WISE: Predicting the Performance of Sparse Matrix Vector Multiplication with Machine Learning

Serif Yesil[*], Azin Heidershenas[*], Adam Morrison[+], ***Josep Torrellas***[*]

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

Motivation

- Sparse Matrix-Vector Multiplication (SpMV)
  - An essential kernel
- Used in many different domains:
  - Graph processing and linear solvers
- Low-locality memory accesses
- Widely different behavior based on the sparse matrix used

The Challenge

- Numerous SpMV methods are proposed
- SpMV methods' performance is hard to predict
- Different methods work best for different classes of sparse matrices

How can we choose the best method for a given sparse matrix?

Our Contribution: WISE

- **WISE: An ML-based framework to predict the best SpMV method for a given sparse matrix**
  - Uses a novel feature set that models size, locality, and skew characteristics
  - Considers a wide range of SpMV methods (i.e., optimizations)
  - Attains a 2.4x speedup on average over state-of-the-art Intel MKL

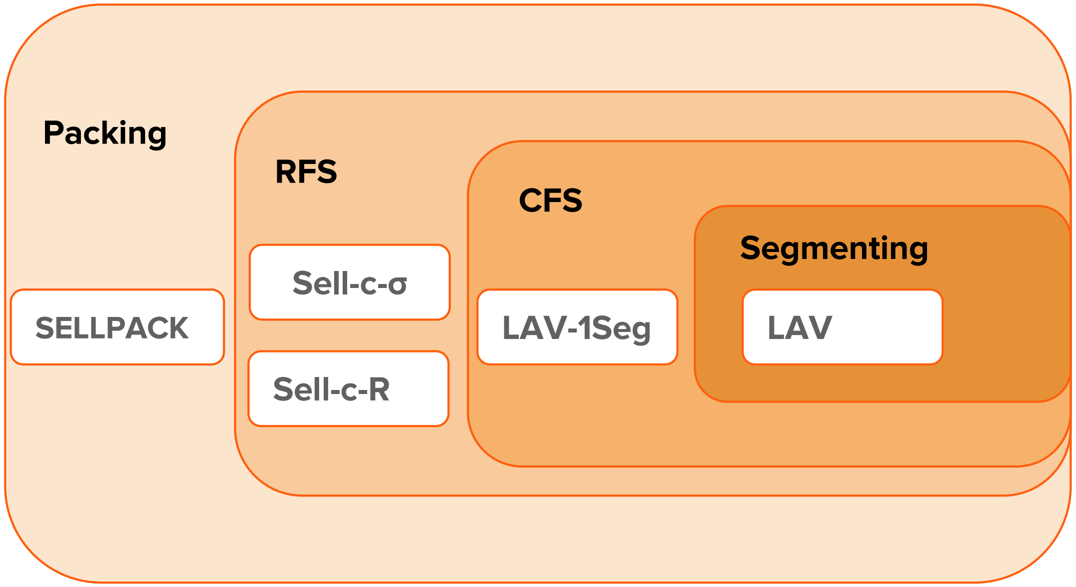
SpMV Method Space

Packing: Enables vectorization

Row Frequency Sorting (RFS): Zero padding minimization to improve vectorization

Column Frequency Sorting (CFS): Places frequently accessed elements of the input vector together

Segmenting: Improves last-level cache use



All methods use vectorization

No One-Size-Fits-All Solution

*SuiteSparse: A matrix collection (sparse.tamu.edu)

Different matrices prefer different SpMV methods

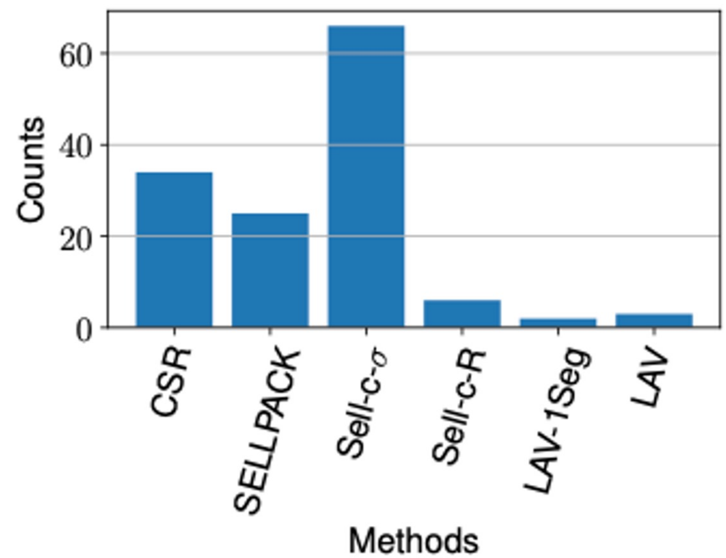
- Sell-c-σ (66), CSR (34), SELLPACK (25)

Highest speedup for a method varies

- SELLPACK: 1.05-1.31×
- Sell-c-σ: 1.00-1.76×

Each method can take different parameters

- Selecting the correct parameter values is crucial: 10× slowdown



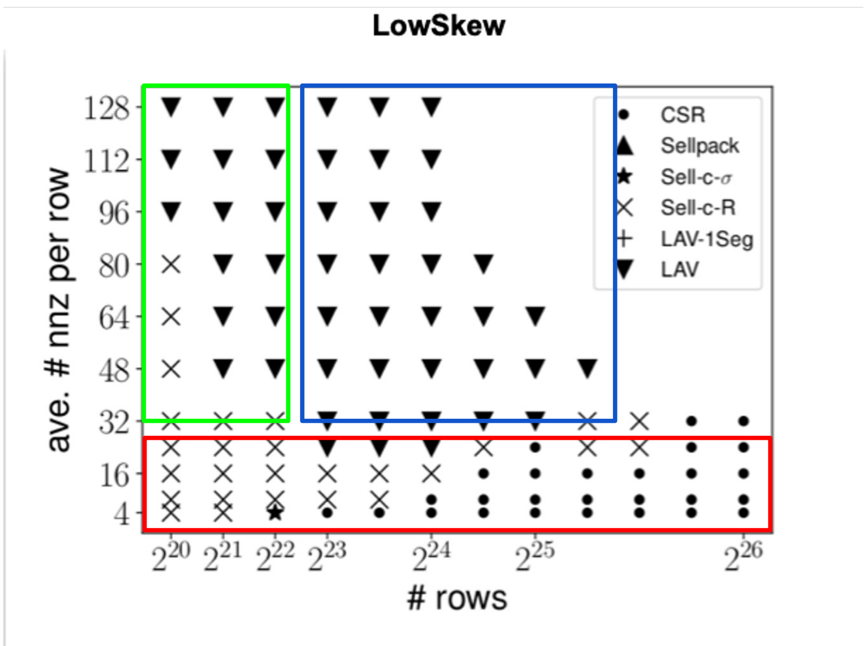
Are there any patterns that we can detect?

Example: Effect of the #Rows and Avg #Non-zeros/row

`LAV`: Large matrices

`Sell-c-R or CSR`: Matrices with for low average nnz per row

LAV and Sell-c-R: Matrices with high average nnz per row and few rows



WISE's Approach

- The fastest method varies across matrices
- Within a method, the magnitude of the speedup varies

⇒**Predict rough speedup**

WISE's Solutions

- The fastest method varies across matrices
- Within a method, the magnitude of the speedup varies

  ⇒**Predict rough speedup**
- Parameter selection for a method affects the speedups substantially

  ⇒**Create individual ML models for {method, parameter} pairs**

WISE's Solutions

- The fastest method varies across matrices
- Within a method, the magnitude of the speedup varies

  ⇒**Predict rough speedup**
- Parameter selection for a method affects the speedups substantially

  ⇒**Create individual ML models for {method, parameter} pairs**
- SuiteSparse matrices are biased towards a few types of matrices (few power law matrices)

  ⇒**Augment SuiteSparse matrices with a representative set of synthetic matrices**

WISE's Solutions

- The fastest method varies across matrices
- Within a method, the magnitude of the speedup varies

⇒**Predict rough speedup**

- Selecting correct parameters for a method affect the speedups substantially

⇒**Create individual ML models for {method, parameter} pairs**

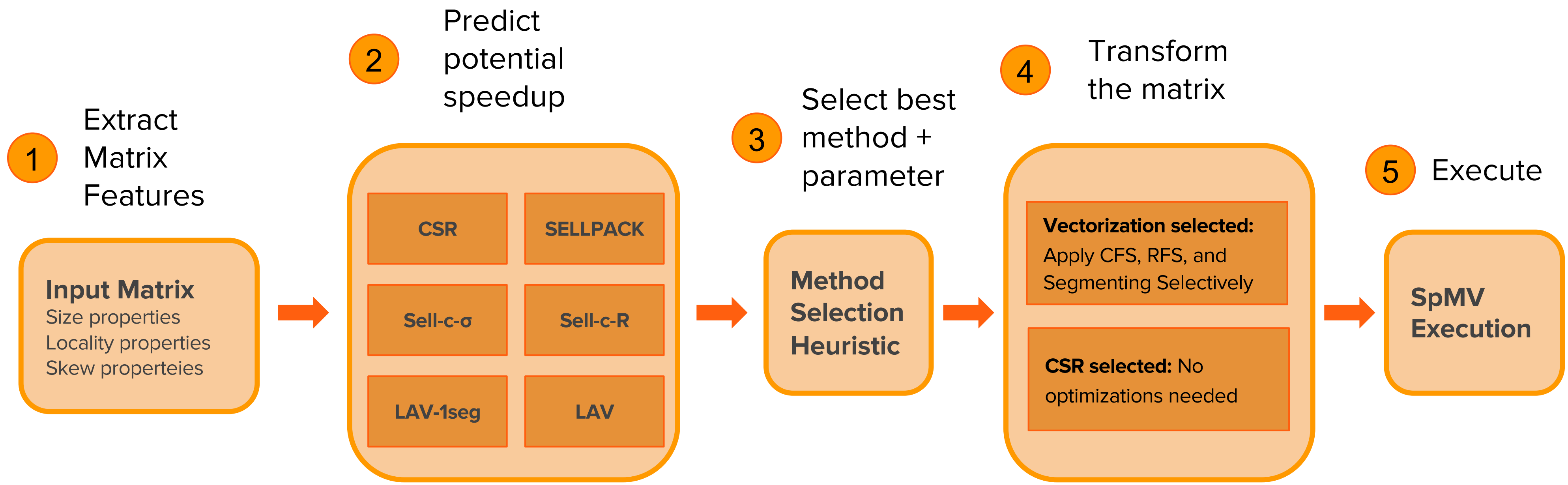
- SuiteSparse matrices are biased towards a few types of matrices (few power law matrices)

⇒**Augment SuiteSparse matrices with a representative set of synthetic matrices**

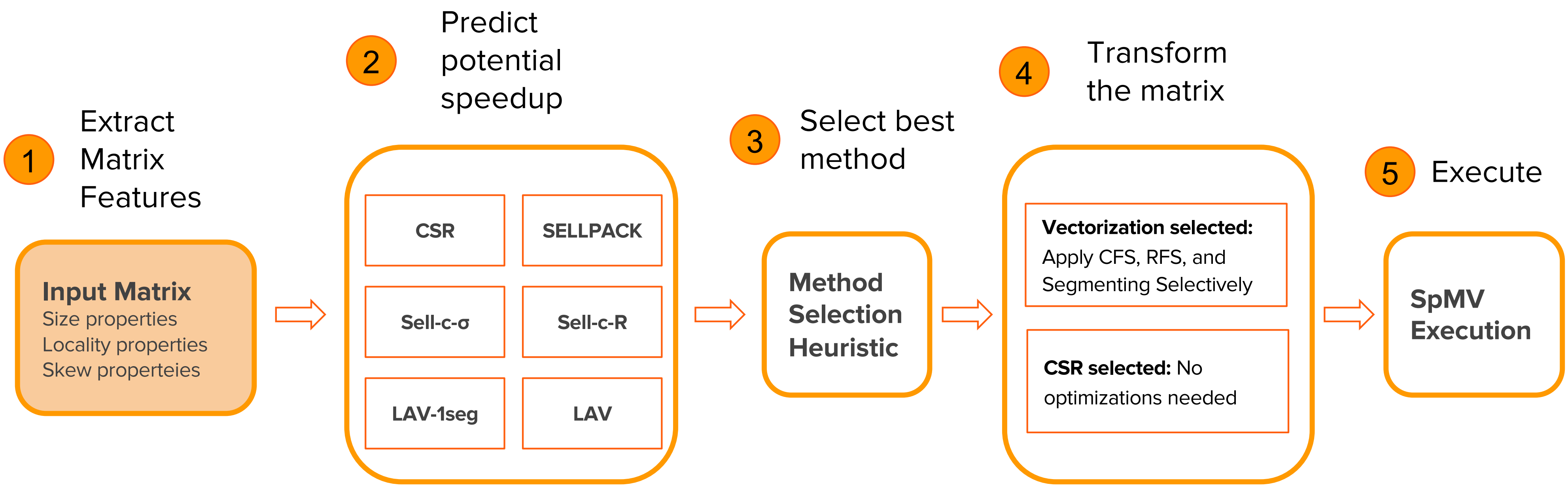
- Complex relationship between matrix size, locality of non-zeros, and skew of non-zeros

⇒**Select a new sparse matrix feature set**

WISE in Action



Extracting Matrix Features



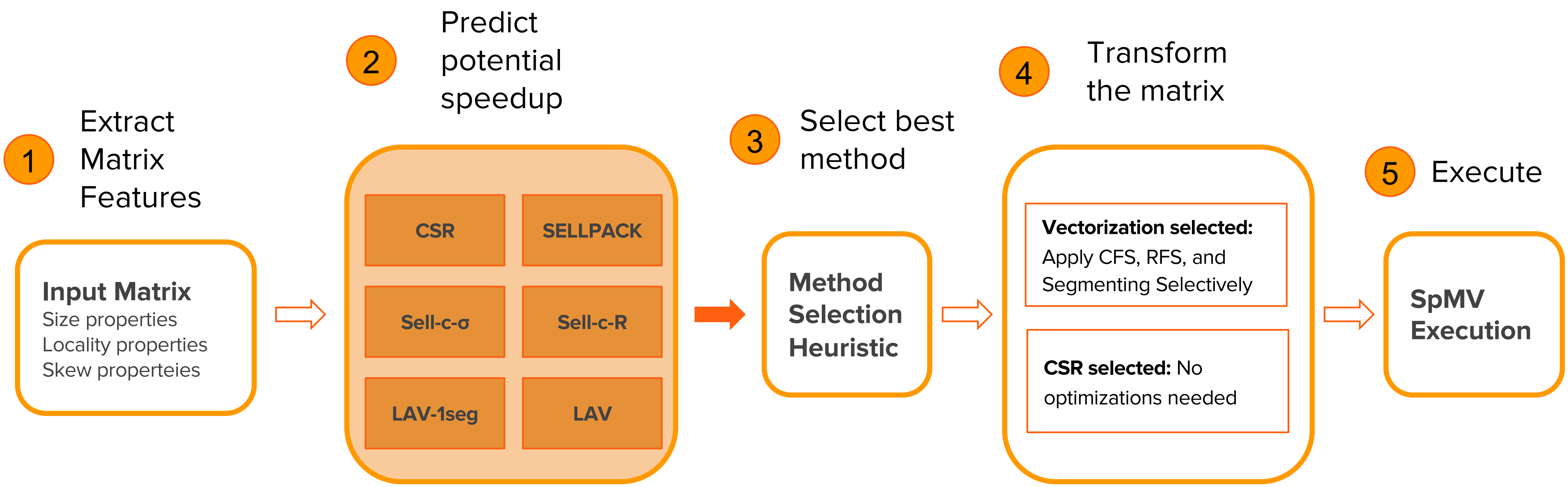
Extracting Matrix Features

- **Size Characteristics**
  - Amount of work to be done: Number of rows, columns, and nonzeros
- **Skew Characteristics of Non-Zeros**
  - Rows: Scheduling, vector unit utilization characteristics
  - Columns: Irregularity of input vector accesses
- **Locality Characteristics of Non-Zeros**
  - Tiles, Row of Tiles, and Column of Tiles: Locality in L1 and L2
  - Behavior across Tiles: Locality in last level cache

Time taken to generate the features: Avg 1 MKL SpMV iterations (max 5)

Predicting The Potential Speedup

Create an individual ML model for each SpMV {method, parameter} pair



WISE ML Models

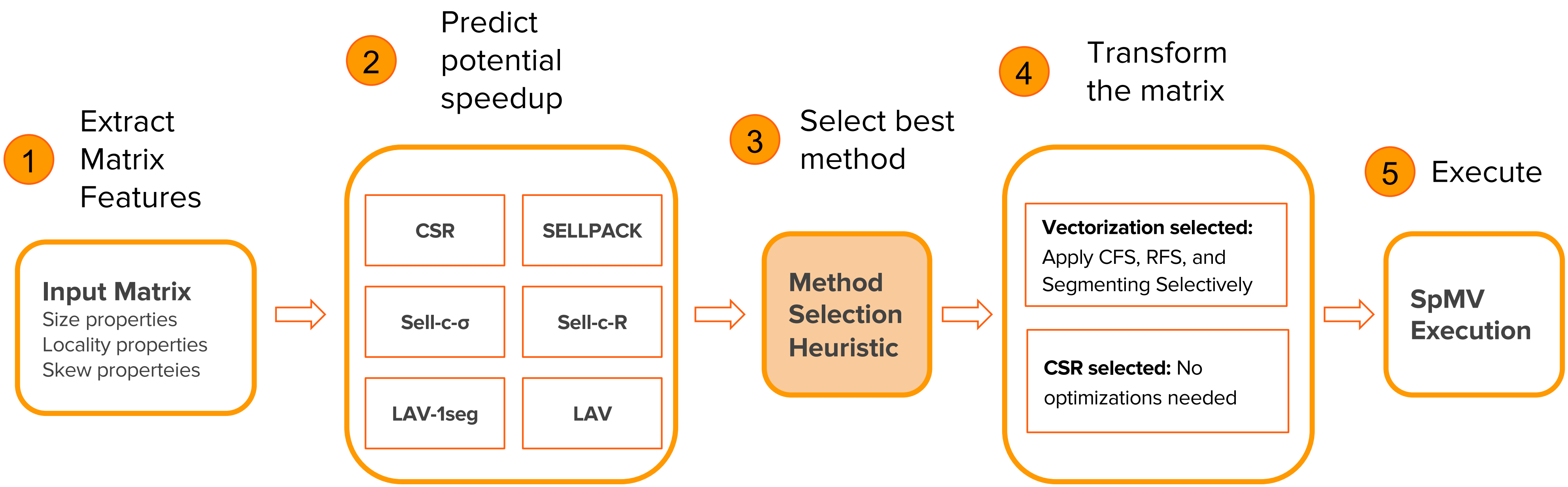
Create an individual decision tree for each SpMV {method, parameter} pair

- **CSR**: Scheduling parameter (dynamic, static, static contiguous)
- **SELLPACK**: SIMD length, scheduling parameter
- **Sell-c-σ**: σ parameter, SIMD length, scheduling parameter
- **Sell-c-R, LAV-1Seg**: SIMD length
- **LAV**: Threshold of dense portion, SIMD length

About 35 different decision trees of max depth 15

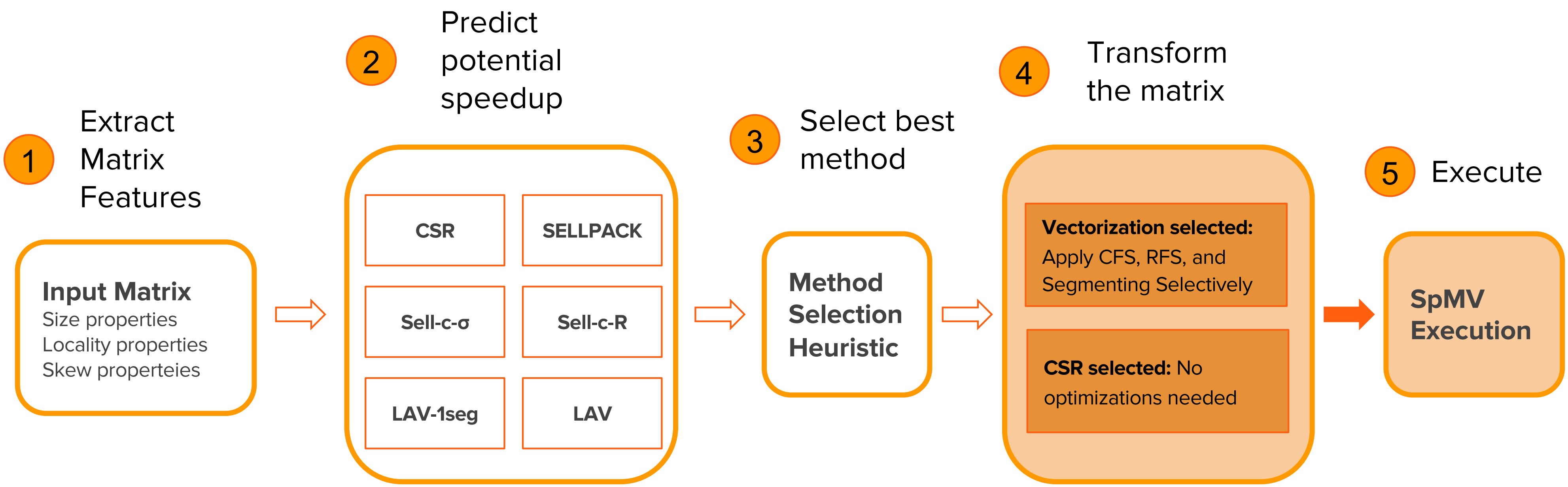
The Method Selection Heuristic

- We do not predict the exact speedup but a range
- If there is a tie: Choose the cheapest method

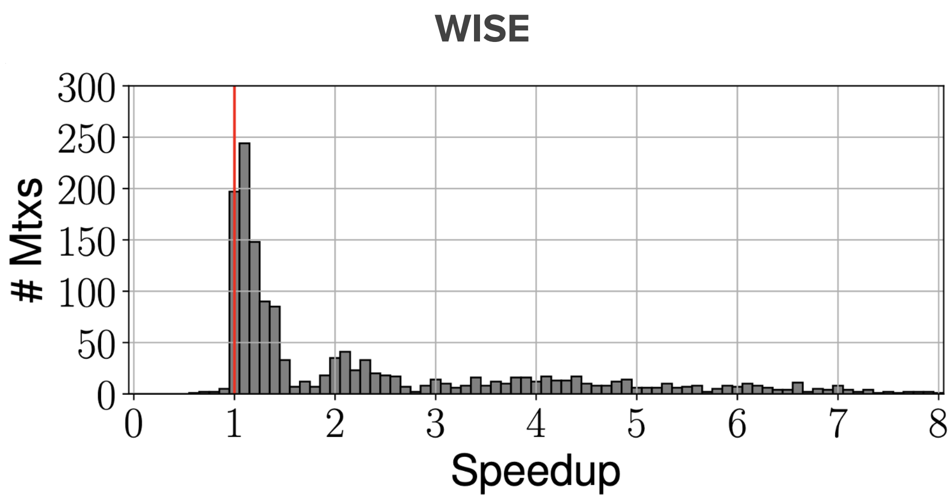


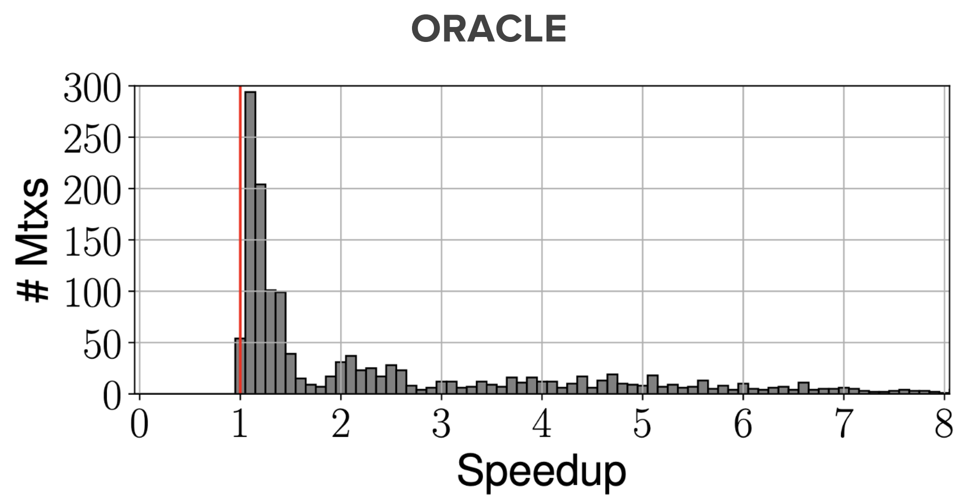
Optimize and Execute

- Transform matrices into correct format and execute SpMV



WISE's Speedup over the Intel MKL Library





An average speedup of 2.4× over Intel MKL

Oracle method (ground truth) 2.5× speedup over Intel MKL

Intel MKL inspector-executor: 2.1× speedup over Intel MKL

Intel MKL inspector-executor overhead is 17 MKL iterations, WISE is 50% lower

More in the paper...

- More analysis on matrix characteristics
- How are the features calculated?
- Details of the ML models
- Performance of individual ML models generated by WISE

Conclusions

- Different SpMV methods work best for different sparse matrices
- WISE: An ML based framework to predict the speedup of SpMV methods
    - A novel feature set that captures the locality and skew characteristics of non-zeros
    - Considered a wide range of SpMV methods and parameter values
    - Attains a 2.4x speedup on average over state-of-the-art Intel MKL

WISE: Predicting the Performance of Sparse Matrix Vector Multiplication with Machine Learning

Serif Yesil*, Azin Heidershenas*, Adam Morrison+, ***Josep Torrellas****

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

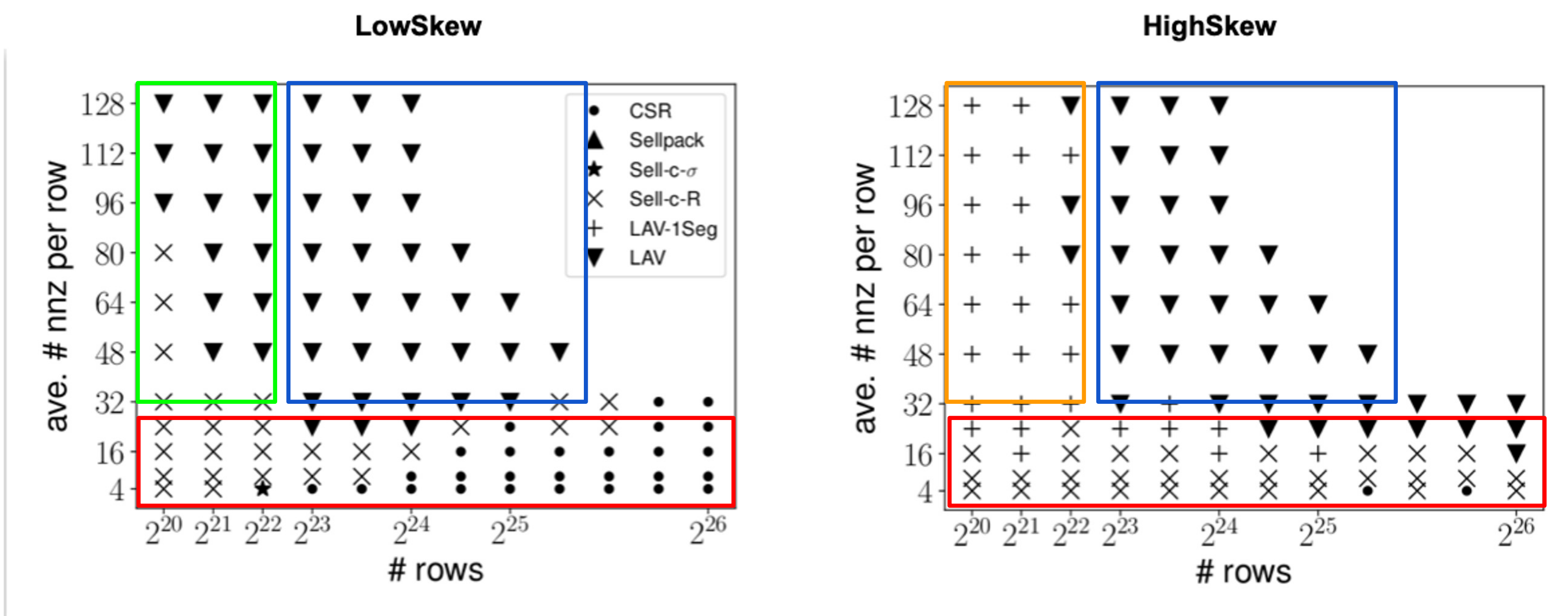
Example: Effect of the Nonzero Skew in the Matrix

`LAV` for large matrices

`Sell-c-R`: Matrices with for low average nnz per row

`LAV-1Seg`: HighSkew matrices with high average nnz per row and few rows

LAV and Sell-c-R: LowSkew matrices with high average nnz per row and few rows



Locality Characteristics vs. SpMV Methods

`Sell-c-`σ is generally the best

`LAV` outperforms for large matrices due to segmenting