INTEL® MKL - SPARSE BLAS

Gennady Fedorov - Technical Consulting Engineer

Intel Architecture, Graphics and Software (IAGS)

PRACE workshop, June 2020

Gennady.Fedorov@intel.com

Intel® Math Kernel Library

Linear Algebra
- BLAS
- LAPACK
- ScaLAPACK
- **Sparse BLAS**
- Iterative sparse solvers
- PARDISO*
- Cluster Sparse Solver

FFTs
- Multidimensional
- FFTW interfaces
- Cluster FFT

Neural Networks
- Convolution
- Pooling
- Normalization
- ReLU
- Inner Product

Removed since MKL v.2020

Vector RNGs
- Congruential
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neiderreiter
- Non-deterministic

Summary Statistics
- Kurtosis
- Variation coefficient
- Order statistics
- Min/max
- Variance-covariance

Vector Math
- **Trigonometric**
- **Hyperbolic**
- **Exponential**
- **Log**
- **Power**
- **Root**

And More
- Splines
- Interpolation
- Trust Region
- Fast Poisson Solver

Benchmarks
- Intel(R) Distribution for LINPACK* Benchmark
- High Performance Computing Linpack Benchmark
- High Performance Conjugate radient Benchmark

Intel® Architecture Platforms

Operating System: Windows*, Linux*, MacOS[1]*

[1] Available only in Intel® Parallel Studio Composer Edition.

Intel® MKL - Deprecation

- **Deep Neural Network (DNN)**
  - DNN is deprecated and will be removed in the next Intel MKL release. We will continue to provide optimized functions for deep neural networks in Intel Math Kernel Library for Deep Neural Networks (Intel MKL-DNN)
- **Removed support for 32 bit applications on macOS***
  - If users require 32-bit support on macOS*, they should use MKL 2018 or early versions
- **SpBLAS (NIST) API**
  - Sparse BLAS API is deprecated* and will be removed in the next Intel MKL release. Please use sparse BLAS IE API instead of.

*** deprecated since MKL 2018 Update 2**

Sparse BLAS API

```
mkl_?csrmv(&transa, &m, &k, &alpha, matdescra, val, indx, pntrb, pntre, x,
&beta, y); y := alpha*A*x + beta*y
```

Overview

- Based on NIST 1995 standard
- One call per operation, all data passed via parameters
- Supports wide range of matrix formats (CSR, CSC, BSR, DIA, COO, SKY)

Limitations

- No way to pass tuning information between calls
  - No autotuning, no balancing
- Works with pre-allocated memory
- Exposes data structure as set of parameters
- Modern formats make calls cumbersome

Inspector– Executor Sparse BLAS API, Overview

IE API for Sparse BLAS two stages:

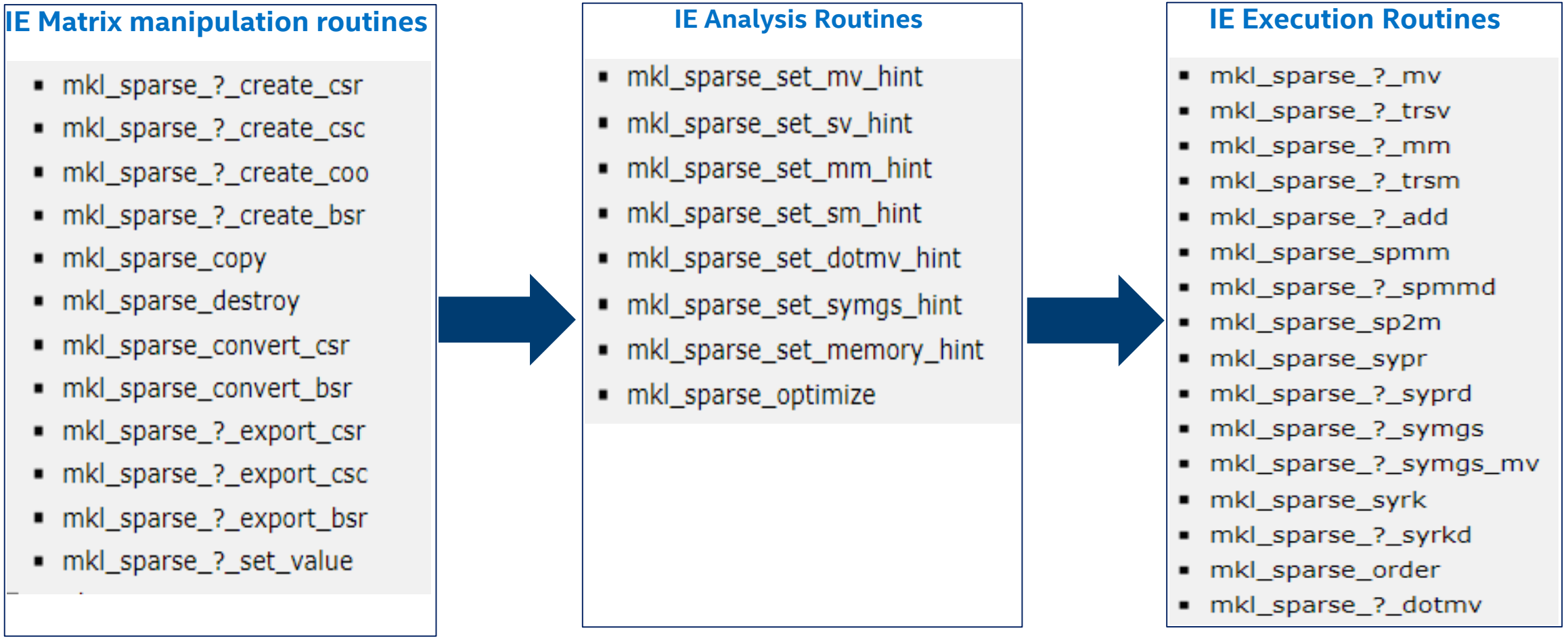
- The first stage constructs the structure of the output matrix.
- The second stage constructs other arrays and performs the desired operation.
- You can **combine** the two stages by performing the entire computation in a single step
- Supported data types – [s,d,c,z]
- Supported matrixes formats – CSR, CSC, COO and BSR

Inspector– Executor Sparse BLAS API, Overview

The Inspector-executor Sparse BLAS routines support the following operations:

- computing the vector product between a sparse matrix and a dense vector:
  ```
  y := alpha*op(A)*x + beta*y
  ```
- solving a single triangular system:
  ```
  y := alpha*inv(op(A))*x
  ```
- computing a product between a sparse matrix and a dense matrix:
  ```
  C := alpha*op(A)*B + beta*C
  ```
- computing a product between sparse matrices with a sparse result:
  ```
  V := alpha*op(A)*op(G)
  ```
- computing a product between sparse matrices with a dense result:
  ```
  C := alpha*op(A)*op(G)
  ```
- computing a sum of sparse matrices with a sparse result:
  ```
  V := alpha*op(A) + G
  ```
- solving a sparse triangular system with multiple right-hand sides:
  ```
  C := alpha*inv(op(A))*B
  ```

Inspector– Executor Sparse BLAS API, Overview



Inspector– Executor Sparse BLAS API, Workflow

Single call

```
mkl_sparse_create_d_csr ( &A,  SPARSE_INDEX_BASE_ZERO,
                    rows, cols,  rowsStart,  rowsEnd,  colIndx,  values );

mkl_sparse_d_mv ( SPARSE_OPERATION_NON_TRANSPOSE,
          alpha, A,  SPARSE_FULL, x,  beta,  y );

mkl_sparse_destroy ( A );
```

Inspector– Executor Sparse BLAS API, Workflow

Iterative method:

```
mkl_sparse_create_d_csr ( &A, SPARSE_INDEX_BASE_ZERO, rows, cols, rowsStart, rowsEnd,
colIndx, values );
mkl_sparse_set_mv_hint ( A, SPARSE_OPERATION_NON_TRANSPOSE, SPARSE_FULL, n_iter );
mkl_sparse_set_memory_hint ( A, SPARSE_MEMORY_AGRESSIVE );
mkl_sparse_optimize ( A );

for (int i=0;i<n_iter;i++) {
      mkl_sparse_d_mv ( SPARSE_OPERATION_NON_TRANSPOSE,  alpha, A, SPARSE_FULL, x, beta,
      y  );
      …
}
mkl_sparse_destroy( A );
```

Inspector– Executor Sparse BLAS API - DEMO

- untar the SpMV.tar: **tar -xvf SpMV.tar**
- IE_SpMV_bechmark.cpp - **review the code**
  - **Open and read** the input file (mtx)
  - **Memory allocation:** av,ay,ax - coordinate format format
  - **Conversion to CSR** : **mkl_dcsrcoo** (job,&n,**a,ja,ia**,&nnz,av,ay,ax,&info);
  - **mkl_dcsrgemv** (&trans, &n, **a**, ia, ja, **x**, **b**); // m-v call
  - **IE API:** create_csr, set_mv_hint, set_memory_hint, optimize.
  - **IE API:** mkl_sparse_d_mv (SPARSE_OPERATION_NON_TRANSPOSE, 1.0, csrA, descr, **x**, 0.0, **bs**);
  - Results validation eps = sum**|b$_i$ – bs$_i$|**

Inspector– Executor Sparse BLAS API - DEMO

- **module load intel64/19.1up01**
- **Compiling :**

  icc -mkl IE_SpMV_bechmark.cpp

  - “warning #1478: function "mkl_dcsrgemv" (declared at line 164 of "/opt/intel/compilers_and_libraries_2020.1.217/linux/mkl/include/mkl_spblas.h") was declared deprecated”
- **How to run:**

./a.out ASIC_100k/ASIC_100k.mtx

Output:

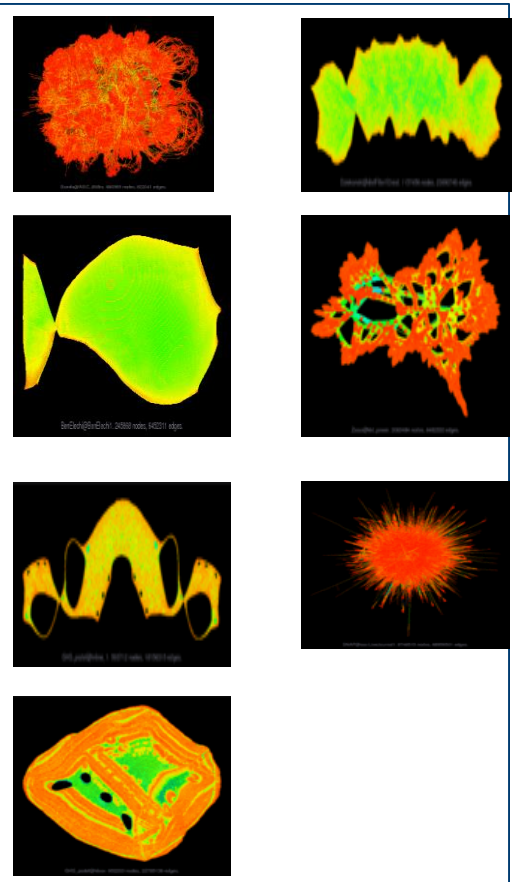
```
.... matrix name = ASIC_100k/ASIC_100k.mtx .....
.... MKL ....
Major version:        2020
Minor version:           0
Update version:          1

Product status:     Product
Platform:           Intel(R) 64 architecture Processor optimization:
Intel(R) Advanced Vector Extensions 512 (Intel(R) AVX-512)
enabled processors

SIZE == 99340, NNZ ==954163

....relative  error == 1.363714e-14
....old API == 2.411947e-04
....new API == 1.011804e-04
....ratio = 2.383809e+00
```

*** - Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz ,192 GB RAM**

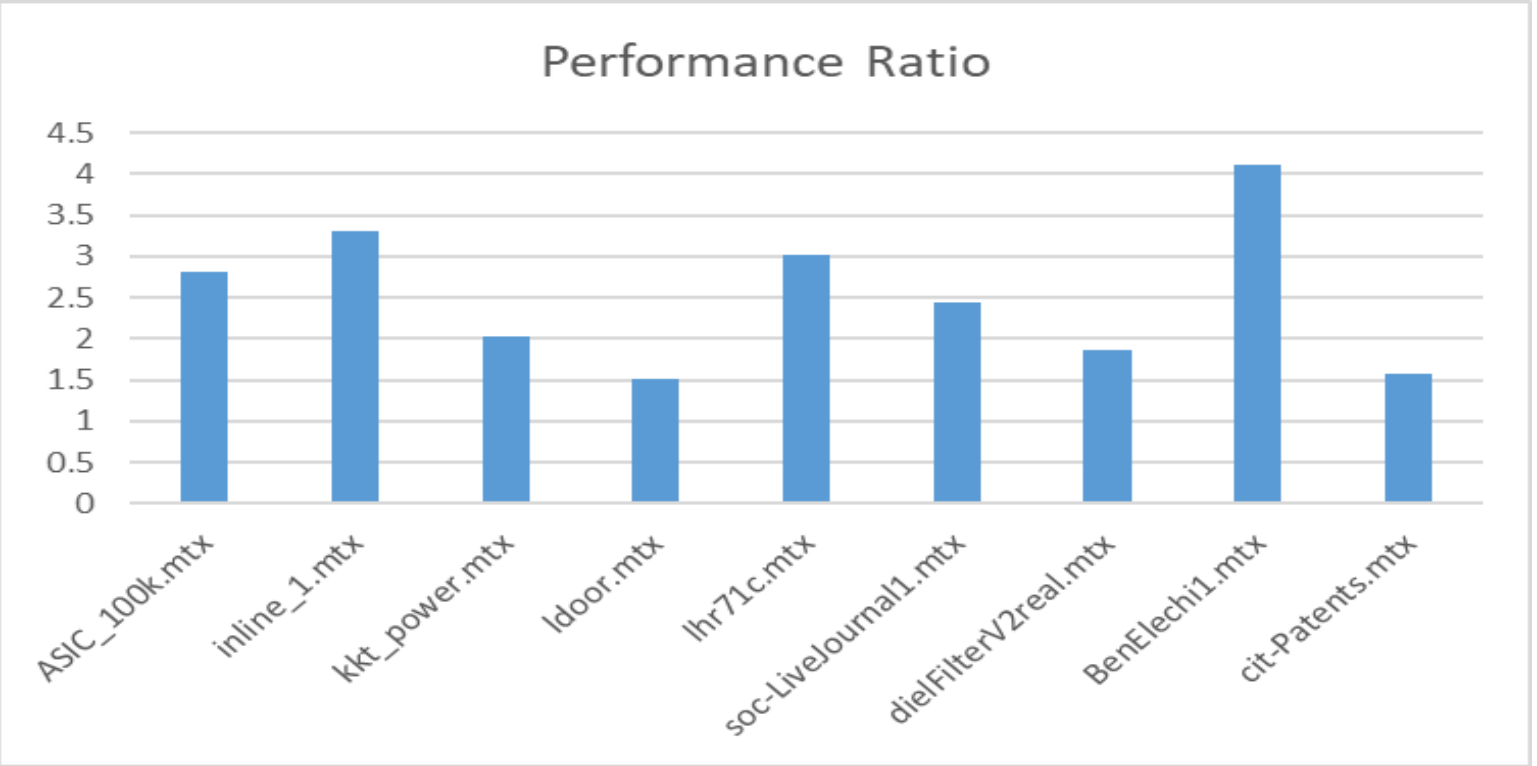
Inspector– Executor Sparse BLAS API

Testing Examples

Florida Collection Suite:			
http://www.cise.ufl.edu/research/sparse/matrices/			
Name	**Dim***	**#nnz**	**Description**
ASIC_100k	99340	954163	circuit simulation problem
BenElechi1	245874	6698185	2D/3D problem
inline_1	503712	18660027	structural problem
ldoor	952203	23737339	structural problem
dielFilterV2real	1157456	24828204	electromagnetics problem
kkt_power	2063494	8130343	optimization problem
soc-LiveJournal1	4847571	68993773	directed graph



IE Sparse BLAS API Performance



Configuration Info – SW Versions: Intel® Math Kernel Library (**Intel® MKL) 2020 u1**. Hardware: **Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz ,192 GB RAM** (12x16GB DDR4-2666). Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

IE Sparse BLAS – Threading, OMP

- icc -mkl **SpMV_test.cpp**
- ./a.out ASIC_100k/ASIC_100k.mtx

  export MKL_NUM_THREADS=**1, 2, 4, 8, 16, 32**

  ./run_threading.sh

  Do you see smth like …

```
ExecTime = 0.001068, #threads = 1
ExecTime = 0.000526, #threads = 2
ExecTime = 0.000262, #threads = 4
ExecTime = 0.000177, #threads = 8
ExecTime = 0.000134, #threads = 16
ExecTime = 0.000130, #threads = 32
```

Intel MKL Resources

Intel® MKL website:

– https://software.intel.com/en-us/intel-mkl

Intel MKL forum:

– https://software.intel.com/en-us/forums/intel-math-kernel-library

Intel® MKL benchmarks:

– https://software.intel.com/en-us/intel-mkl/benchmarks#

Intel® MKL link line advisor:

– http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor/

Legal Disclaimer & Optimization Notice

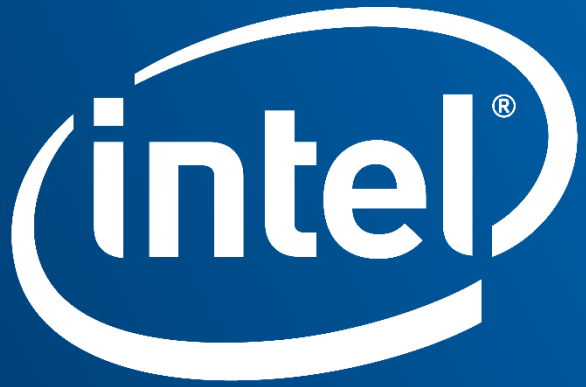
INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804



Software