SNIA Japan

インメモリデータ共有による異種言語混合アプリケーションの高速化

野澤 真伸[1]、今村 智史[2*]、河野 健二[1]

[1]慶應義塾大学、[2]富士通株式会社

*SNIA日本支部 次世代メモリ＆ストレージ分科会会長

自己紹介

- 所属：富士通株式会社 コンピューティング研究所
  - 2017~2021年：不揮発性メモリやストレージの研究
  - 2022年~：量子コンピューティングと量子化学計算の研究
- 専門分野：コンピュータアーキテクチャ
  - 九州大学 井上研究室にて博士号取得
  - OS（特に仮想メモリ周り）も好きです
- 社外活動：
  - 2019~2023：情報処理学会 ARC研究会幹事、ACS論文誌編集委員
  - 2023~：**SNIA-J 次世代メモリ&ストレージ分科会 会長**

本日の内容

- IEEE Big Data 2023 にて発表した慶応大河野研との共著論文の紹介
- タイトル：*Accelerating Multilingual Applications with In-memory Array Sharing*
- 著者：野澤 真伸、今村 智史、河野 健二
- ３行まとめ：
  - データ分析アプリに **Python** と **Julia** を併用すれば両者の強みを活かして高速化できる
  - ただし、両言語間でのデータやり取りオーバーヘッドが結構大きい
  - メモリ上のデータを両言語間で直接共有できるようにしてそのオーバーヘッドをほぼ０に！

Accelerating Multilingual Applications with In-memory Array Sharing

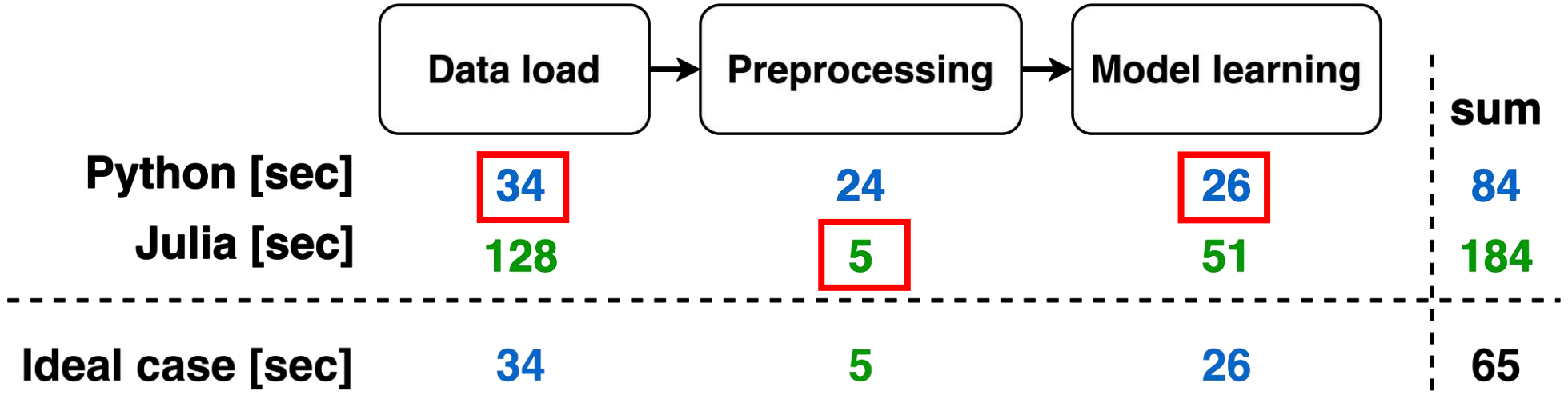
Masanobu Nozawa*, Satoshi Imamura**, Kenji Kono*

Keio University*, Fujitsu Limited**

Multilingual Data Analysis Applications

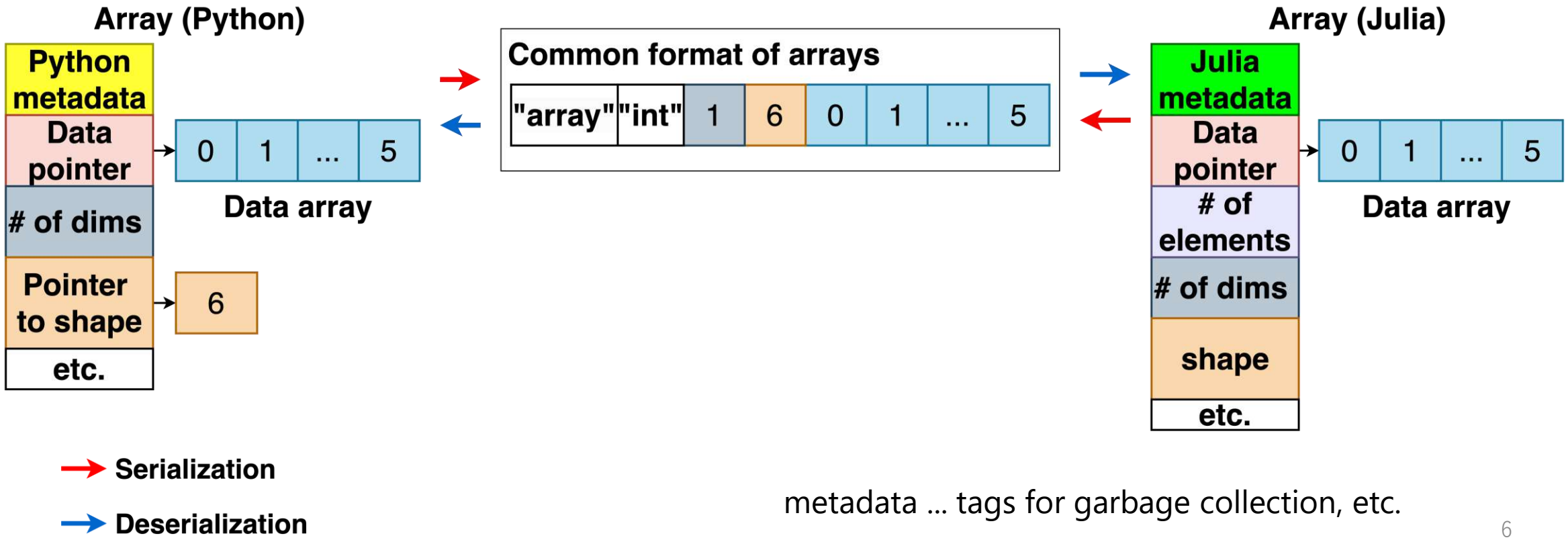
- Combine multiple programming languages for data analysis
  - Can take the advantage of each language
- Can outperform *monolingual* implementations

E.g., Execution time of Kmeans applications



Data Exchange in Multilingual Applications

- The in-memory object layout depends on each language
- **Serialization/deserialization (S/D)** is necessary for data exchange

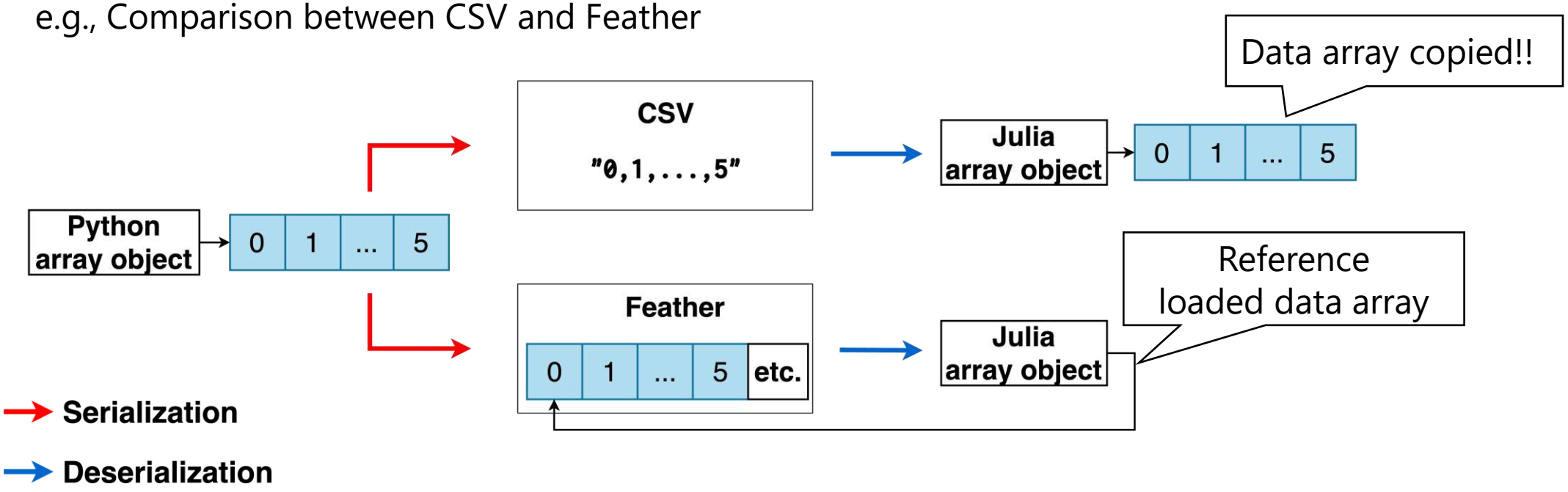


metadata ... tags for garbage collection, etc.

Feather: Fast S/D Format

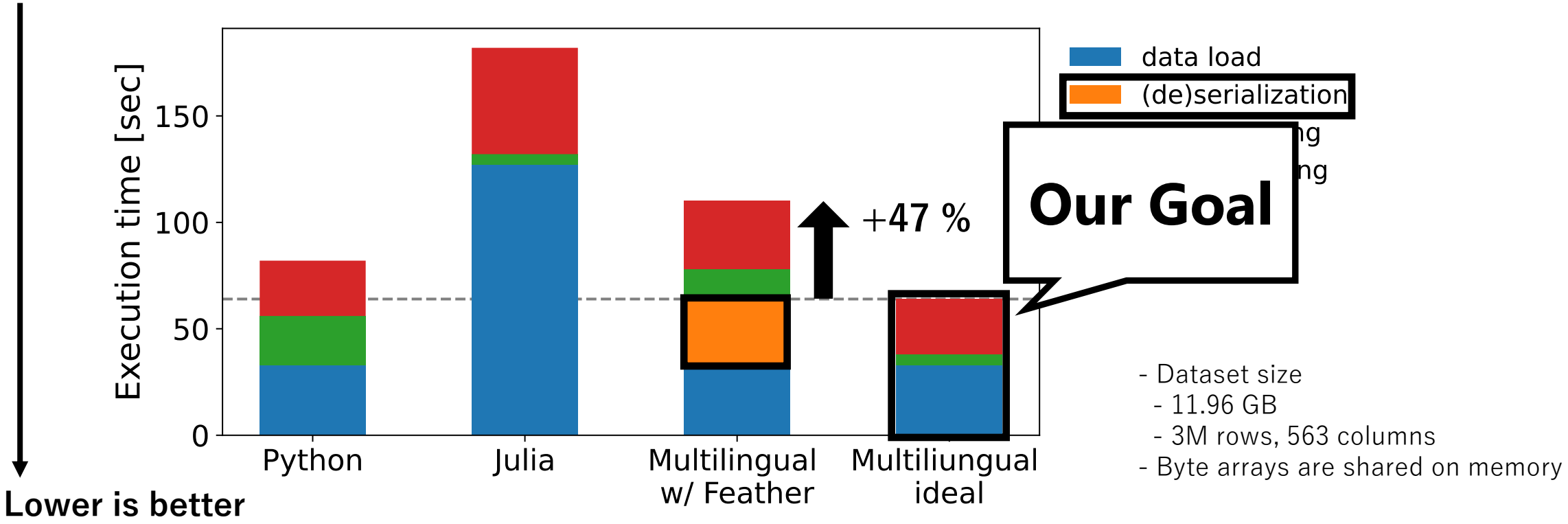
- S/D format for dataframes and tables (= collections of arrays)
- Feather holds in-memory layout of array as it is
  - ✓ **Eliminates the copy of data arrays in deserialization**
  - Serialization copies data arrays

e.g., Comparison between CSV and Feather



S/D is Time-consuming

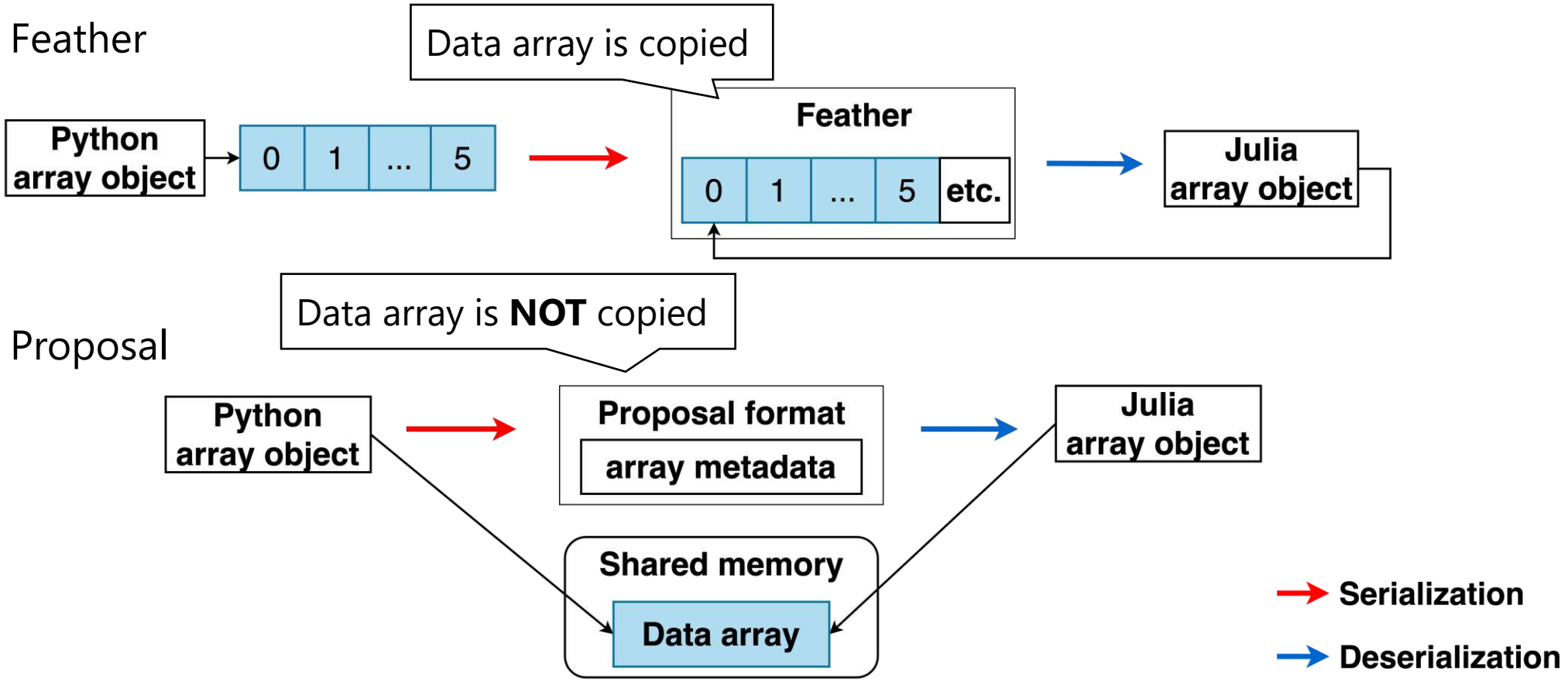
- **Array serialization** degrades the total execution time
  - Multilingual implementation can be rather slower than monolingual one!!



e.g., the execution time of the Kmeans application

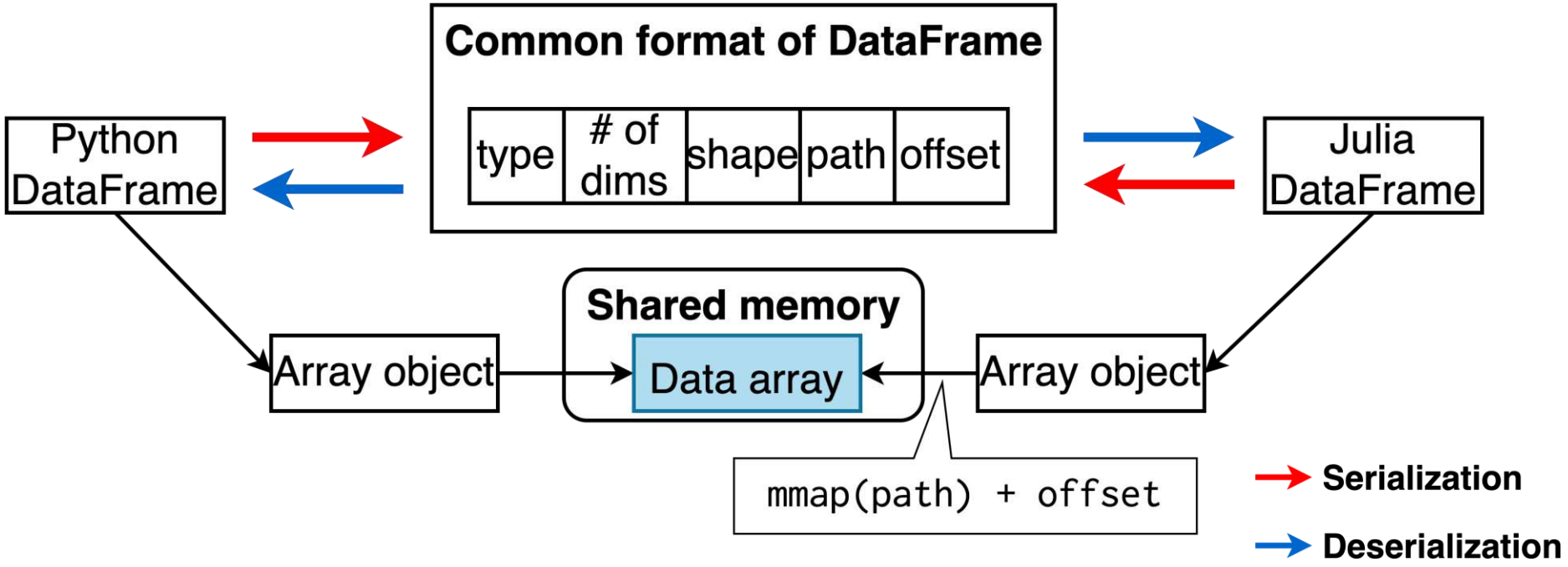
Proposal: In-Memory Array Sharing

- **Directly exchanges data arrays via shared memory**
- **Only S/D array metadata**



Implementation

- S/D modules for dataframe objects in Python and Julia
- Each array metadata contains:
  - Type, number of dimensions, and shape of the array
  - **Path** and **offset** to access shared memory



Programming Model

```
import pandas
import in_memory_array_sharing as imas # proposal module

# Load dataset from CSV
df = pandas.read_csv("/path/to/csv", engine="mas_csv")

# Serialize dataframe
imas.dump(df, "/path/to/common_format","julia")

# Deserialize dataframe
df = imas.load("/path/to/common_format")
```

Place deserialized arrays on shared memory

Specify Julia deserializes df

Evaluation

- Compare with two criteria
  - Execution time
  - Scalability to data size

Experimental setup

OS	linux-5.4.0
CPU	dual Intel® Xeon® Gold 6330
DRAM	DDR4 377 GB
CPython	ver. 3.9.7
Julia	ver. 1.6.2
Apahce Arrow	ver. 10.0.1

Targets

Python

Julia

Feather
(Python + Julia)

Proposal
(Python + Julia)

Evaluated applications
(referred from Kaggle notebooks)

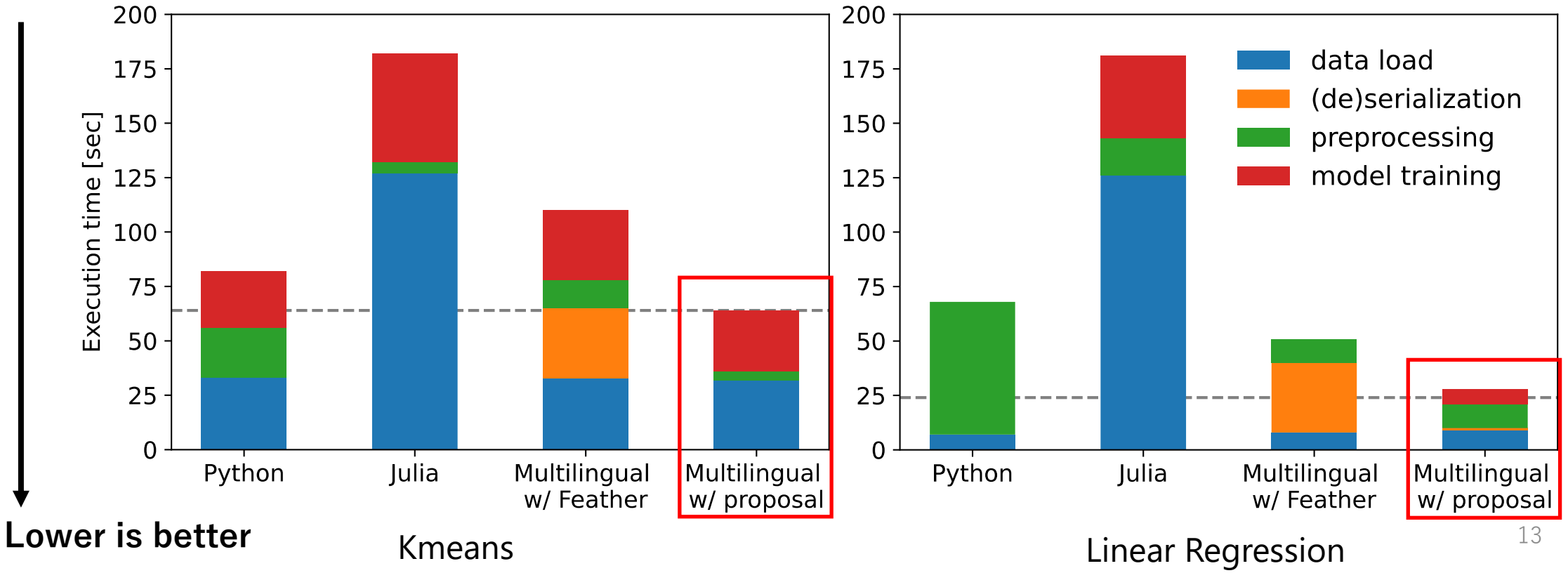
Kmeans

Linear Regression

Execution Time

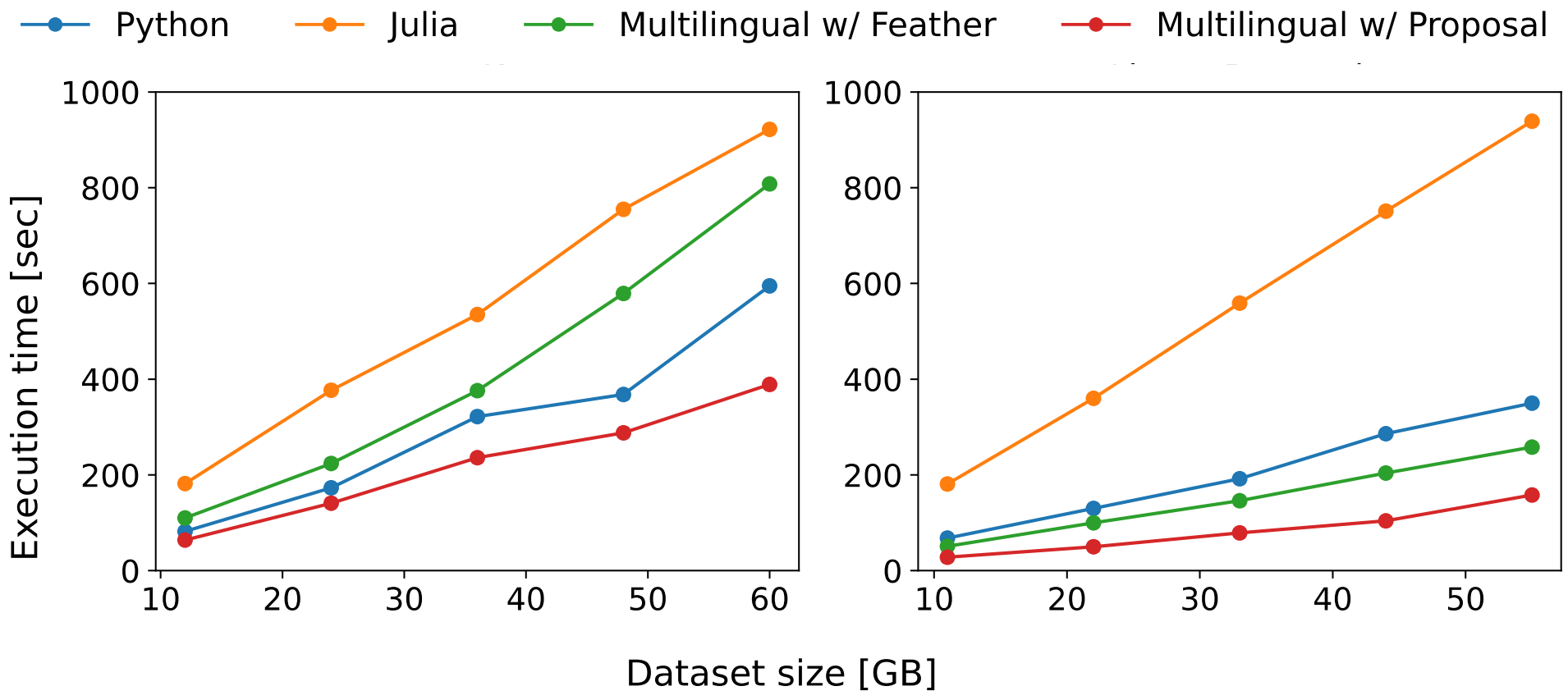
- Proposal achieves nearly **ideal performance**
  - faster than Python/Julia/Feather by up to 56.3/83.0/43.6 %

Horizontal dashed line: *ideal* execution time



Scalability to Data Size

- Proposal achieves **higher scalability** than the others



Lower is better

Kmeans

Linear Regression

Conclusion

- Multilingual applications can outperform monolingual ones
- Multilingual applications suffer from array S/D
- We propose **in-memory array sharing for Python+Julia**
  - Only serialize and deserialize array metadata
  - Copying data arrays can be eliminated
- The evaluation shows
  - **speedups** by up to 56.3/83.0/40.8 % against Python/Julia/Feather
  - **higher scalability** than the other implementations
- Our proposal would go well with Compute Express Link (CXL)
  - Exchange arrays in CXL memory with in-memory array sharing

Q&A

