Sparkling: Identification of Task Skew and Speculative Partition of Data for Spark Applications

Peilong Li (*), Xiaobing Huang (+), Tian Zhao (+),
Yan Luo (*) and Yu Cao (*)
(*) University of Massachusetts Lowell,
(+) University of Wisconsin Milwaukee

Motivations

Motivations	Improvements
Spark web UI and other instrumentation tools (Ganglia etc.) ➔ ✗ Why an application does not yield expected performance	Sparkling web tool provides: ✓ Better execution visualization ✓ In-depth statistical analysis
✗ Spark may generate unbalanced task (data/ computational skew).	Two partitioning methods: ✓ Decremented partitioning ✓ Application-aware partitioning

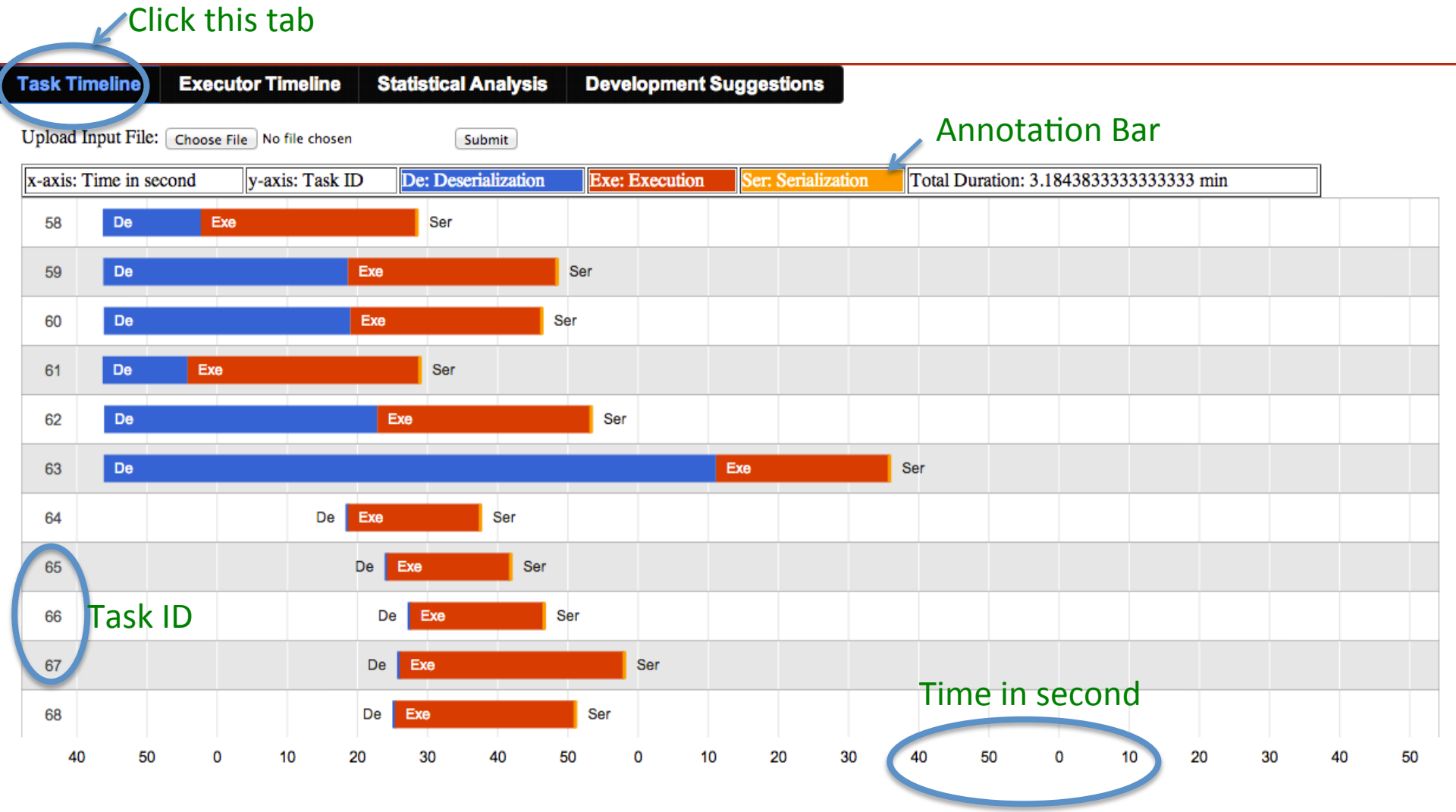
Outline

- Part I. *Sparkling* Web Tool
- Part II. Mitigate Data Skew in Spark Application

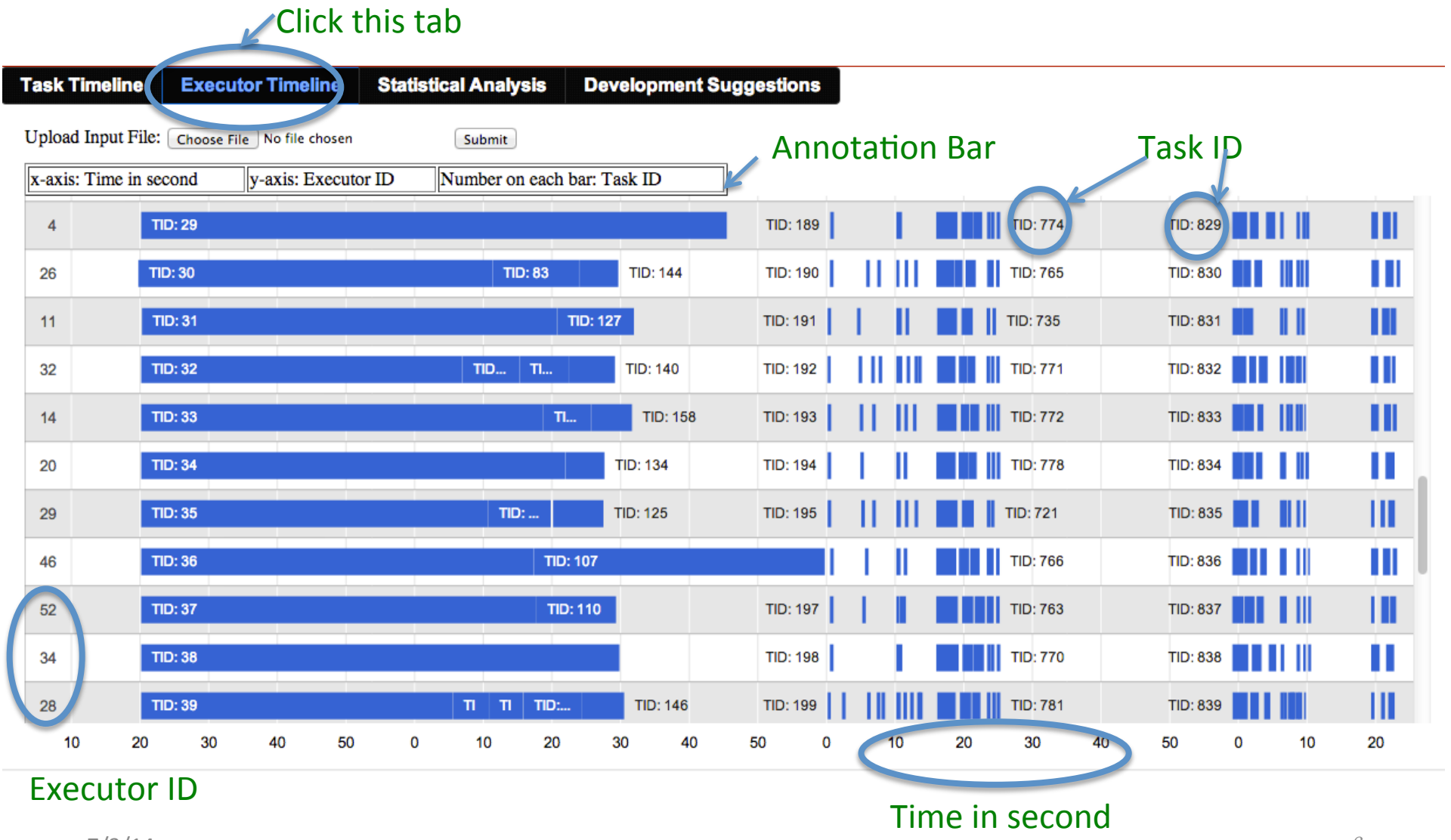
Part I. *Sparkling* Web Tool

- Sparkling enhances Spark application development:
  - More detailed metrics
  - Task view and executor view
  - Overall performance with statistics
  - Help avoid “data skew” and “small task”

Task Timeline View



Executor Timeline View

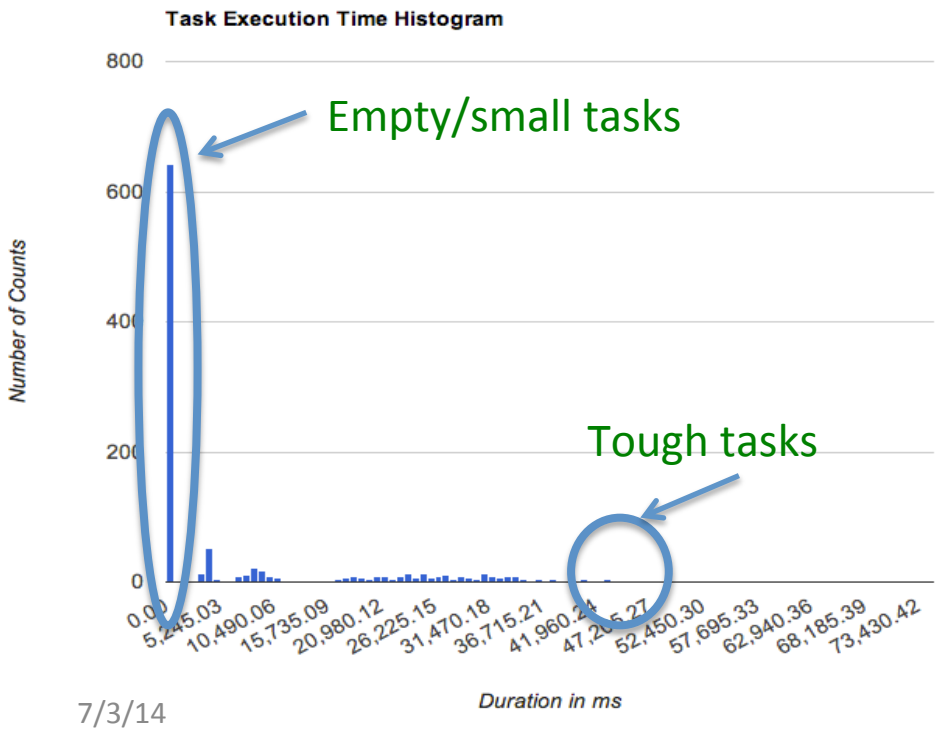


Statistical Analysis

	Item	Max Time (ms)	Min Time (ms)	Average Time (ms)	Standard Deviation (ms)
1	Deserialization Stage	45329	9	1423.2756944444445	6924.886105639271
2	Execution Stage	51332	0	1869.0159722222222	4788.835964934546
3	Serialization Stage	149	0	8.395138888888889	22.471620466036303
4	Scheduler Delay	47130	19	2199.0027777777777	7196.321615357318

	Item	Empty Task Percentage (%)
1	Empty Task (Small Task)	46.875

Diserialization Time Histogram Execution Time Histogram Serialization Time Histogram Scheduler Delay Histogram



- In contrast with Spark web UI, the table shown above provides in-depth analysis on statistical properties of the app behavior.
- Histogram use case:
  - Find a high percentage of "empty/small tasks".
  - Find tough tasks -> avoid data skew.

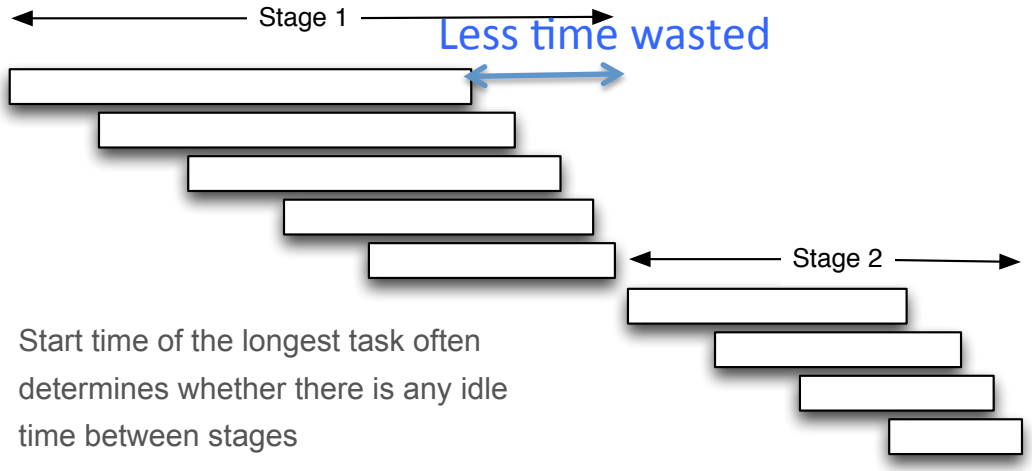
Part II. Mitigate Data Skew in Spark Application

- Spark speculative mechanism: handles stragglers
- Biomedical multimedia processing application ➔computational skew.
- Spark naïve partitioning: divide workload into slices with equal head-count.

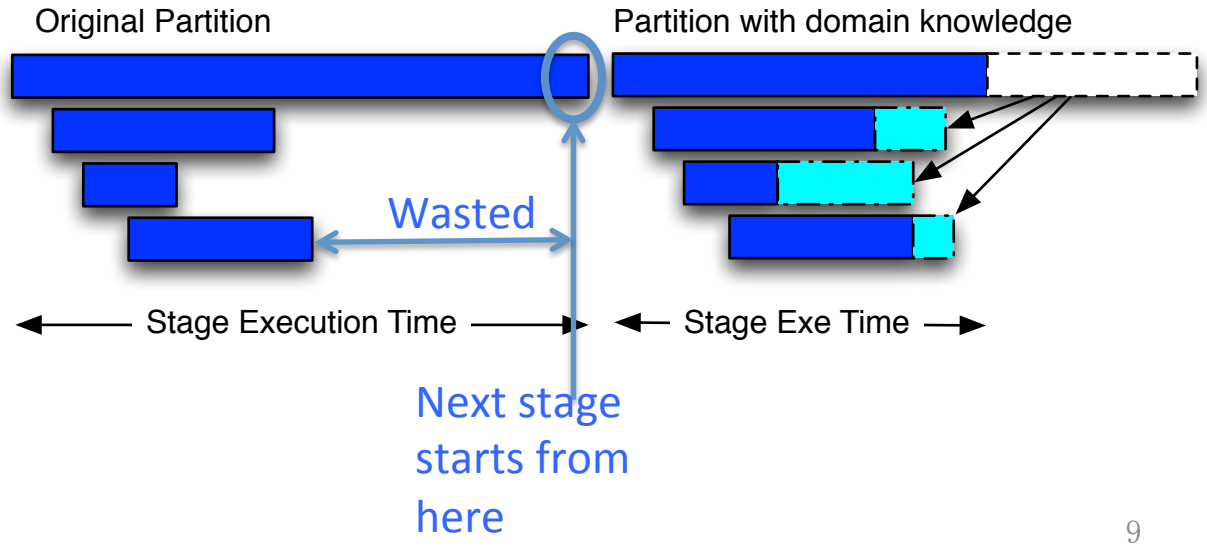
What if there are unbalanced workload on each slice?

Two Ways to Mitigate Data Skew

- Decremented workload:



- Partition RDD with domain knowledge:

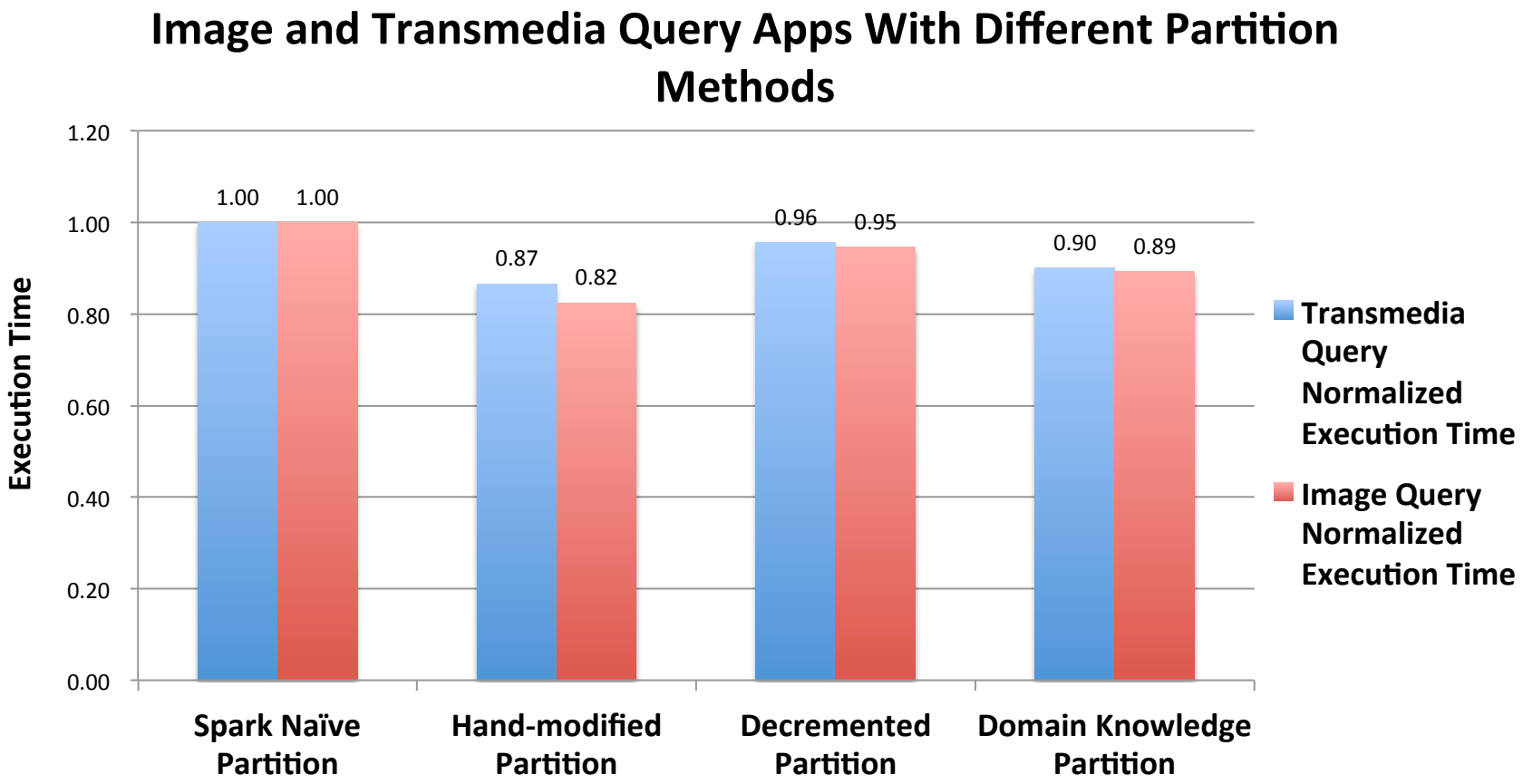


Our Test Benchmark

- PIR Benchmark [1]
  - Image Query and Transmedia query application (SIFT, LDA).
  - Dataset: Wikipedia articles (2866 multimedia documents: images + texts)[2]
- Domain knowledge
  - Edge Pixel Percentage (EPP) = # of Edge Pixels / Total # of Pixels
- Why use EPP
  - Image/Transmedia query app use local feature.
  - EPP is a good stand for local feature complexity.

Reference:
[1] Xiaobing Huang; Tian Zhao; Yu Cao, "PIR: A Domain Specific Language for Multimedia Retrieval," Multimedia (ISM), 2013 IEEE International Symposium on , vol., no., pp.359,363, 9-11 Dec. 2013
[2] UCSD Statistical Visual Computing Lab. http://www.svcl.ucsd.edu/projects/crossmodal/

Experiment Results



Conclusion

- Sparkling web tool provides more insight for developers on how to improve app performance by decreasing data skew and small tasks.
- Our proposed automatic data skew mitigation algorithm shows a 11% increase of performance on our biomedical benchmarks.