INTEL® MATH KERNEL LIBRARY (INTEL® MKL) SMALL MATRIX MULTIPLICATION OPTIMIZATIONS USING JIT COMPILATION

Arthur Mitrano, Sarah Knepper, Louise Huot, Kazushige Goto, Peter Caday, Mesut Meterelliyoz, Shane Story

17 September 2018, BLIS RETREAT 2018

Outline

- Problem statement and solutions
- Simple example
- Performance comparison

Overheads for small sizes

- Low vectorization
- Low parallelization
- Non-local data access for large leading dimensions
- Error checking
- High function call overheads
  - Dispatching to ISA-specific codepath

Methods for improving performance for small sizes

- Specific kernels
- Compile-time optimizations
- Just-in-time (run-time) compilation
- Batching operations together
  - Modifying data layout

Direct call compilation flags for Intel MKL

Define the preprocessor macro `MKL_DIRECT_CALL` or `MKL_DIRECT_CALL_SEQ`

- Instead of calling a library function, a C implementation may be used
- Starting from Intel MKL 2018.1, compiler intrinsics may be used for some kernels

Starting from Intel MKL 2019 Beta: `MKL_DIRECT_CALL_JIT` or `MKL_DIRECT_CALL_SEQ_JIT`

- A JIT-ted kernel may be used

```
// compile with: icc -DMKL_DIRECT_CALL …
#include <mkl.h>
void main(void) {
    dgemm(…);
}
```

```
! compile with: ifort -DMKL_DIRECT_CALL -fpp …
#     include “mkl_direct_call.fi”
      program DGEMM_MAIN
      DGEMM(…)
```

Intel MKL JIT API

```
// Declare variables and initialize data (not shown)
void *jitter;
// Create jitter handle and generate GEMM kernel
mkl_jit_status_t status = mkl_jit_create_sgemm(&jitter, layout, transA,
         transB, m, n, k, alpha, lda, ldb, beta, ldc);
// Check that creation was successful
if (MKL_JIT_ERROR == status) {
    printf("Error: cannot create jitter\n");
    return 1;
}
// Get kernel associated with jitter handle
sgemm_jit_kernel_t kernel = mkl_jit_get_sgemm_ptr(jitter);
for (i = 0; i < nb; i++) {
    …
    kernel(jitter, a[i], b[i], c[i]); // Repeatedly execute the GEMM kernel
    …
}
mkl_jit_destroy(jitter); // Destroy the created jitter/GEMM kernel
```

Intel MKL JIT API

- Creates a handle on a jitter for sgemm

```
mkl_jit_status_t mkl_jit_create_sgemm(void **jitter, <sgemm paramters>)
```

- MKL_JIT_SUCCESS: indicates that a sgemm kernel has been generated;
  MKL_NO_JIT   : indicates standard sgemm function will be used;
  MKL_JIT_ERROR   : indicates an error happened due to lack of memory.

- Returns a function pointer to generated sgemm kernel

```
sgemm_jit_kernel_t mkl_jit_get_sgemm_ptr(const void *jitter)
```

- `typedef void (*sgemm_jit_kernel_t)(void *, float *, float *, float *)`

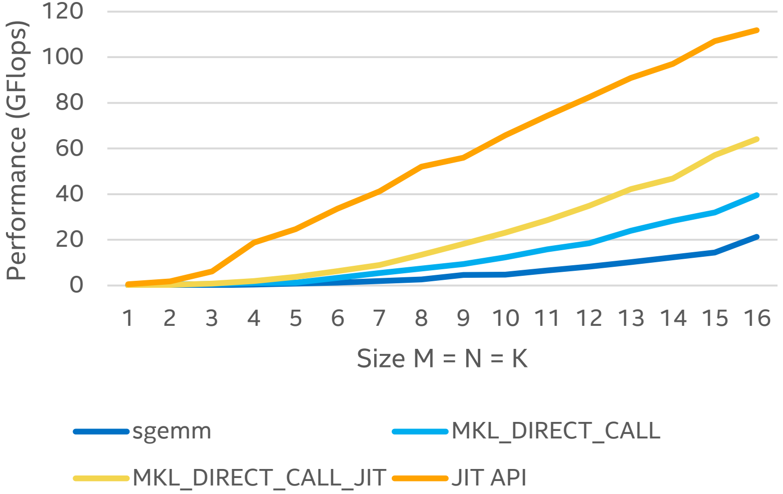
- Free the memory associated with code generator and sgemm kernel

```
mkl_jit_status_t mkl_jit_destroy(void *jitter)
```

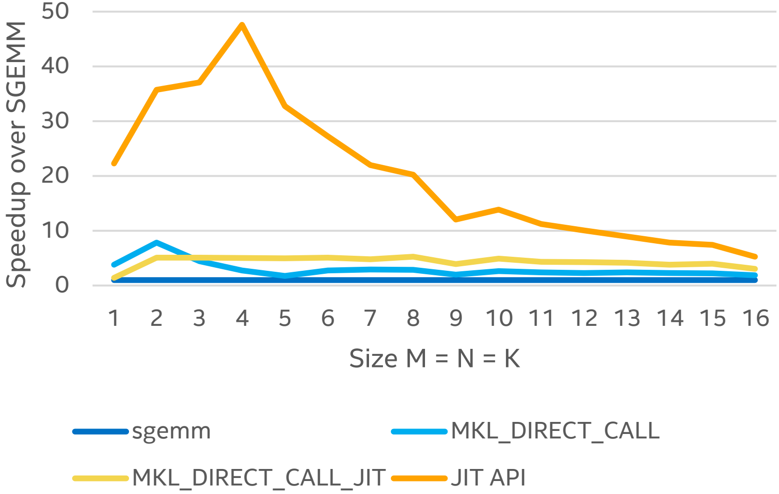
Optimization Notice

Performance of SGEMM on Intel® Xeon® Platinum

Intel® MKL 2019 Gold Performance of SGEMM and Variants



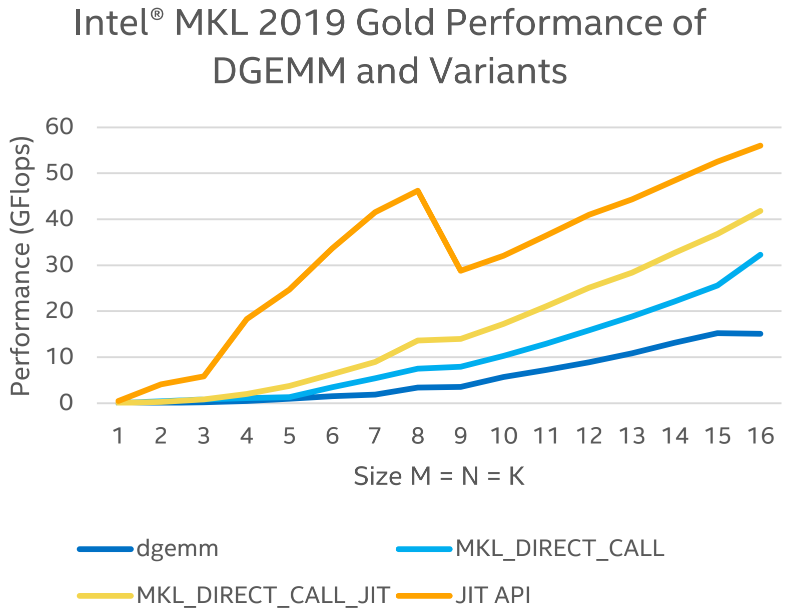
Intel® MKL 2019 Gold Speedup over SGEMM

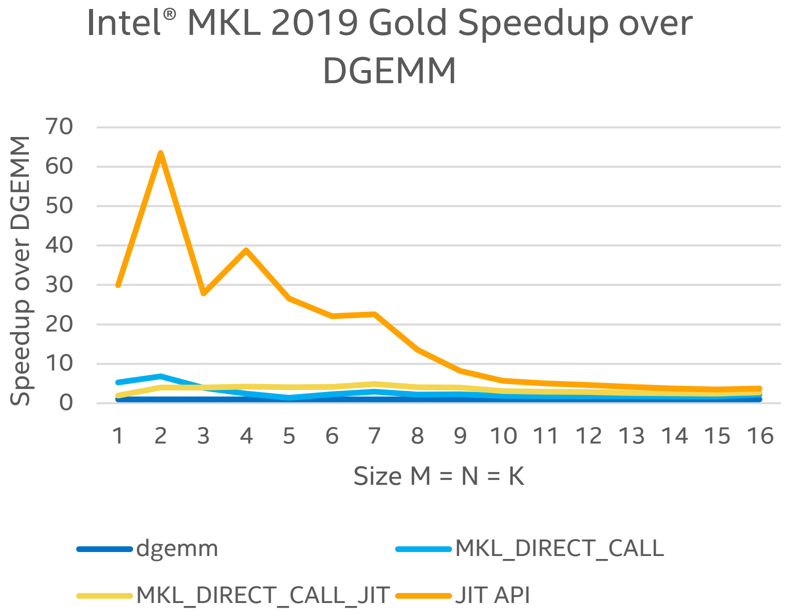


Configuration: Intel® Xeon® Platinum 8180, 2x28 cores, 2.5 GHz, 376 GB RAM, OS Ubuntu, 16.04 LTS; Intel® MKL 2018.
Performance results may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure. Software and workloads used in 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. For more complete information visit www.intel.com/benchmarks. 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

Performance of DGEMM on Intel® Xeon® Platinum





Configuration: Intel® Xeon® Platinum 8180, 2x28 cores, 2.5 GHz, 376 GB RAM, OS Ubuntu, 16.04 LTS; Intel® MKL 2018.
Performance results may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure. 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. For more complete information visit www.intel.com/benchmarks. 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

Summary

- Small linear algebra problems:
  - Are ubiquitous
  - Suffer performance overheads
- Just-in-time compilation can help:
  - Generator can create customized kernels for any parameters
- Performance gains can be significant

Resources

- Intel MKL Developer Reference: https://software.intel.com/en-us/articles/mkl-reference-manual
- Intel MKL Forum: https://software.intel.com/en-us/forums/intel-math-kernel-library
- No cost option for Intel MKL: https://software.intel.com/en-us/articles/free-mkl
- Intel MKL-DNN: https://github.com/01org/mkl-dnn
- Xbyak: https://github.com/herumi/xbyak
- libxsmm: https://github.com/hfp/libxsmm

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.

Performance results may not reflect all publicly available security updates. See configuration disclosure for details. 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

intel®