Packing - the next BLIS Frontier?

Tze Meng Low

BLIS Retreat 2017

Looking back

- Original Goals
  The BLAS-like Library Instantiation Software (BLIS) framework is a new infrastructure for rapidly instantiating Basic Linear Algebra Subprograms (BLAS) functionality. ....
- Increase Productivity
- Extensible

Etc...

Ease of use

- What happens with a new architecture?
  - Write/Generate micro-kernel

  - Use analytical modeling for parameters
  \[
  C_A \leq \left[ \frac{m_r}{m_r + n_r} (W_{L1} - 1) \right] \\
  k_c = \frac{C_{A_r} N_{L1} C_{L1}}{m_r S_{data}}
  \]

Ease of use

- Actual work
  - Create directory (recursive copy and rename)
  - Edit `bli_kernel.h`
  - Drop your kernel(s) into `kernels/3`
  - `./configure new_kernel & make install`
Unexpected benefits

- More than just BLAS

**Linkage Disequilibrium**

<table>
<thead>
<tr>
<th>SNP A</th>
<th>SNP B</th>
</tr>
</thead>
<tbody>
<tr>
<td>01011...</td>
<td>010011...</td>
</tr>
</tbody>
</table>

**Comm**

- 010001 → 2

N. Alachiotti, Popovic, T.M. Low, Efficient Computation of Linkage Disequilibria as Dense Linear Algebra Operations, HiCOMB 2016

**What everyone is saying...**

PACKING
Reasons for packing

- Work with different data layouts
- Hide additional operations
  \[ C = f \odot ((A \otimes B) \oplus C) \]
- Generalize the framework

Work with Different Data Layout

- Limited SIMD support for complex multiplication
- Interleaved vs Non-interleaved
- Switch format during packing

Performance on Intel Kaby Lake 7770K

D. Popovici, F. Franchetti, T.M. Low, Mixed Data Format for Vectorized Complex Kernel, HPEC, 2017
Hide additional operations

- Finite Field over large primes
  - Reuse BLAS & LAPACK for finite fields

J. Johnson, T.M. Low, M. Lambert, P. Oostema, B. D. Saunders, High-Performance Kernels for Exact Linear Algebra, ACA 2017
Generalize the framework

- Finite fields for small primes
  - 4 Russians Method

<table>
<thead>
<tr>
<th>Table Creation</th>
<th>Compute with A as index</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>(1 1 0 1)</td>
</tr>
<tr>
<td></td>
<td>(0 0 0 0)</td>
</tr>
<tr>
<td></td>
<td>(1 0 1 0)</td>
</tr>
<tr>
<td></td>
<td>(0 1 1 1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T0</th>
<th>T1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>0 0 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>1 0 1 0</td>
<td>1 0 1 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0 0 0 1)</td>
</tr>
<tr>
<td>(0 1 0 0)</td>
</tr>
<tr>
<td>(0 0 0 0)</td>
</tr>
<tr>
<td>(1 1 0 1)</td>
</tr>
</tbody>
</table>

J. Johnson, T.M. Low, M. Lambert, P. Oostema, B. D. Saunders, High-Performance Kernels for Exact Linear Algebra, ACA 2017

Performance

Bits Ops / Cycle

- 4R - BLIS
- m4ri (4 bit tables)
- O(n^3)
- Naive Peak
- 4R - Peak (4 bit tables)

Need Packing for A

N = M = K

J. Johnson, T.M. Low, M. Lambert, P. Oostema, B. D. Saunders, High-Performance Kernels for Exact Linear Algebra, ACA 2017
What everyone is NOT saying...

PACKING

Remember how easy to add kernels?

Ease of use

- Actual work
  - Create directory (recursive copy and rename)
  - Edit bll_kernel.h
  - Drop your kernel(s) into kernels/3
  - ./configure new_kernel & make install
Introducing packing is not easy

- Files edited or added

  frame/1m/bli_l1m_ft.h
  frame/1m/bli_l1m_ker
  frame/1m/packm/bli_packm.h
  frame/1m/packm/bli_packm_cxk.c
  frame/include/bli_kernel_macro_defs.h
  frame/include/bli_kernel_pre_macros.h
  frame/include/bli_kernel_prototypes.h
  frame/ind/include/bli_kernel_ind_macro_defs.h
  frame/3/gemm/bli_gemm_cntl.c

  frame/1m/packm/bli_packm_blk_var2.c
  frame/1m/packm/bli_packm_blk_var2.h
  frame/1m/packm/bli_packm_cxk_cmu.c
  frame/1m/packm/bli_packm_cxk_cmu.h
  frame/1m/packm/bli_packm_struc_cxk_cmu.c
  frame/1m/packm/bli_packm_struc_cxk_cmu.h
  frame/1m/packm/ukernels/bli_packm_cxk_cmu_ref.c
  frame/1m/packm/ukernels/bli_packm_cxk_cmu_ref.h
  frame/ind/include/bli_packm_cmu_macro_defs.h

Calling sequence
# My wish list

- As easy as adding kernels
- Files to edit or add
  
  ```
  frame/1m/bli_1m_ft.h
  frame/1m/bli_1m_ker.h
  frame/1m/packm/bli_packm.h
  frame/1m/packm/bli_packm_cxk.c
  frame/include/bli_kernel_macro_defs.h
  frame/include/bli_kernel_pre_macro_defs.h
  frame/include/bli_kernel_prototypes.h
  frame/ind/include/bli_kernel_ind_macro_defs.h
  frame/ind/include/bli_packm_ind_pre_macro_defs.h
  frame/3/gemm/bli_gemm_cntl.c
  frame/1m/packm/bli_packm_blk_var2.c
  frame/1m/packm/bli_packm_blk_var2.h
  frame/1m/packm/bli_packm_cxk_cmu.c
  frame/1m/packm/bli_packm_cxk_cmu.h
  frame/1m/packm/bli_packm_struc_cxk_cmu.h
  frame/1m/packm/ukernels/bli_packm_cxk_cmu_ref.c
  frame/1m/packm/ukernels/bli_packm_cxk_cmu_ref.h
  frame/ind/include/bli_packm_cmu_macro_defs.h
  ```

- Move from frame to kernel

---

# Summary

- Lighter and more nimble framework
- Refactoring of the packing is proposed
- Inputs from BLIS user and developers wanted
Discussion