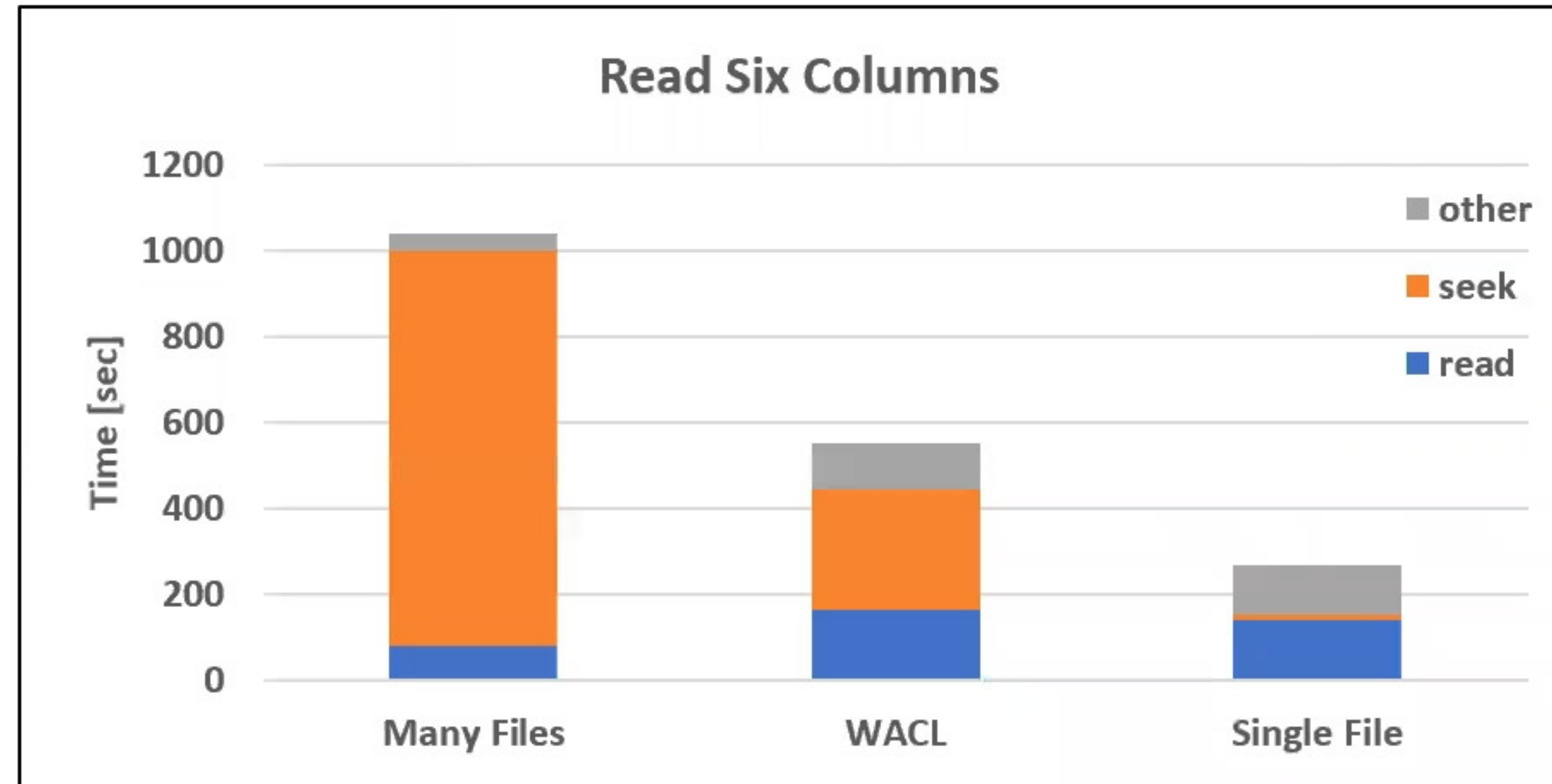
- LTFS 2.4.0/CentOS 7.6

Results: Time taken to read one column



Wrap-aware Columnar Layout outperforms Single File Layout
when reading a column at lower LPOS

Results: Time taken to read six columns



Wrap-aware Columnar Layout incurs long seek time,
but is still 53% faster than Many Files Layout

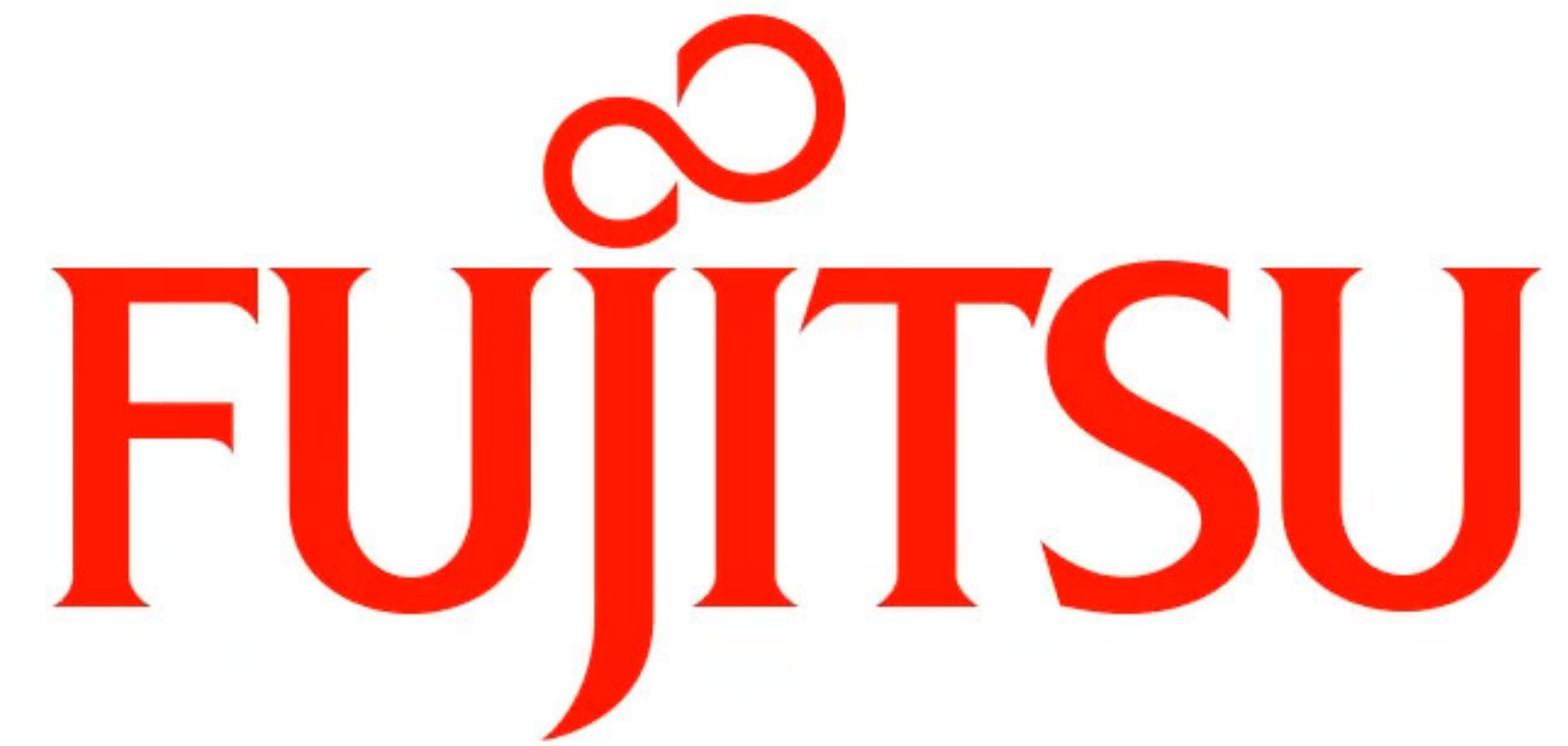
■ Wrap-aware Columnar Layout

- 53% faster than traditional wrap-unaware columnar layout

■ Future work

- optimize scheduling
 - change the order in which the wraps are read
 - read multiple columns together if they are continuous on the wrap
- optimize data layout
 - place frequently accessed columns at lower LPOS
 - arrange the columns, which tend to be accessed at the same time, continuously on a wrap

Thank you!



shaping tomorrow with you