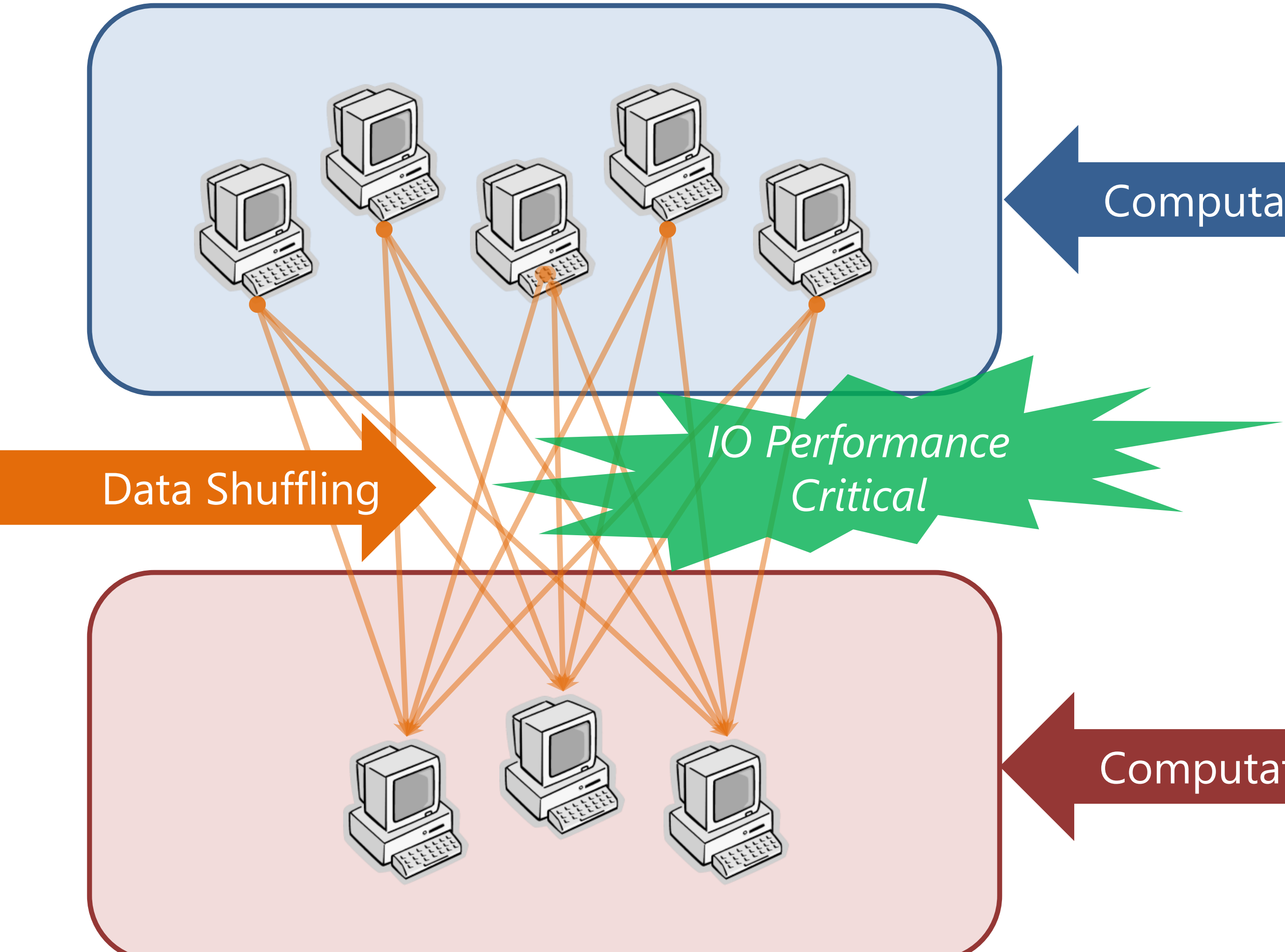
# Spotting Code Optimizations in Data-Parallel Pipelines through **PeriSCOPE**

Zhenyu Guo, Xuepeng Fan, Rishan Chen, Jiaxing Zhang, Hucheng Zhou,
Sean McDirmid, Chang Liu, Wei Lin*, Jingren Zhou*, Lidong Zhou

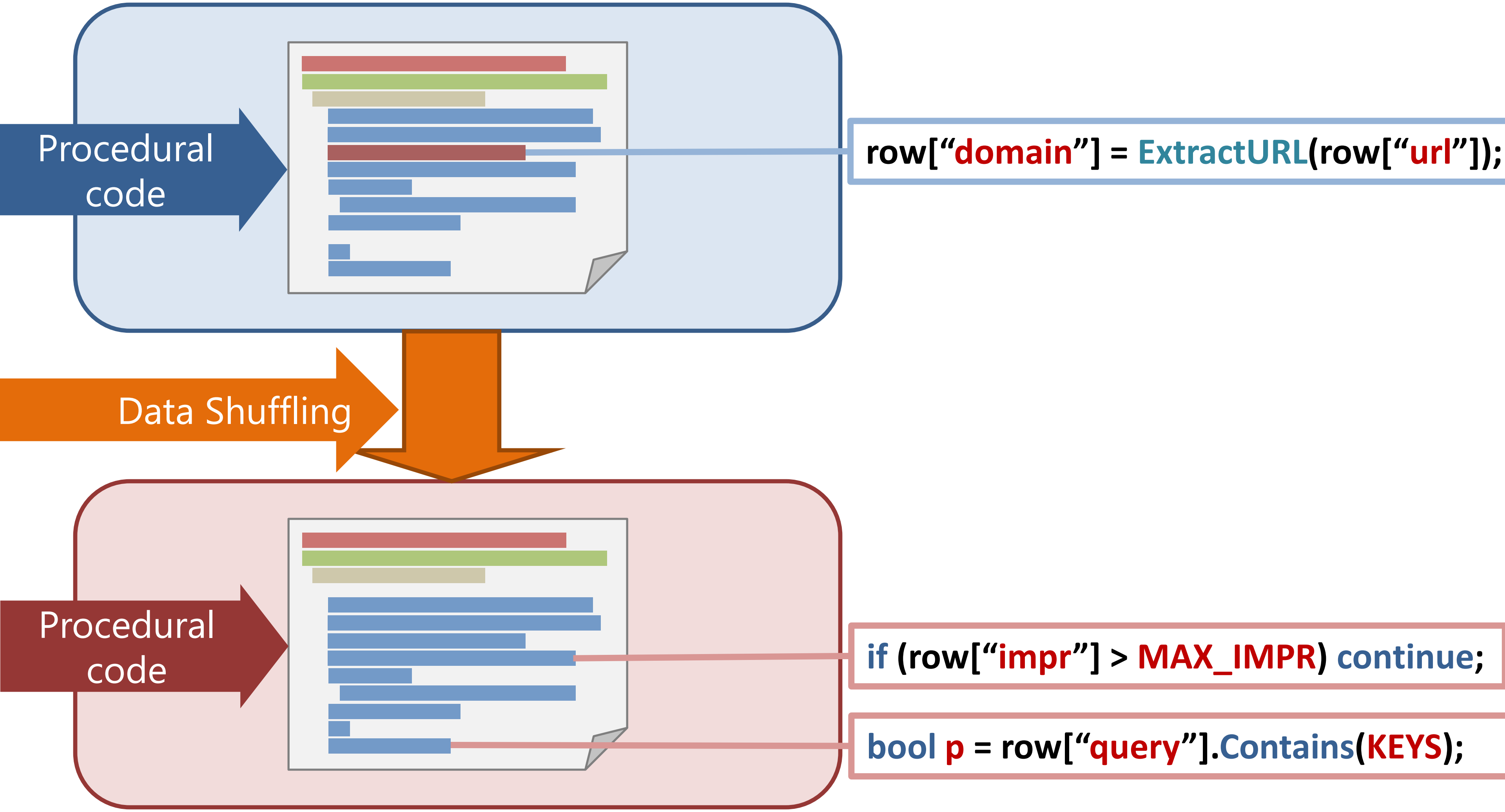
Microsoft Research Asia

*Microsoft BING

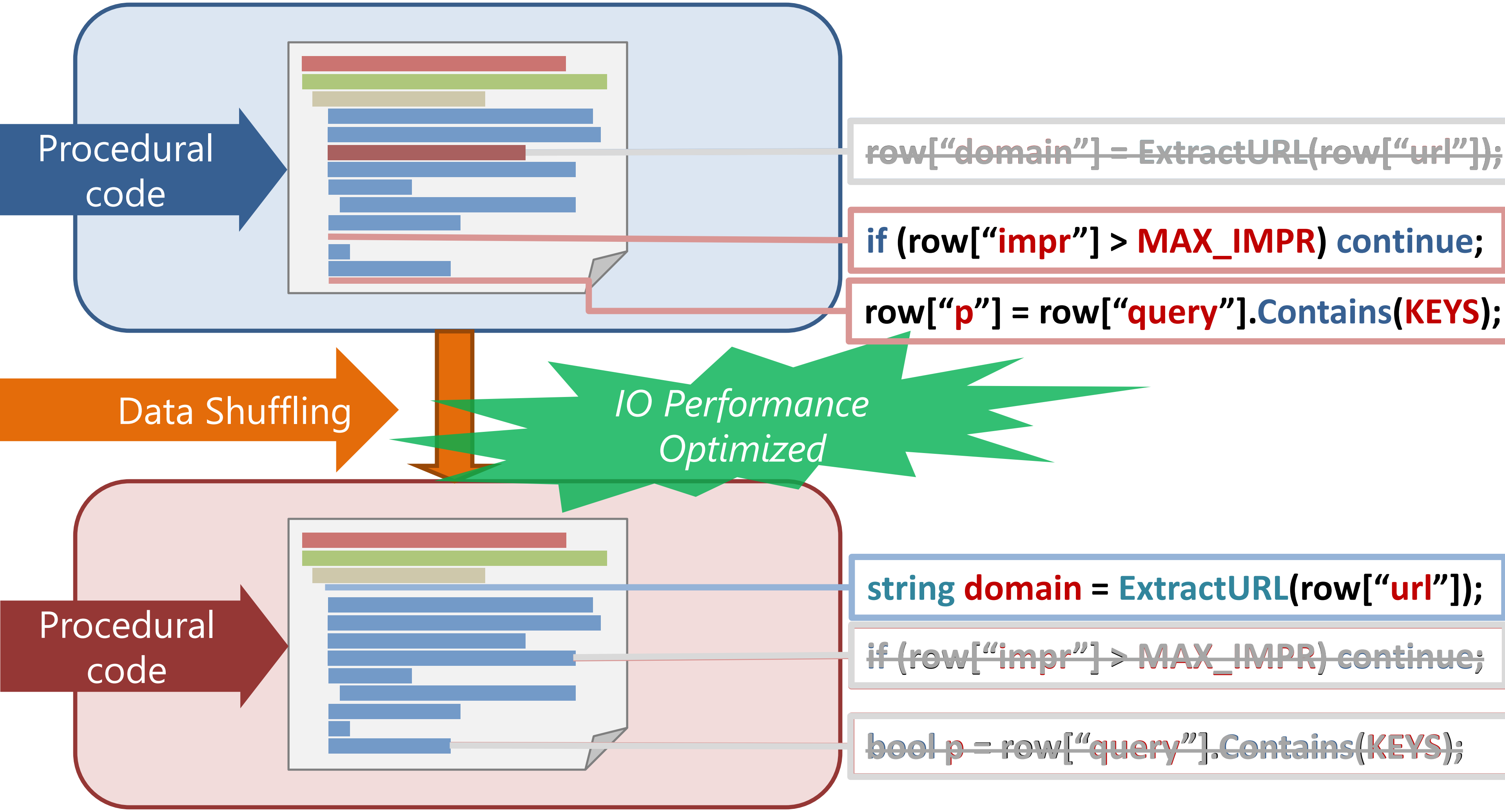
# Distributed Data-Parallel Pipelines



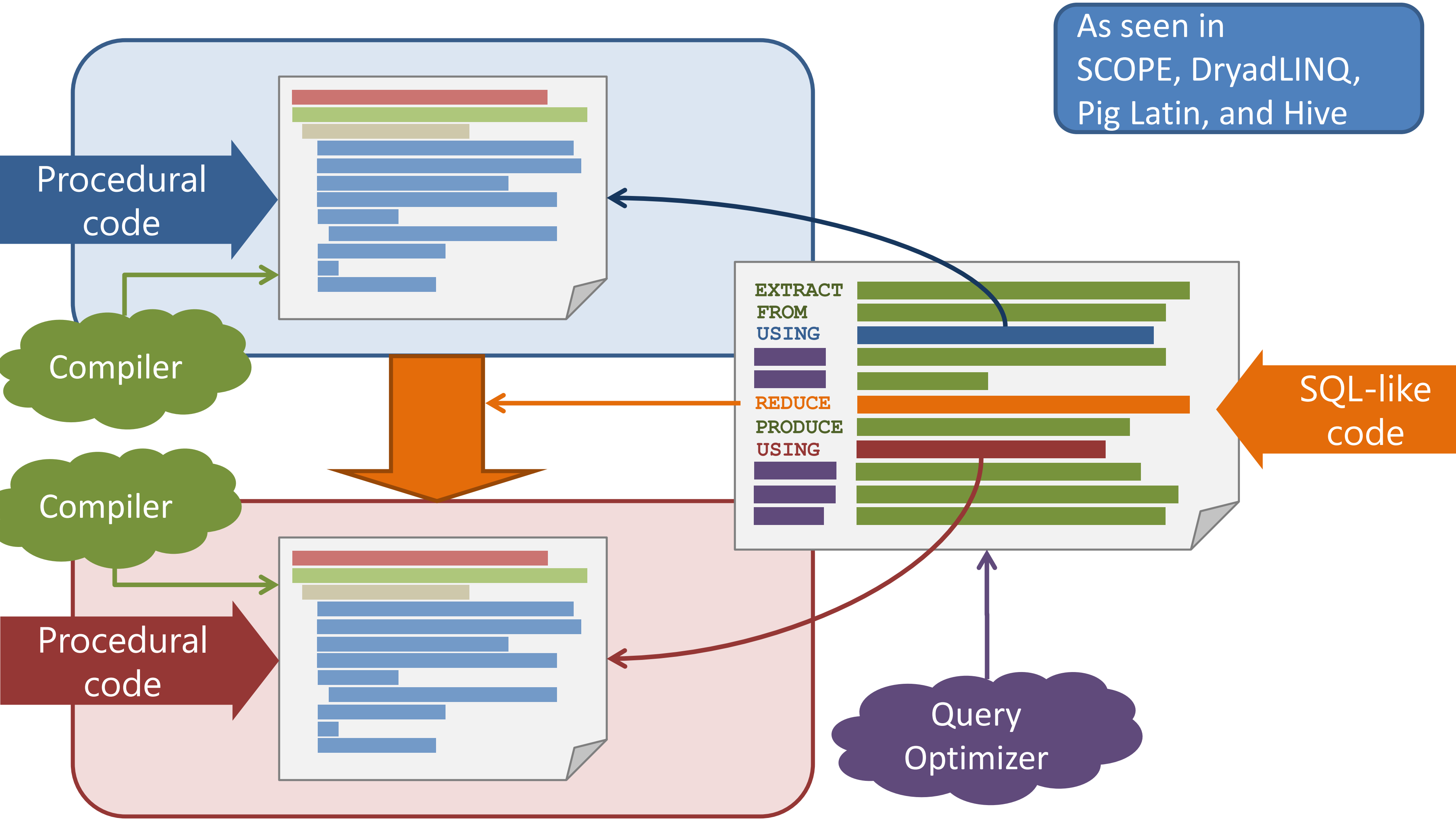
# Opportunities



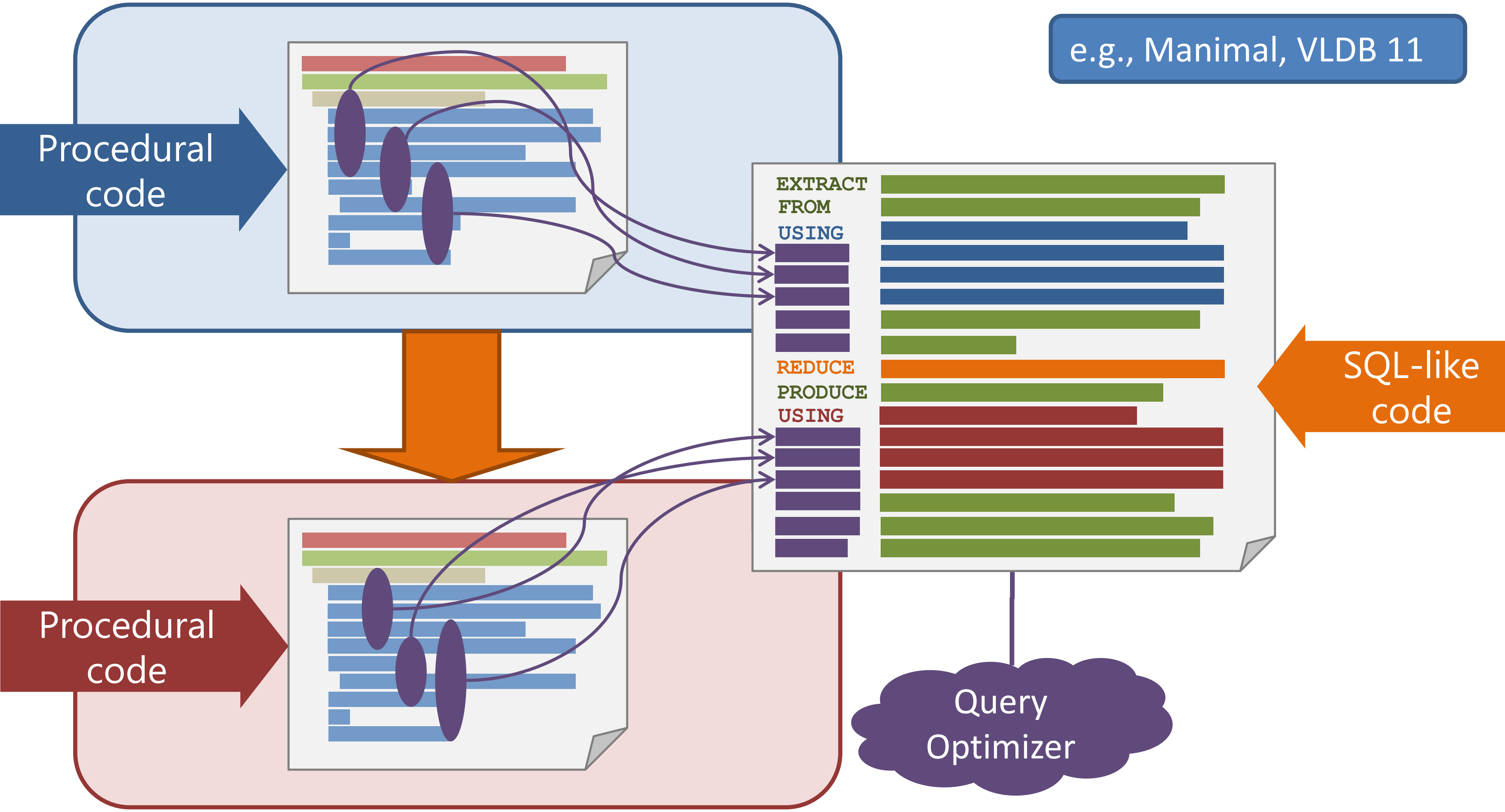
# Opportunities



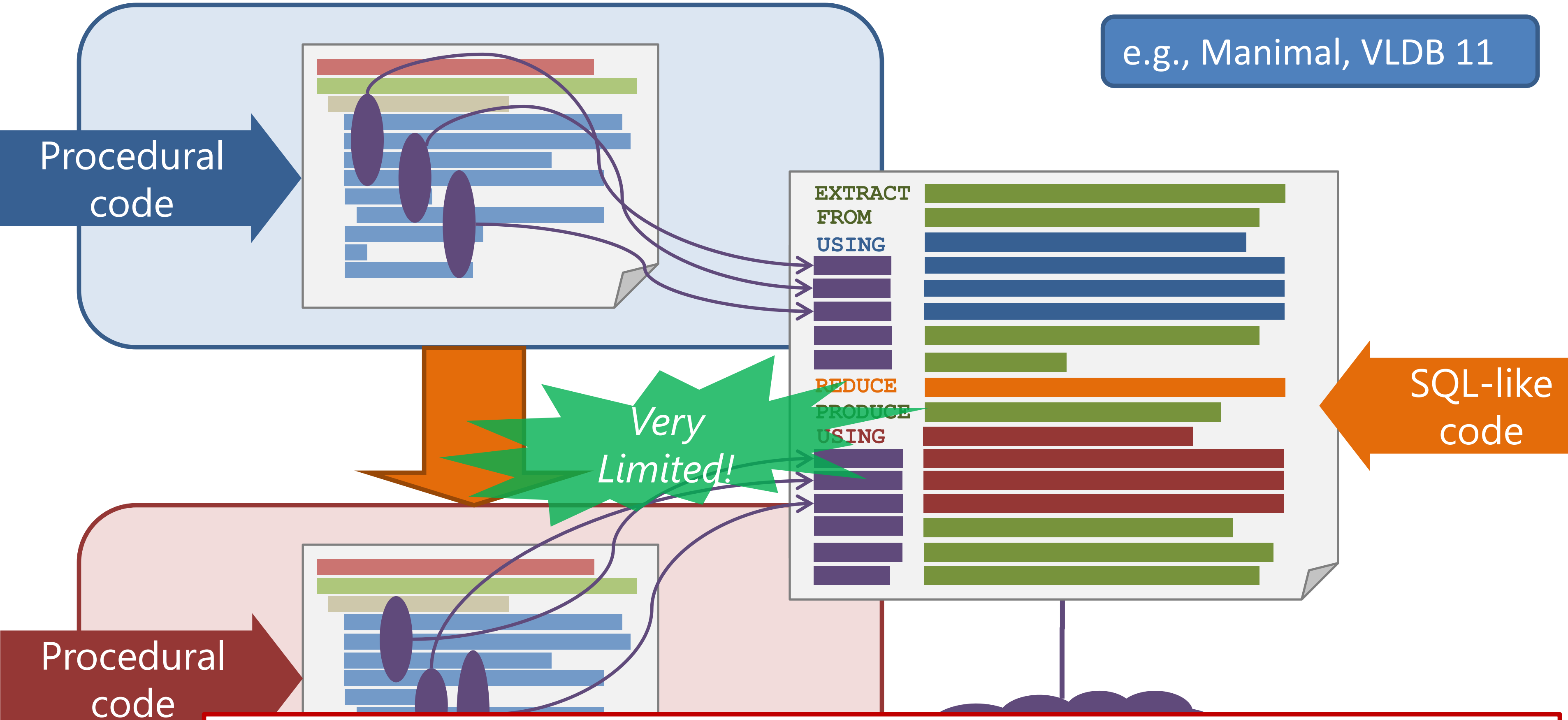
# Current Practice: Separated Optimization



# Holistic optimization using query optimizer



# Holistic optimization using query optimizer



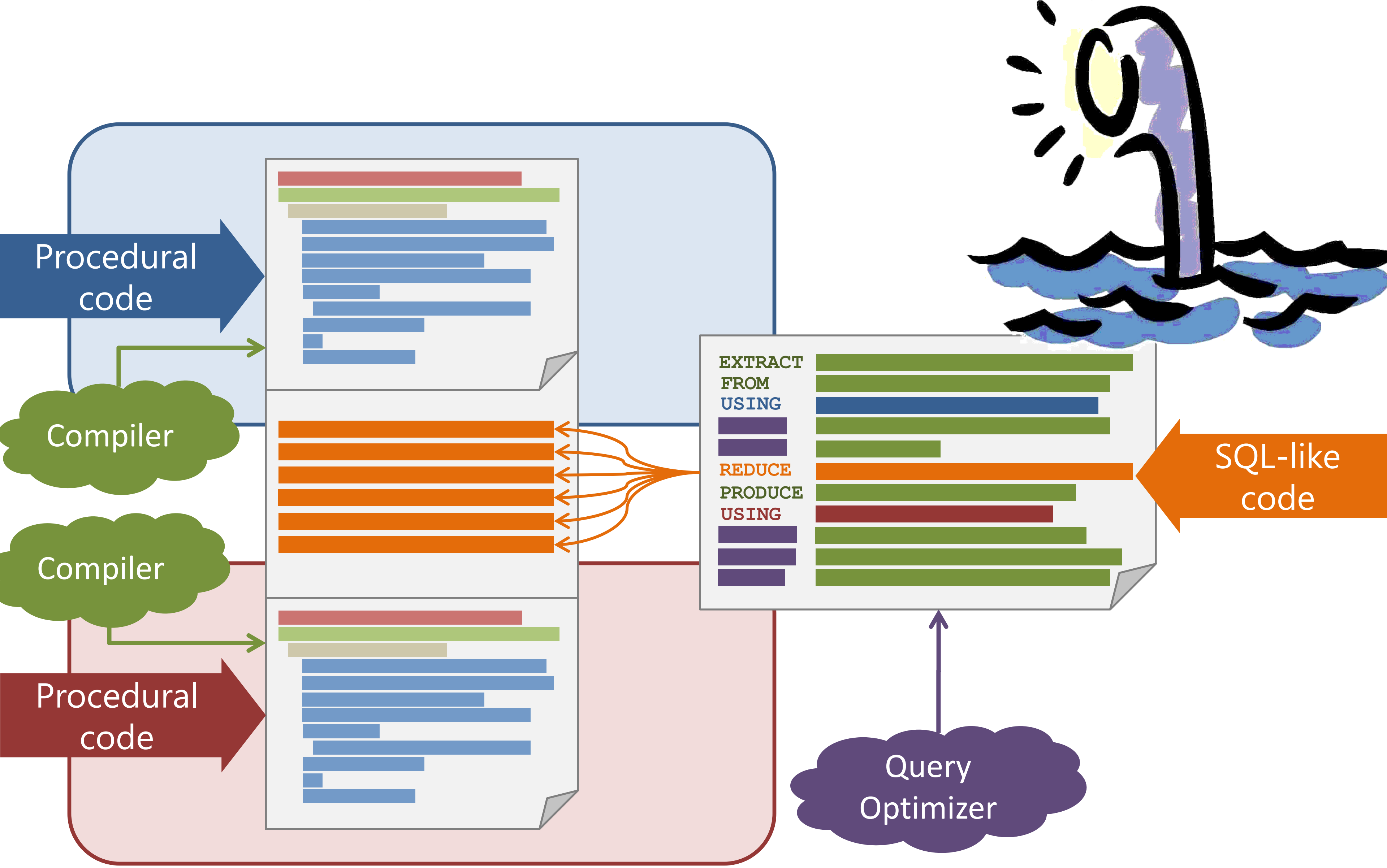

if (row[“impr”] > MAX_IMPR) continue;

=> WHERE !(impr > MAX_IMPR)

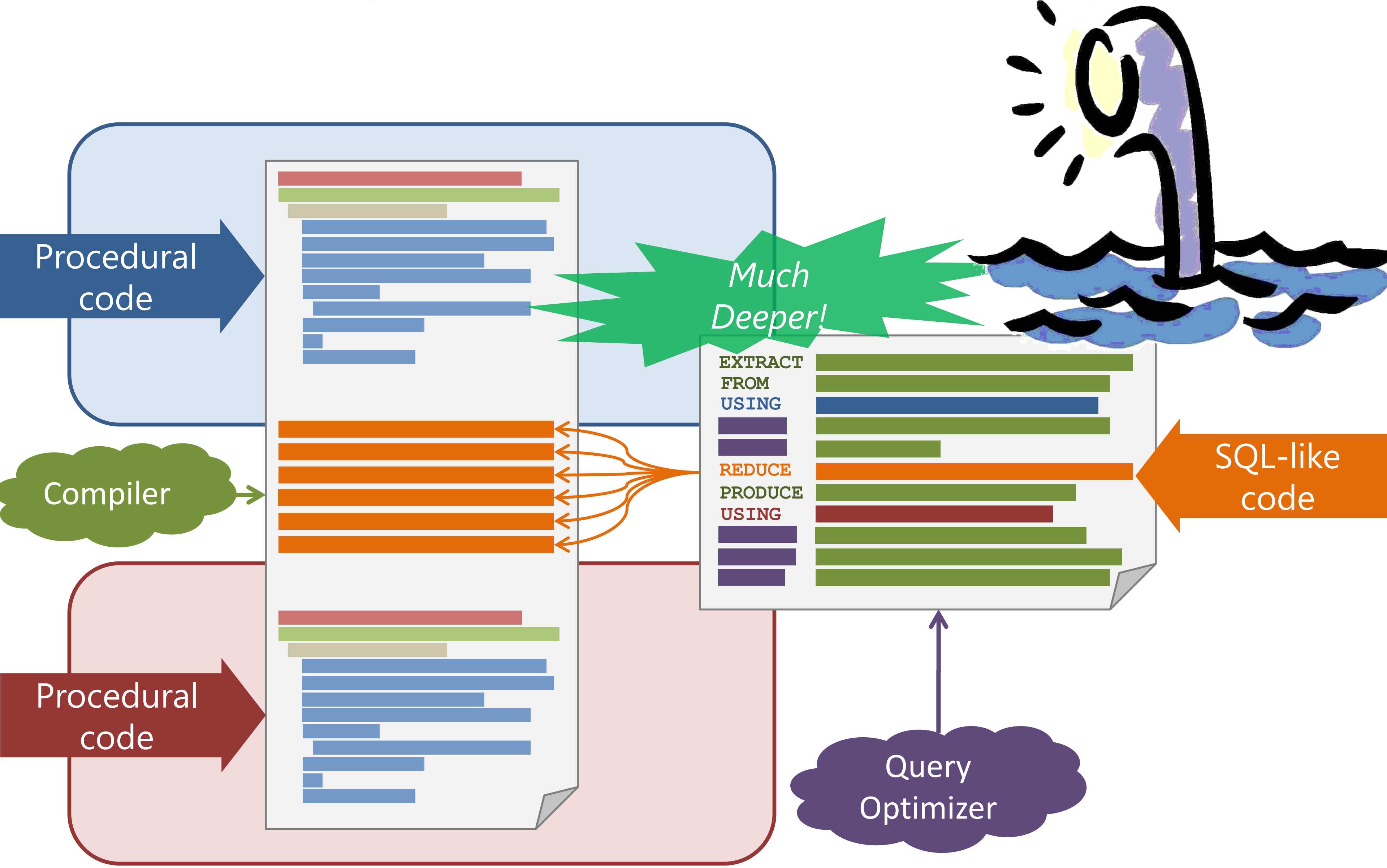
# A New Perspective


Source: http://adsoftheworld.com/media/print/alliance_francaise_quito_new_perspective?size=_original

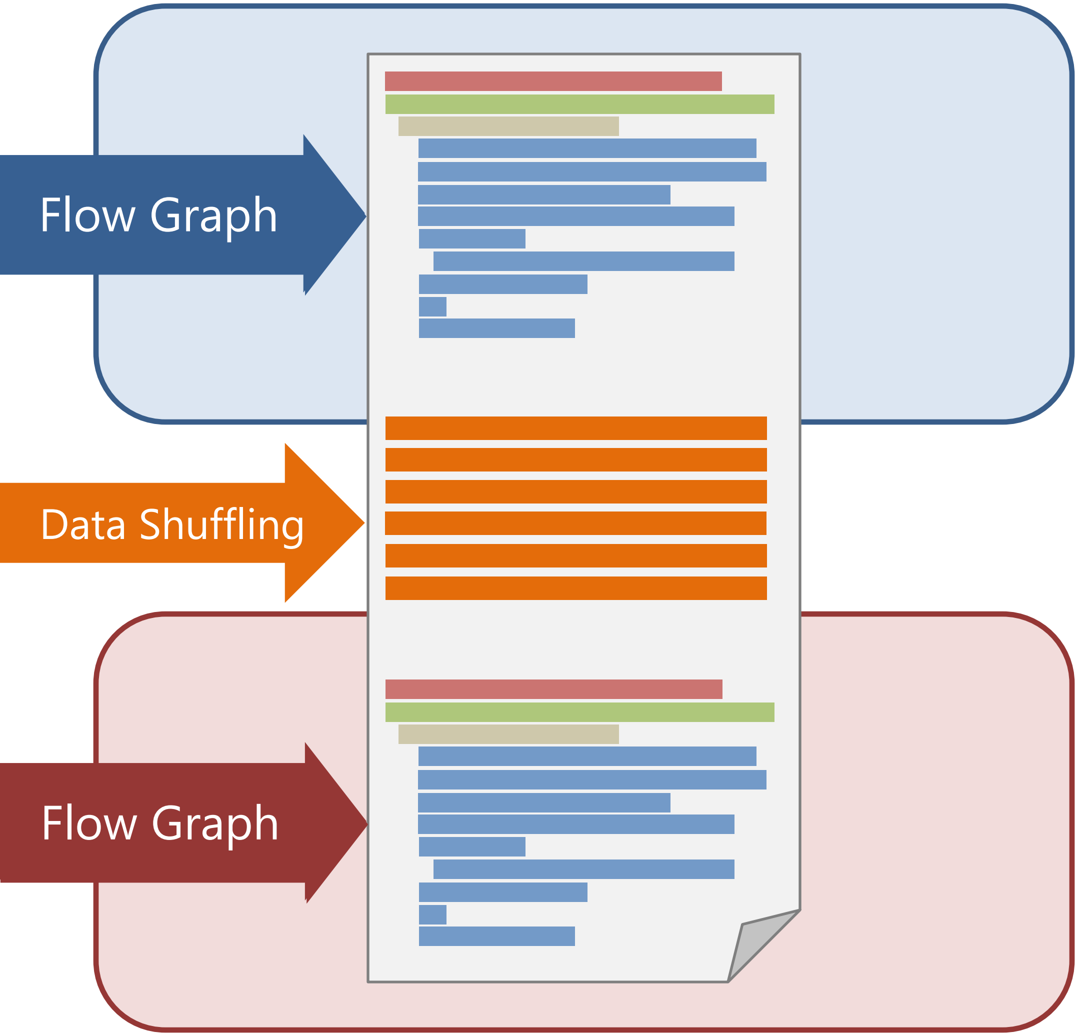
# **PeriSCOPE**: Pipeline-aware Holistic Code Optimization



# **PeriSCOPE**: Pipeline-aware Holistic Code Optimization

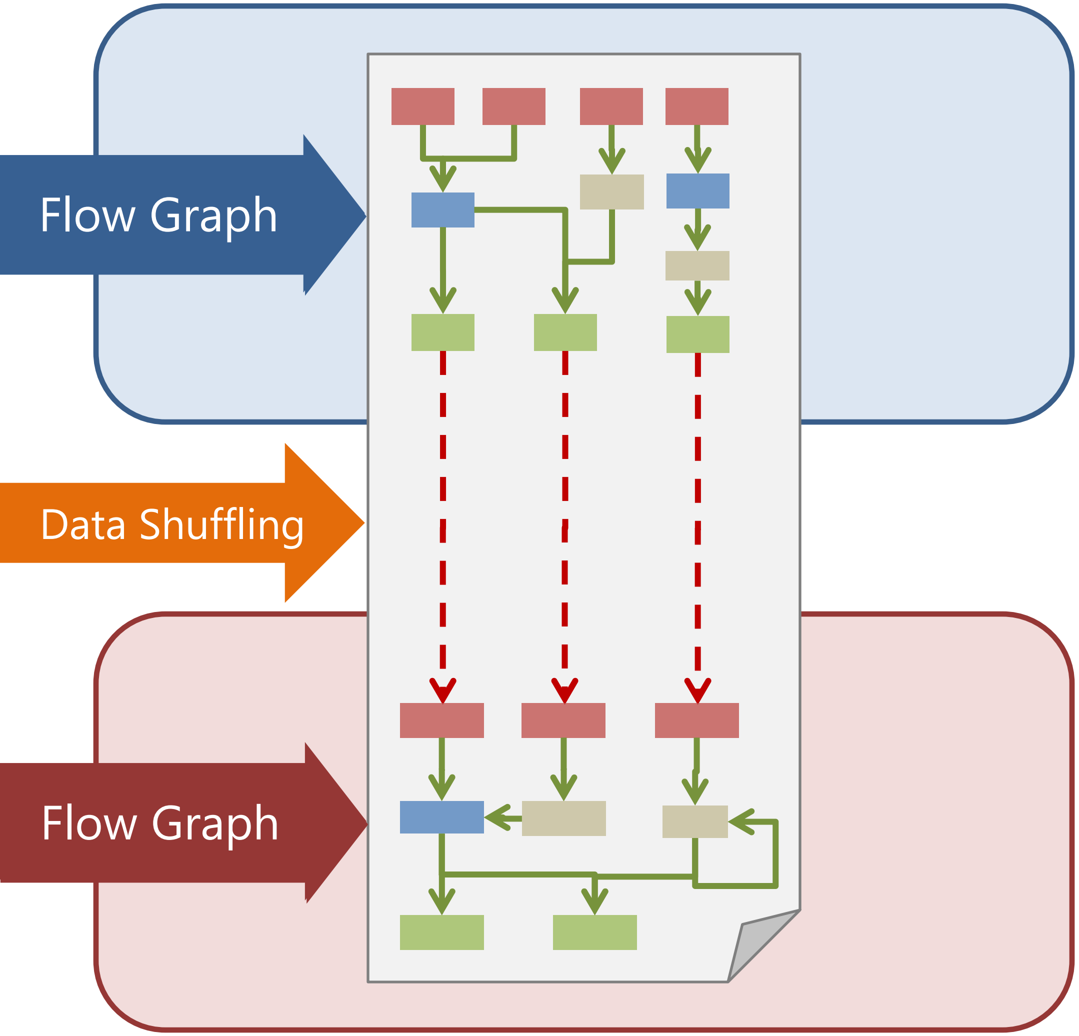


# Optimization Steps



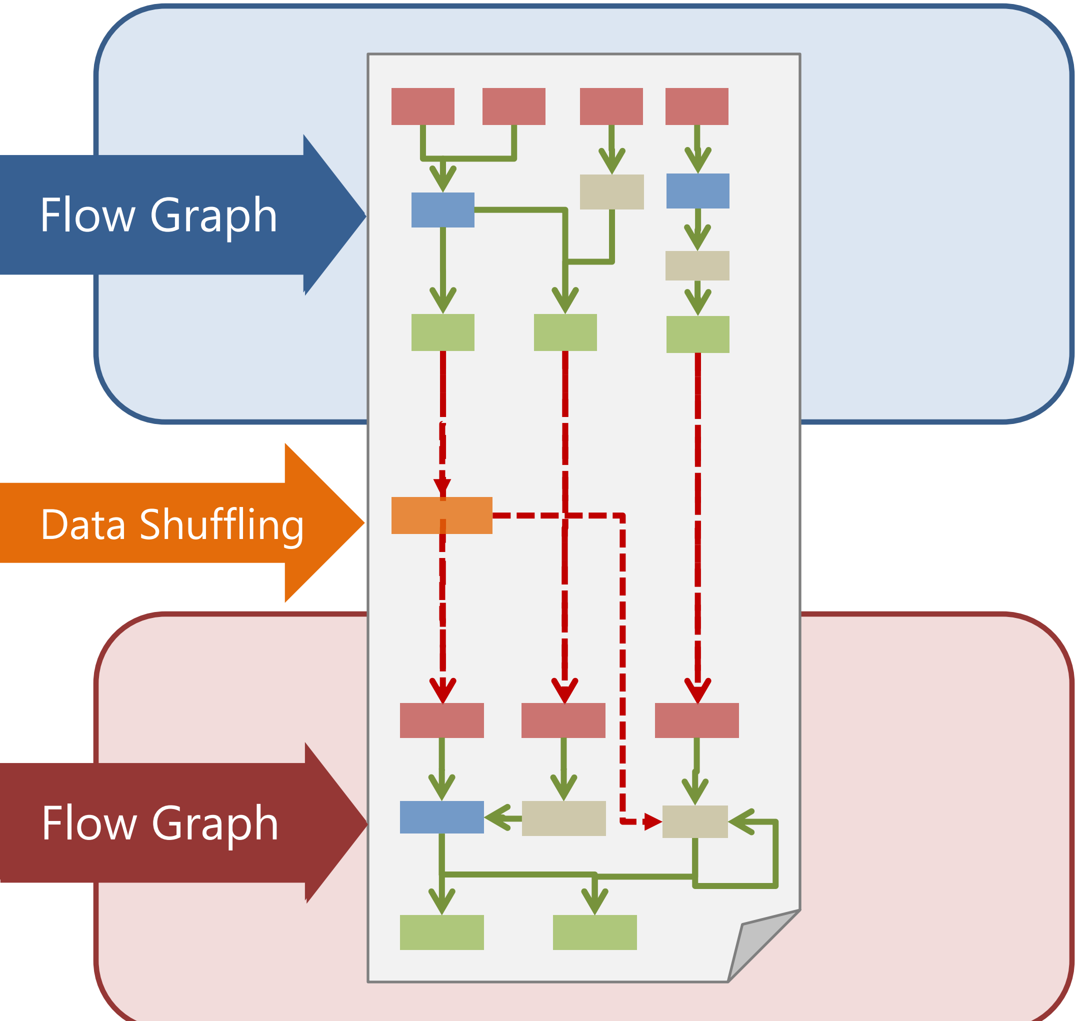

Step 1: Construct inter-procedural flow graph

# Optimization Steps




Step 1: Construct inter-procedural flow graph

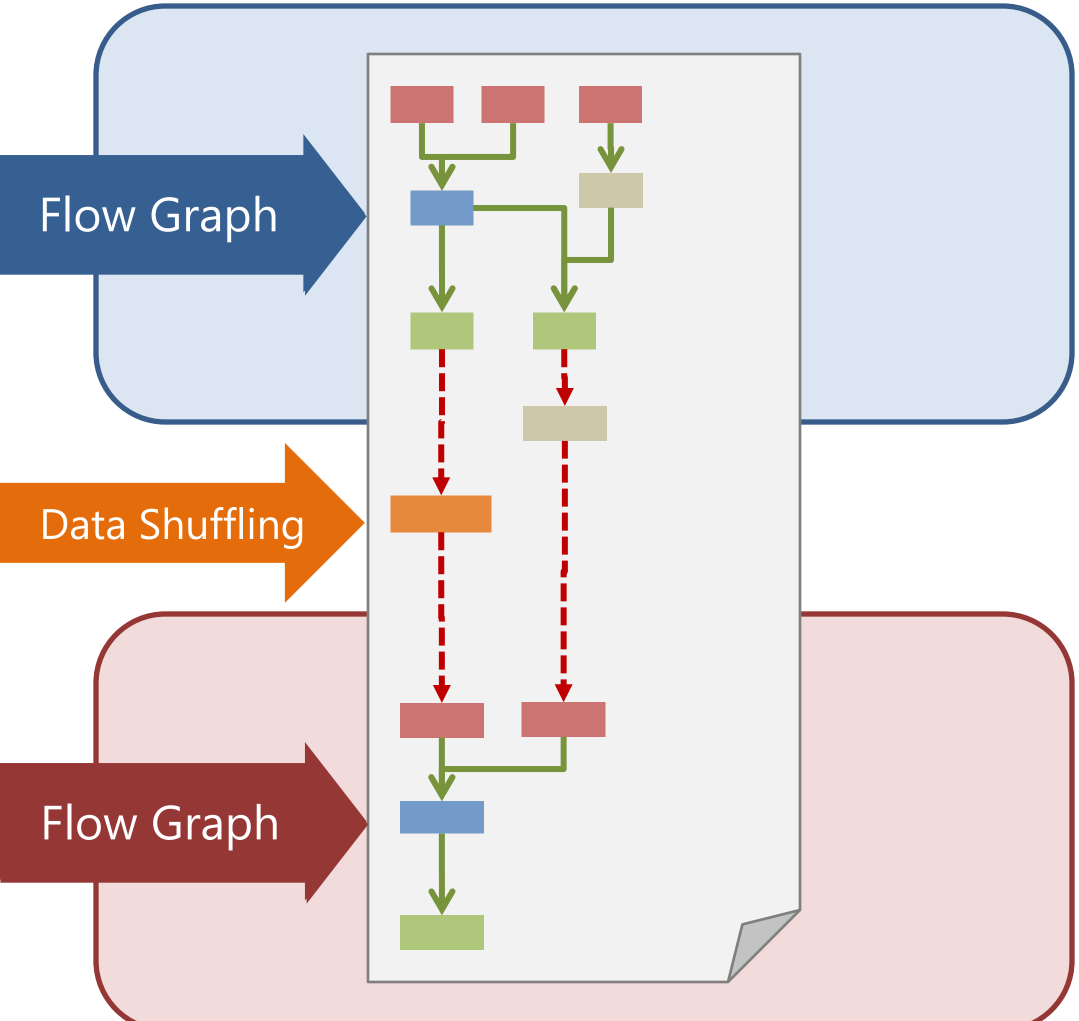
# Optimization Steps




Step 1: Construct inter-procedural flow graph

Step 2: Add safety constraints for skipping shuffling code

# Optimization Steps

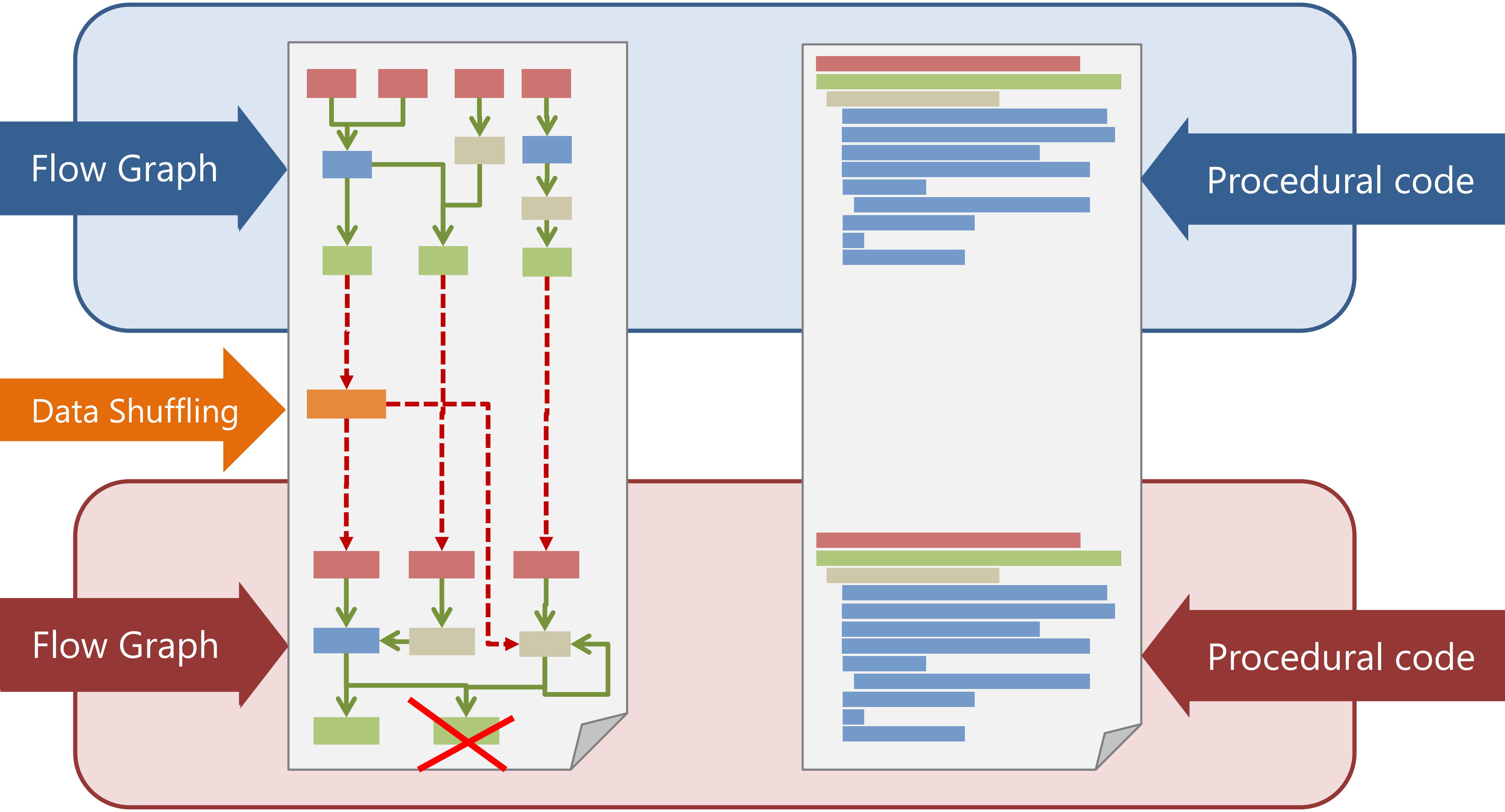



Step 1: Construct inter-procedural flow graph

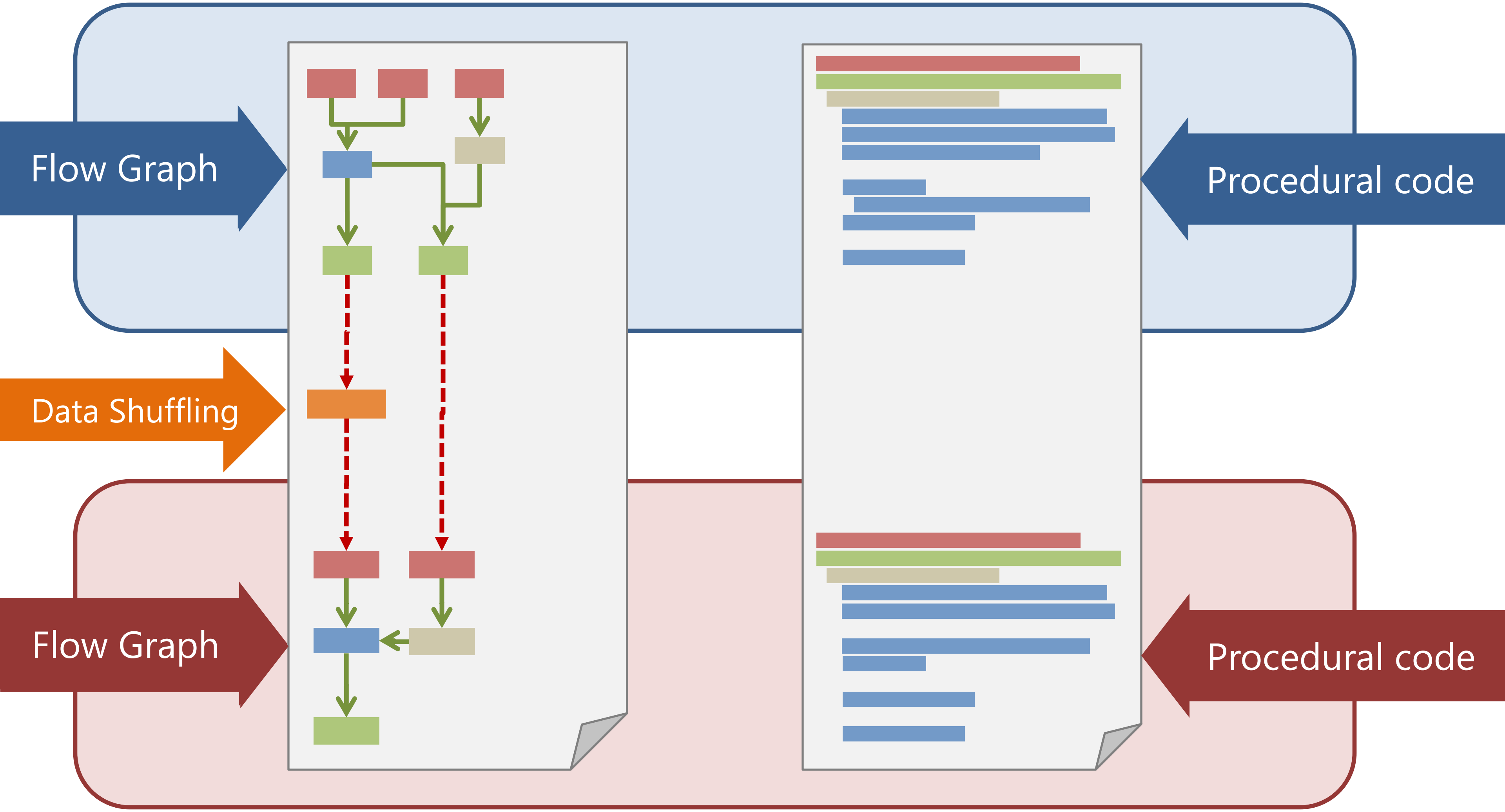
Step 2: Add safety constraints for skipping shuffling code

Step 3: Transform code for reducing shuffling I/O

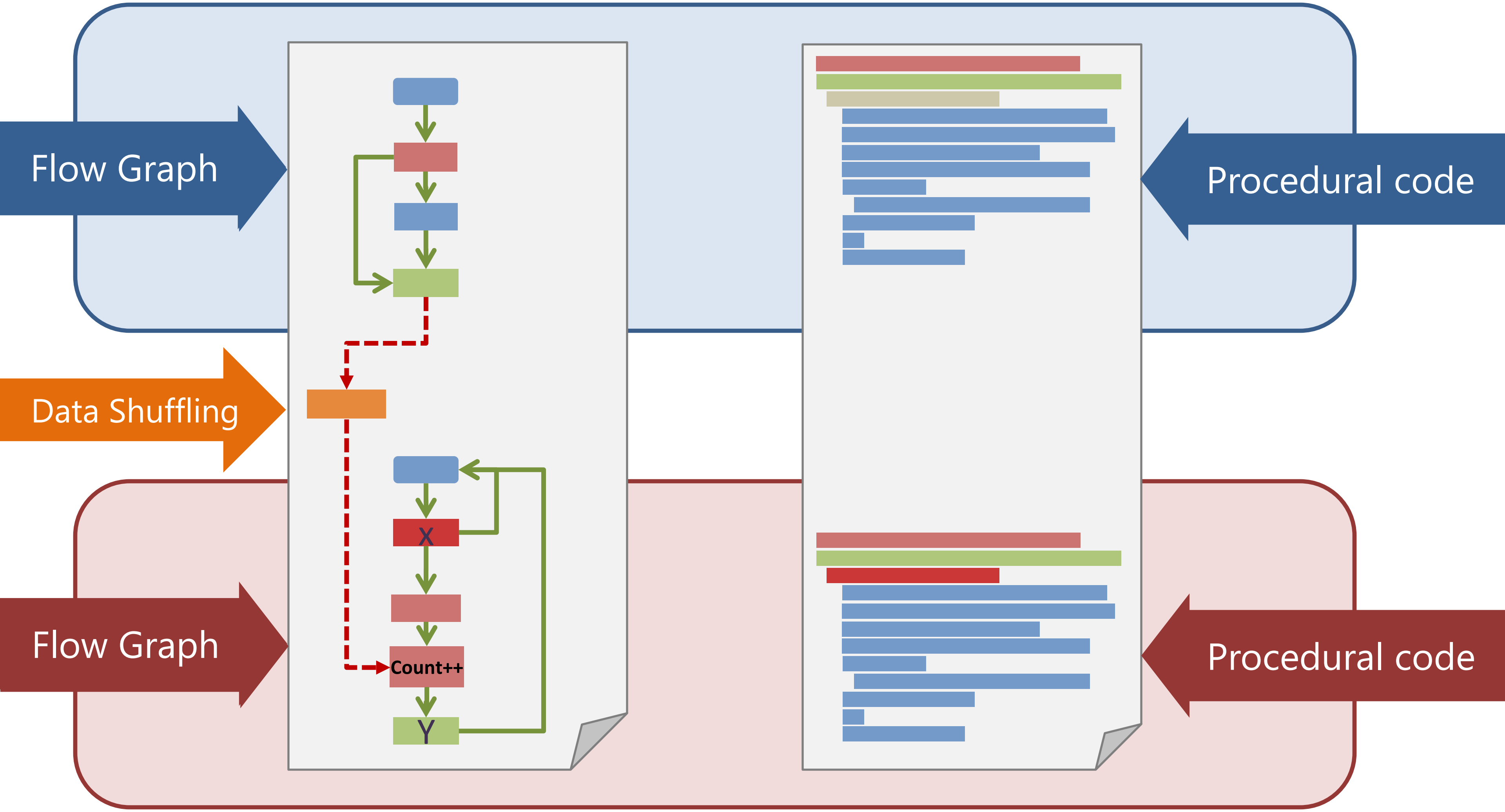
# Column Reduction: Reduce Number of Columns



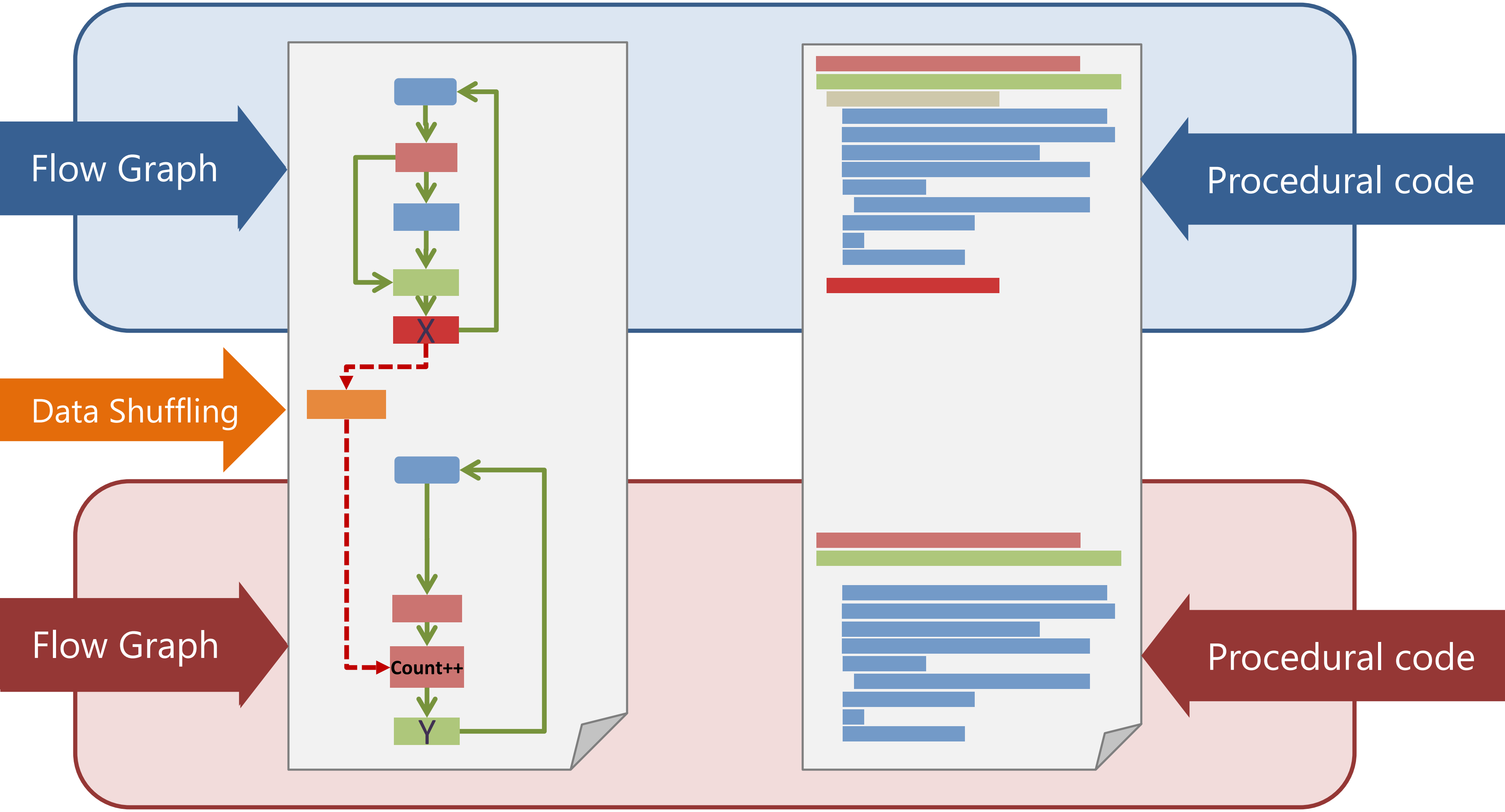
# Column Reduction: Reduce Number of Columns



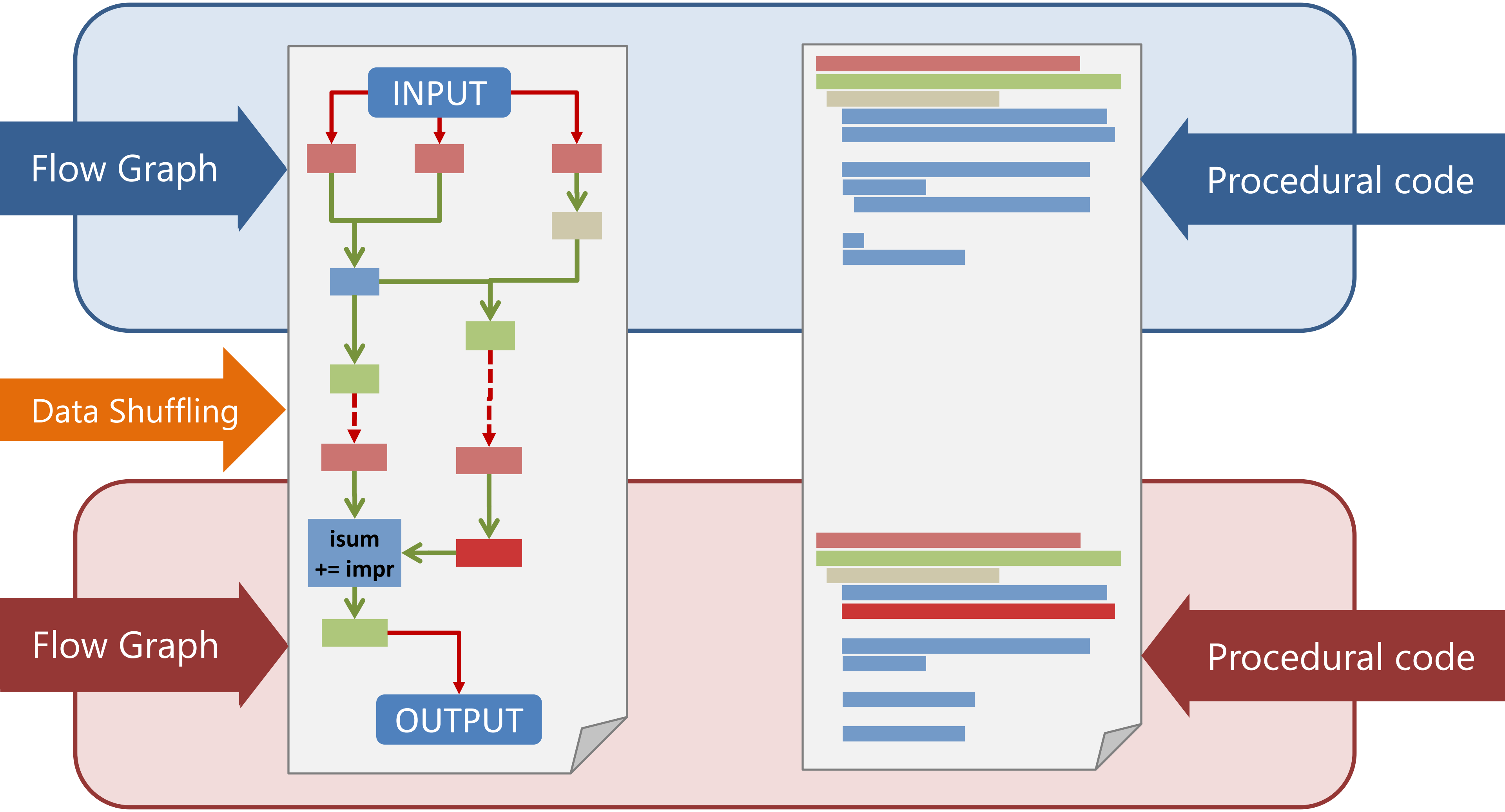
# Early Filtering: Reduce Number of Rows



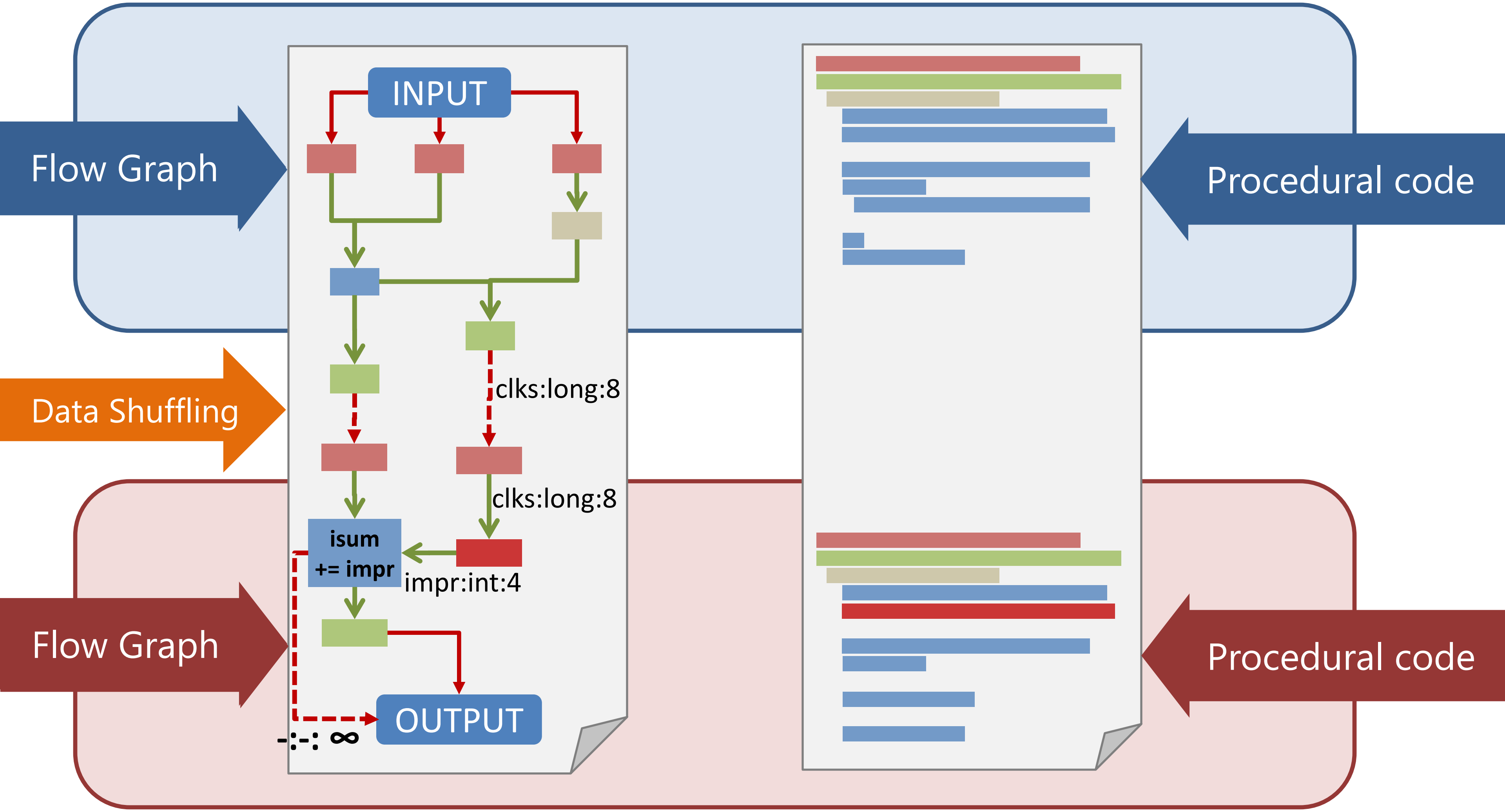
# Early Filtering: Reduce Number of Rows



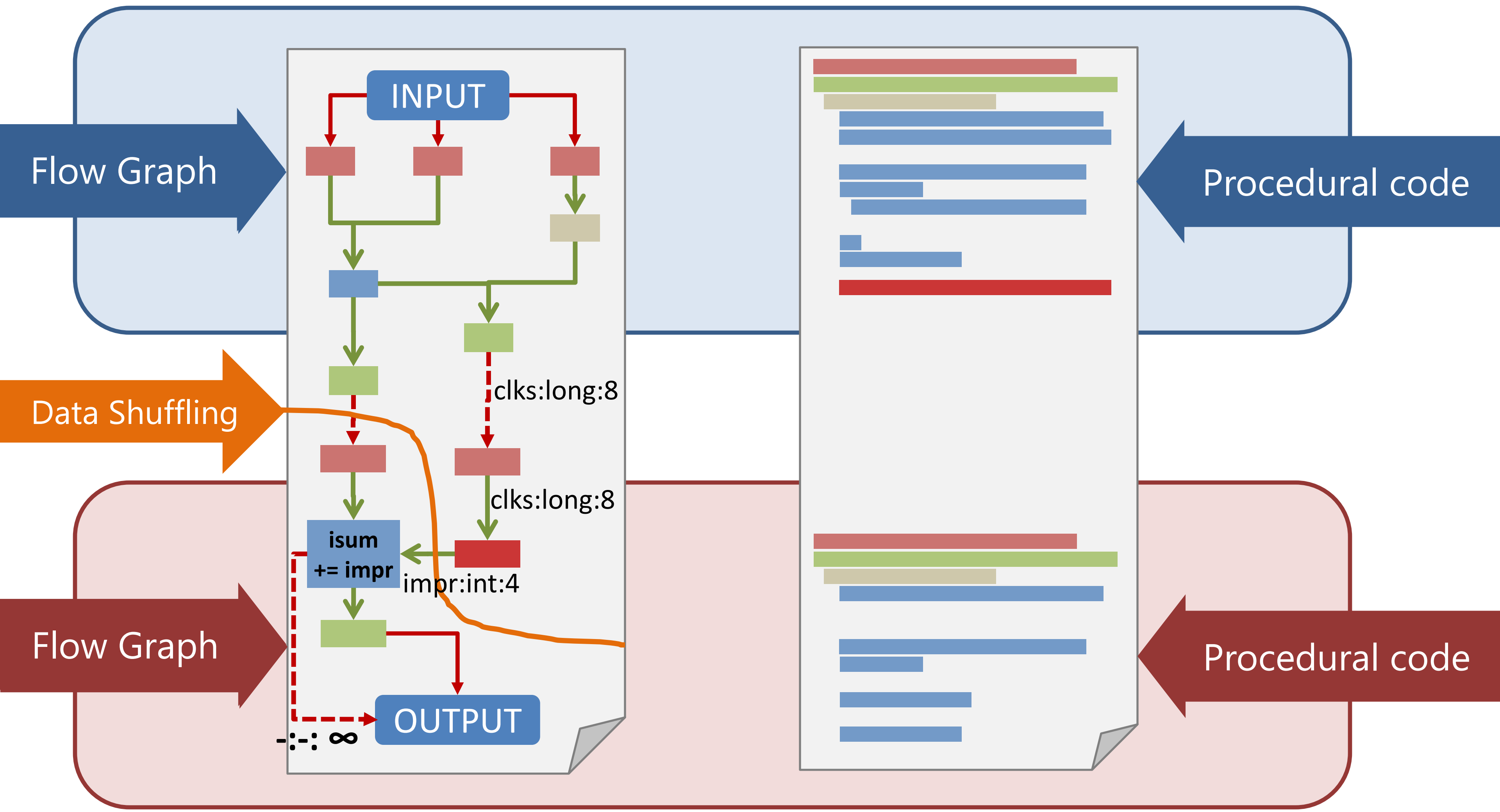
# Smart Cut: Reduce Size of Each Row



# Smart Cut: Reduce Size of Each Row



# Smart Cut: Reduce Size of Each Row

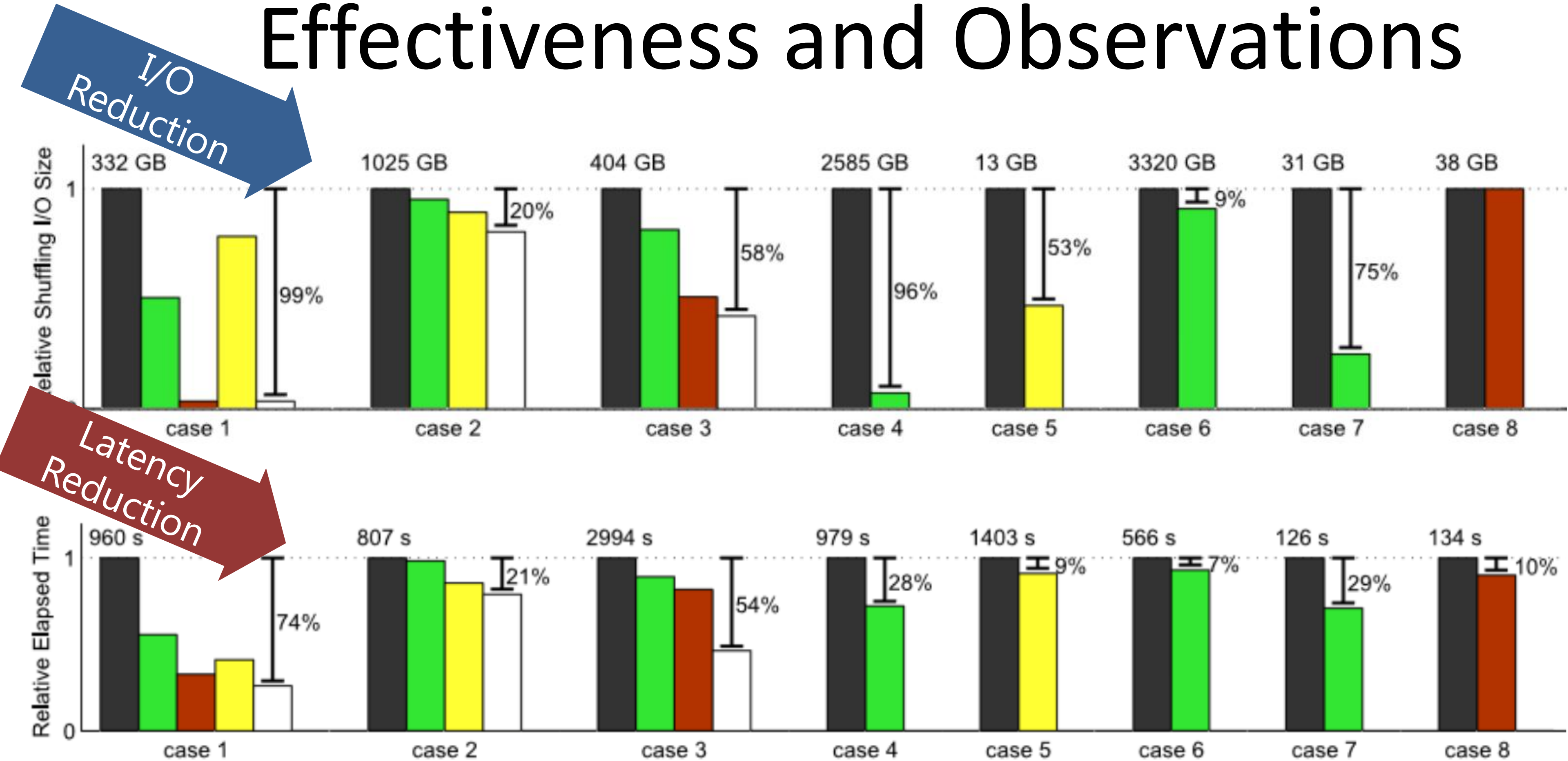


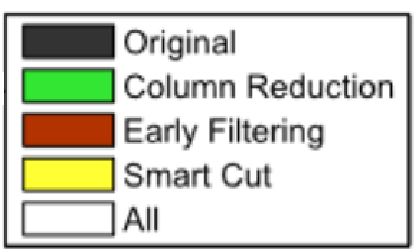
# Coverage Study*

| Optimization | Eligible jobs |
|---|---|
| Column Reduction | 4,052 (14.05%) |
| Early Filtering | 3,020 (10.47%) |
| Smart Cut | 1,544 ( 5.35%) |
| Overlapped Total | 6,397 (22.18%) |

* Study on **28,838** jobs collected from SCOPE clusters in 2010/2011.

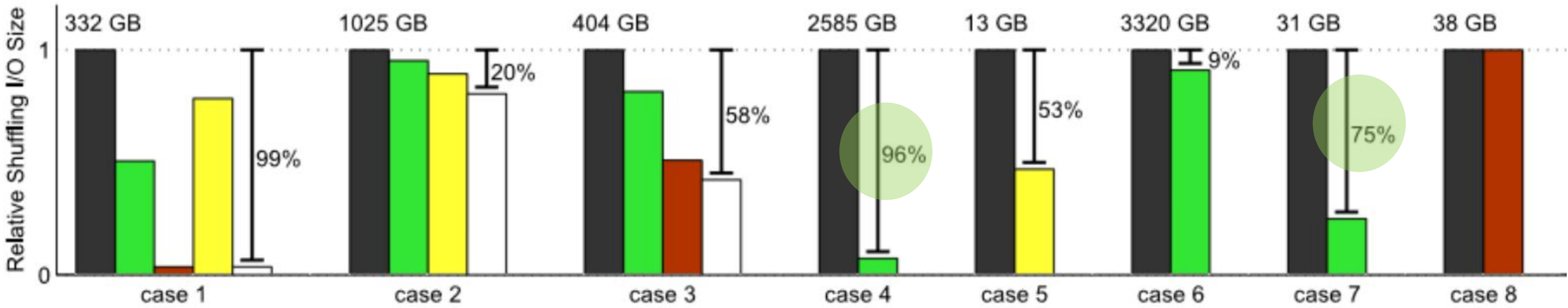
# Effectiveness and Observations



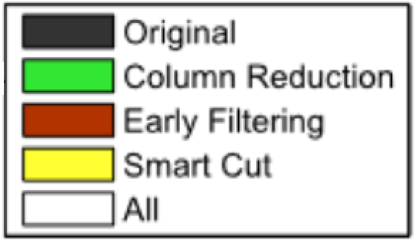




- I/O reduction is nice
- Latency reduction is generally smaller

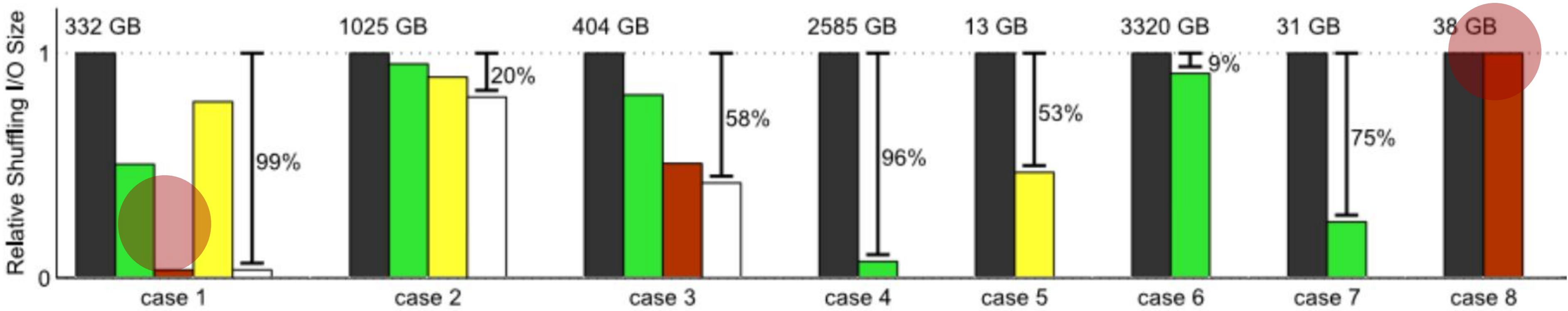
# Effectiveness and Observations



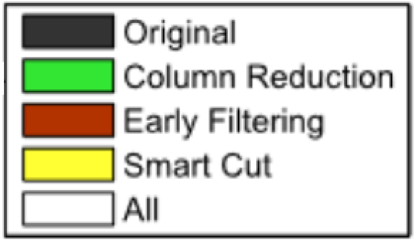

- Column Reduction
  - Case 4: 18 in 22 columns are eliminated
  - Case 7: 29 in 31 columns are eliminated
  - Mostly due to UDF reuse
    - 80.2% of the functions eligible for column reduction are reused more than 13 times



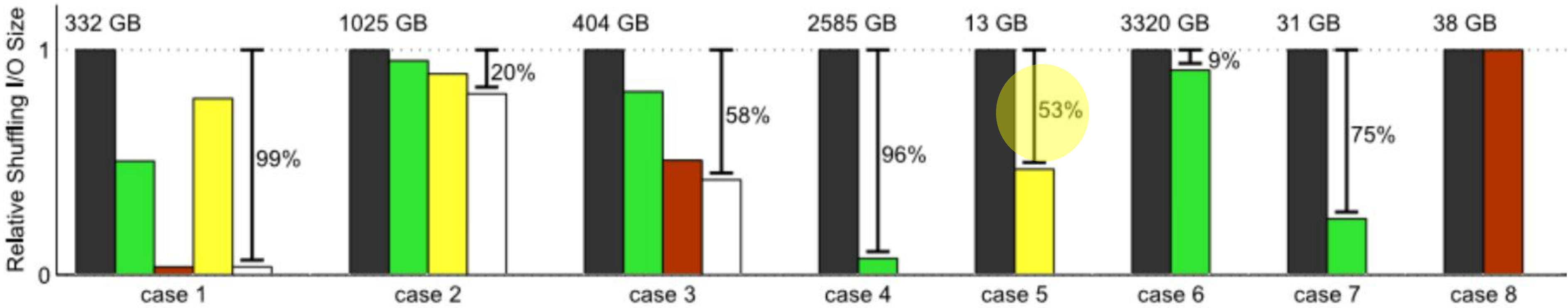
# Effectiveness and Observations



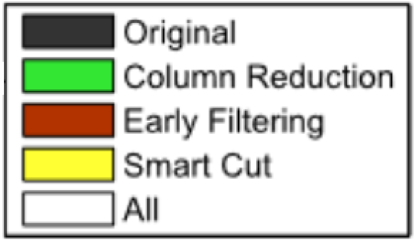

- Early Filtering
  - Exclude rows with invalid format
    - Case 8: ~0% reduction
  - Exclude rows with certain unwanted values
    - Case 1: 99% reduction



# Effectiveness and Observations




- Smart Cut
  - Unary operations
    - String to integer types
    - Trim, SubString
  - Binary operations
  - Case 5: DateTime.Parse(EndTs) - DateTime.Parse(StartTs)



# Applicability to various data-parallel computation systems

- Generally applicable (e.g., Scope/DryadLINQ/Hive/Pig Latin)
- Impact factors to the coverage and effectiveness
  - Data model
    - Relational
    - Object
  - API interface
    - Map(List<Row> *rows*, …)
    - Map(Row *row*, …)

# Future Directions

- Balance how easy it is for programming and how easy it is for automatic optimization
  - Extract common computation patterns
  - Redesign programming interface to achieve better trade-off
    - Interfaces higher than MapReduce?

# Future Directions

- Explore other components other than distributed data-parallel computation systems in large scale internet service systems
  - e.g., automatic caching & prefetching for user-facing web service frameworks

# Conclusion

- Pipeline-aware holistic code optimization is promising
  - Project pipeline information to procedural code
  - Add safety rules to ensure correctness
  - I/O driven compiler-like optimization
- Improve performance without sacrificing programmability
- Considering more about how easy it is for optimization when designing programming frameworks

# Thanks!
# Questions?

Spotting Code Optimizations in Data-Parallel Pipelines through **PeriSCOPE**

Zhenyu Guo, Xuepeng Fan, Rishan Chen, Jiaxing Zhang, Hucheng Zhou,
Sean McDirmid, Chang Liu, Wei Lin, Jingren Zhou, Lidong Zhou
Microsoft Research Asia
Microsoft BING