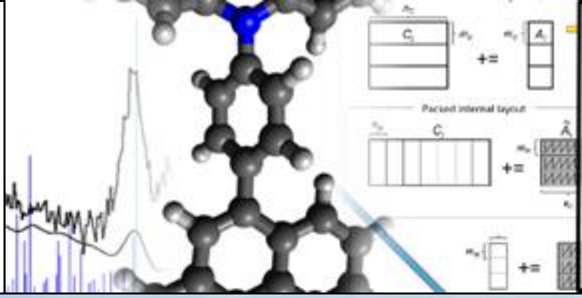
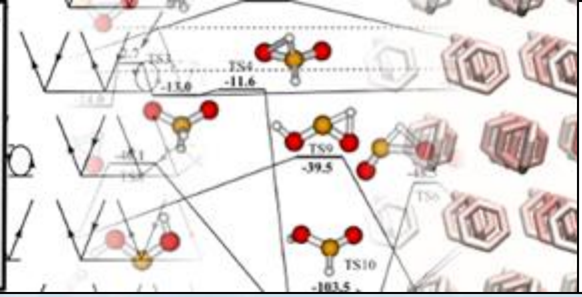
BLIS Day the 13th:

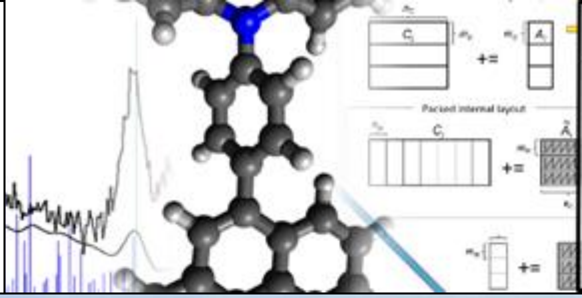
BLIS out of BLAS

Devin Matthews, Southern Methodist University

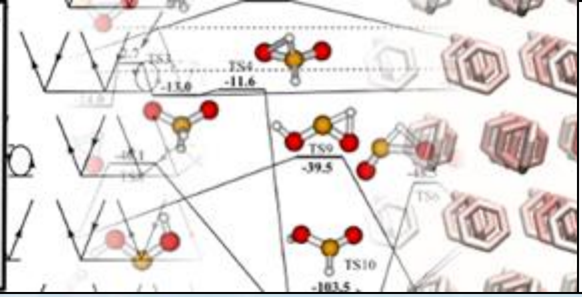


MATTHEWS
RESEARCH
GROUP





MATTHEWS
RESEARCH
GROUP



ORACLE

Outline

- **Part I:** BLIS Yesterday and Today
- **Part II:** TBLIS
- **Part III:** Revisiting libflame


DGEMM

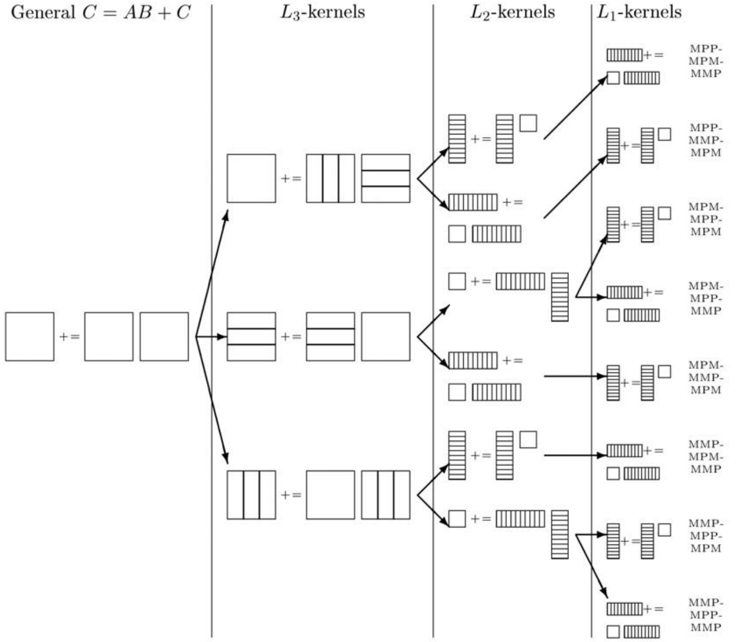


Figure 3: Possible algorithms for matrices in memory level L_3 given all L_2-kernels.

ITX-GEMM

John Gunnels (UT, now NVIDIA)

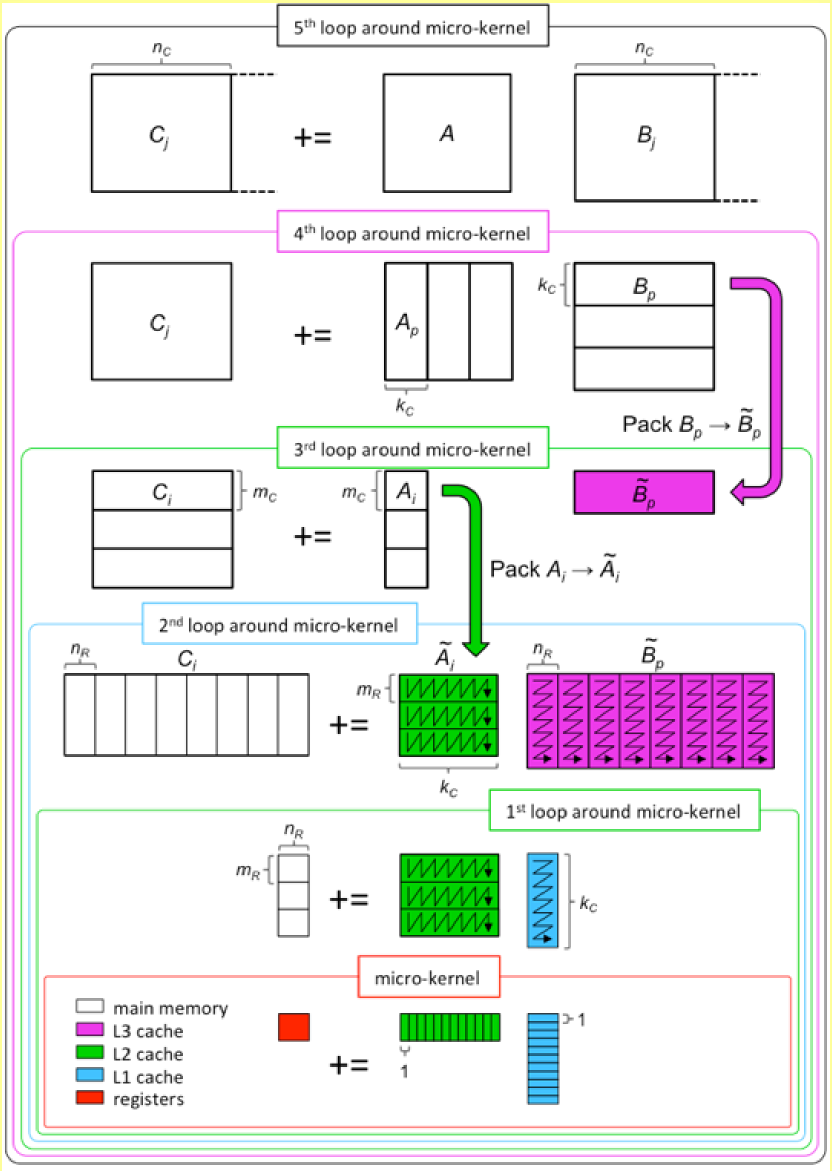
Greg Henry (Intel, now NVIDIA)

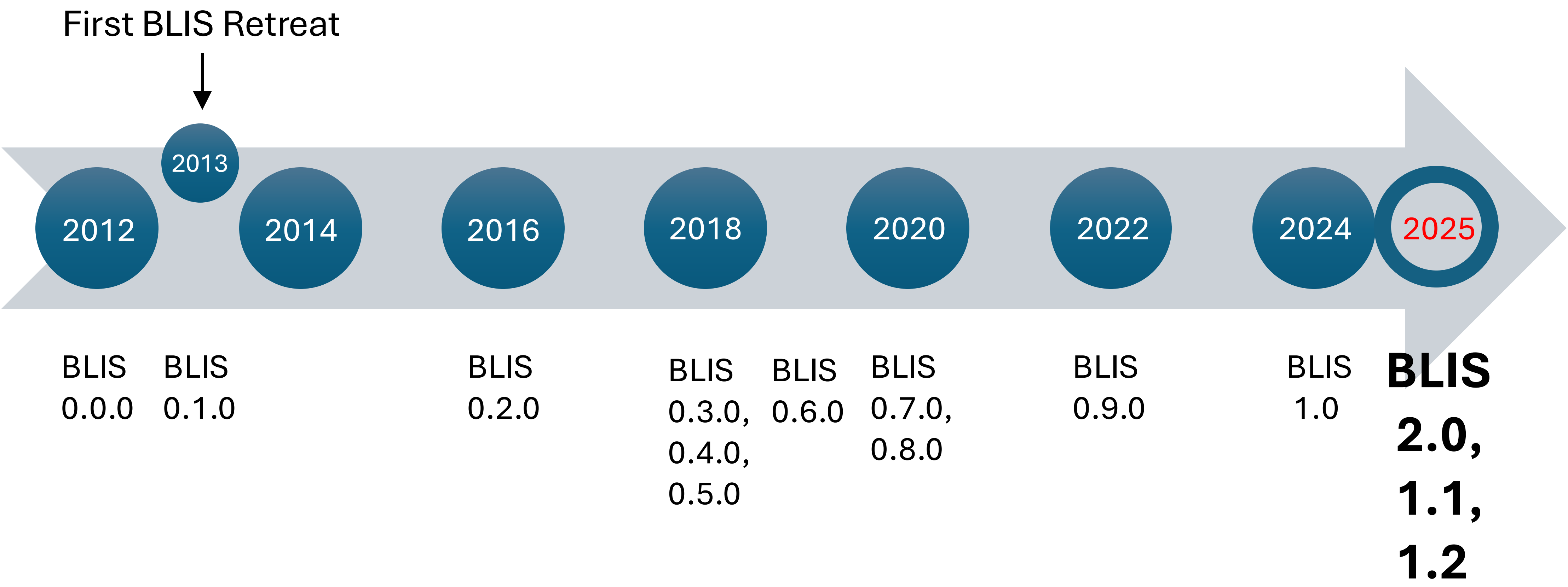
High-performance implementation of the level-3 BLAS (GotoBLAS)

Authors: Kazushige Goto, Robert Van De Geijn Authors Info & Claims

ACM Transactions on Mathematical Software, Volume 35, Issue 1 • Article No.: 4, pp 1–14 • https://doi.org/10.1145/1377603.1377607

Published: 25 July 2008 Publication History


5th loop around micro-kernel
n_C
C_j
+=
A
B_j
4th loop around micro-kernel
C_j
A_p
k_C
B_p
Pack $B_p \rightarrow \tilde{B}_p$
3rd loop around micro-kernel
C_i
m_C
A_i
Pack $A_i \rightarrow \tilde{A}_i$
$\tilde{B}_p$
2nd loop around micro-kernel
n_R
m_R
$\tilde{A}_i$
1st loop around micro-kernel
micro-kernel
main memory
L3 cache
L2 cache
L1 cache
registers
1


First BLIS Retreat
2012
2013
2014
2016
2018
2020
2022
2024
2025
BLIS 0.0.0
BLIS 0.1.0
BLIS 0.2.0
BLIS 0.3.0, 0.4.0, 0.5.0
BLIS 0.6.0
BLIS 0.7.0, 0.8.0
BLIS 0.9.0
BLIS 1.0
BLIS 2.0, 1.1, 1.2

BLIS is a **Community Project**

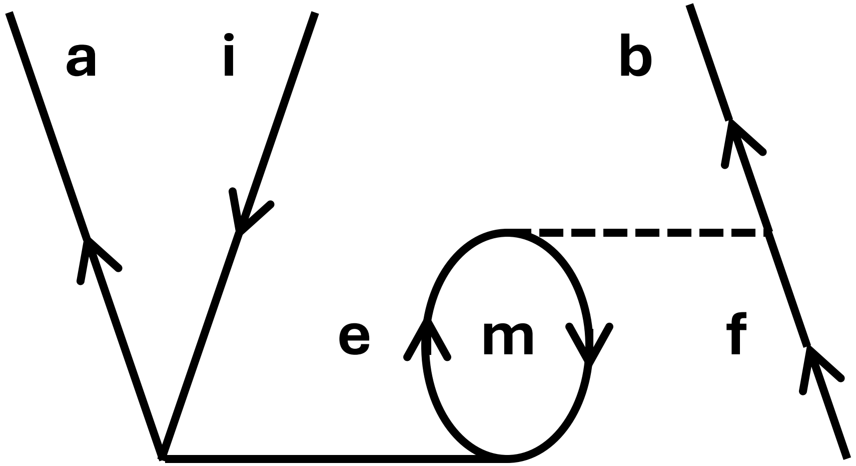
Bryan Marker

141+ contributors

AMD
ARM
ORACLE
NATIONAL INSTRUMENTS
hp
Microsoft
NEC
NASA
facebook
MathWorks
IBM
HUAWEI
intel
NSF
Qualcomm
NVIDIA
SHPC
ODEN INSTITUTE FOR COMPUTATIONAL ENGINEERING & SCIENCES

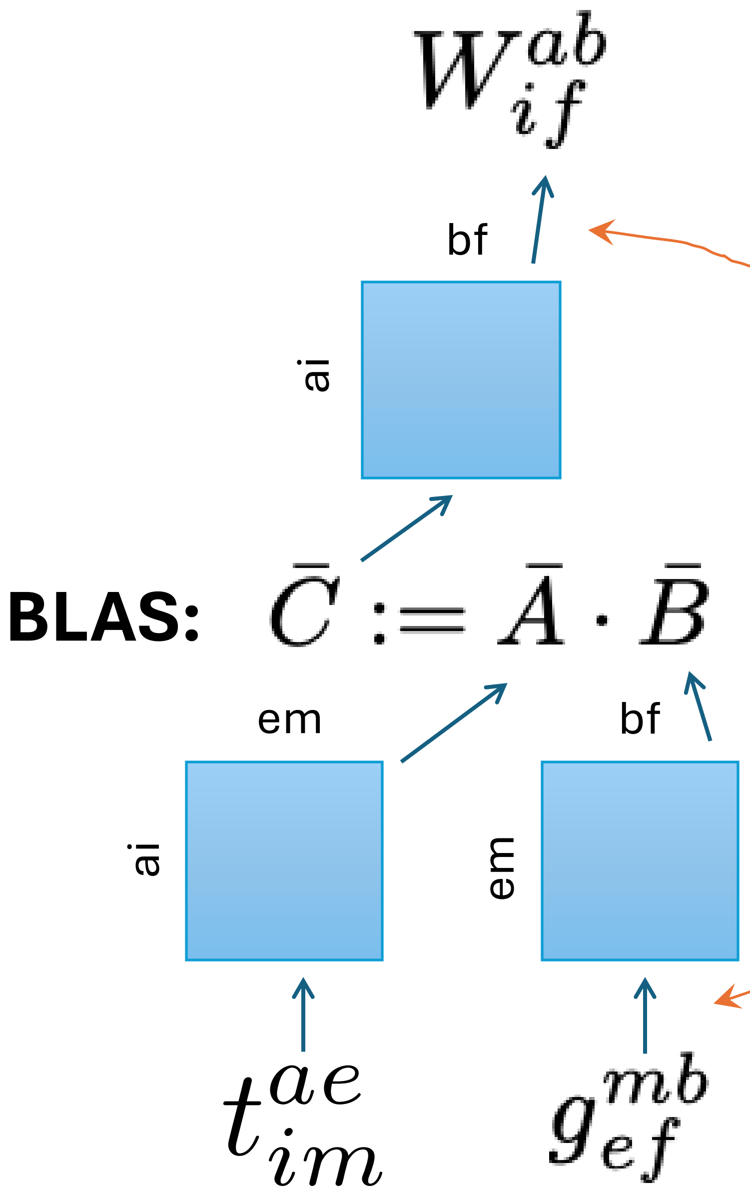
The Tensor-Based Library Instantiation Software (TBLIS)

a.k.a. “Tensor BLIS”



$$W^{ab}_{if} \leftarrow 2t^{ae}_{im}g^{mb}_{ef}$$

```
tblis::mult(2.0, t2, "aeim",
            g, "mbef",
            1.0,  W, "abif");
```

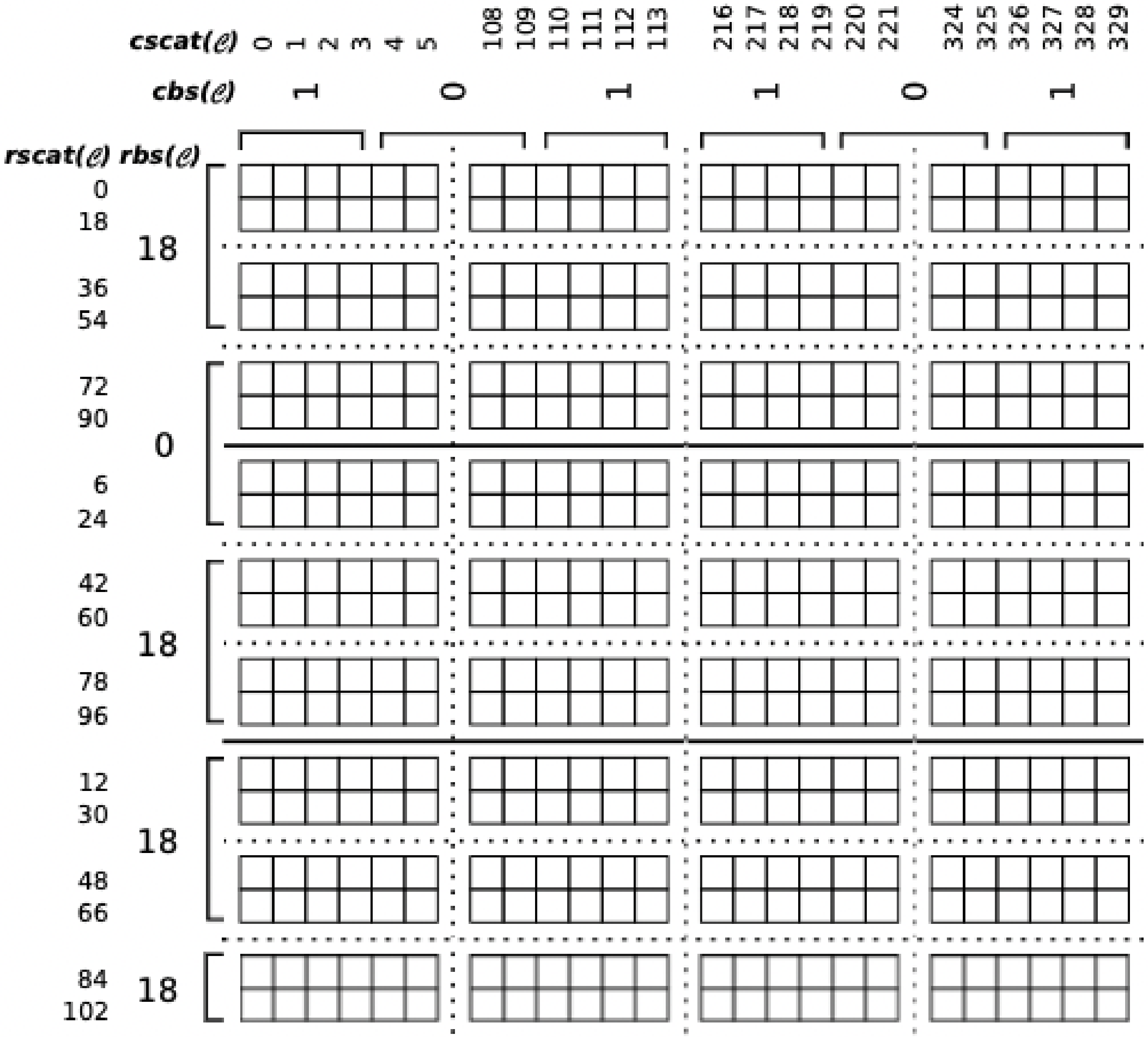


Costly **tensor transpose** (permute) operations

"Transpose-Transpose-GEMM-Transpose" (TTGT)

$$\mathscr{C}_{abif} := \mathscr{A}_{aeim} \cdot \mathscr{B}_{mbej}$$

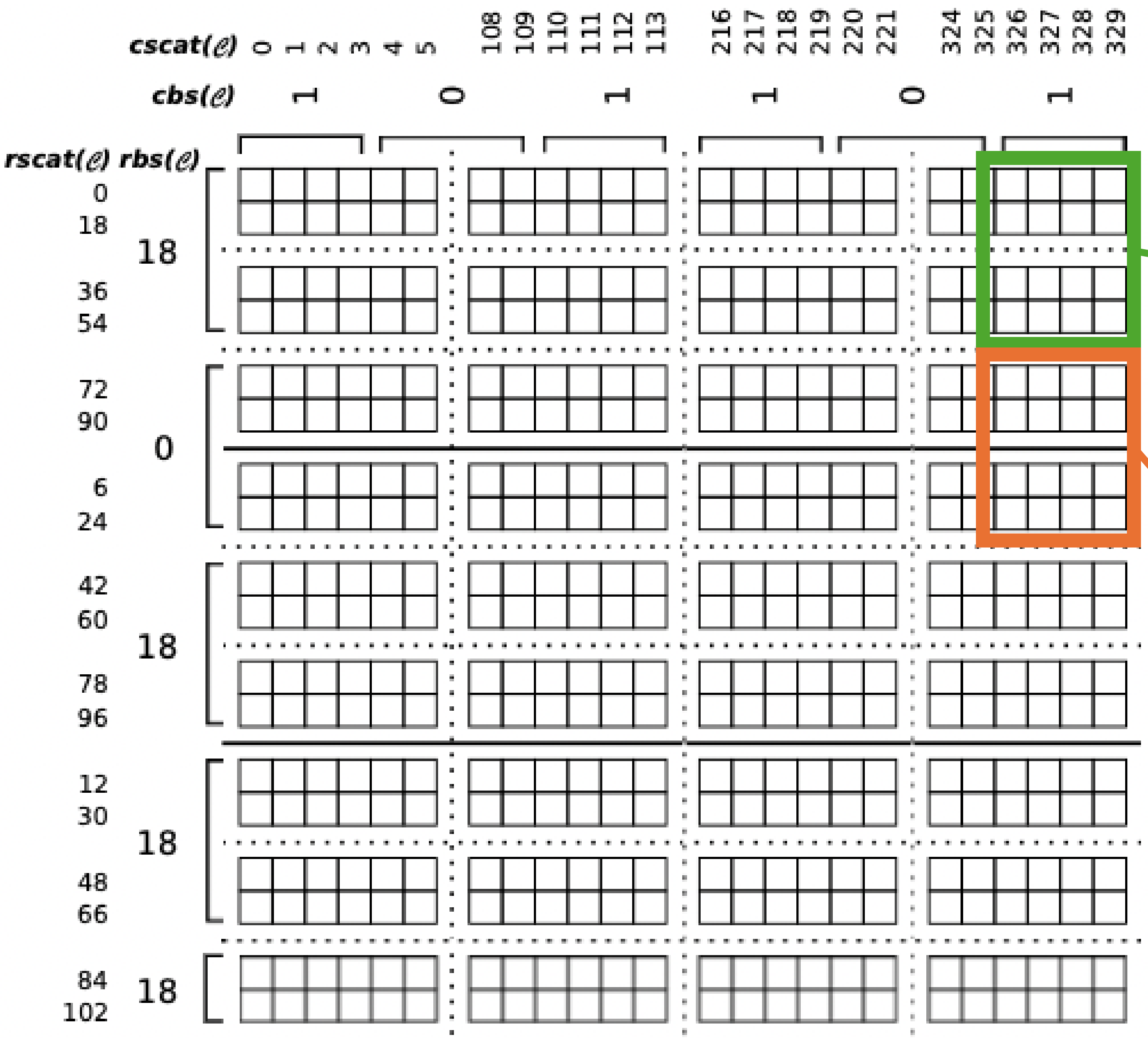
$$\tilde{C}_{IJ} := \tilde{A}_{IP} \cdot \tilde{B}_{PJ}$$



“Block-Scatter Matrix Tensor Contraction” (BSMTC)

$$\mathscr{C}_{abif} := \mathscr{A}_{aeim} \cdot \mathscr{B}_{mbej}$$

$$\tilde{C}_{\bar{I}\bar{J}} := \tilde{A}_{\bar{I}\bar{P}} \cdot \tilde{B}_{\bar{P}\bar{J}}$$



Well-defined row and column strides:

Use **standard matrix-multiply microkernel**

Irregular "jump" along rows and/or columns:

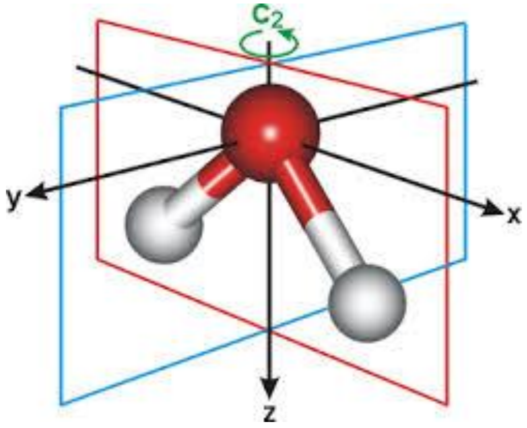
Write from matrix-multiply microkernel into temporary buffer, then write back using **scatter vectors**.

TBLIS 2.0

- Uses a **BLIS plugin** to leverage optimized kernels across x86_64 (AMD, Intel), ARM, IBM, etc.
- Significantly improved complex number support (1m).
- New C++ API which supports various front-ends/tensor manipulation libraries (e.g. MArray, Eigen::Tensor, xtensor, Armadillo, std::mdspan). Compatibility mode for TBLIS 1.x is supported.
- New CMake build system makes integration and packaging easier.
- C++20 (☺/☹)

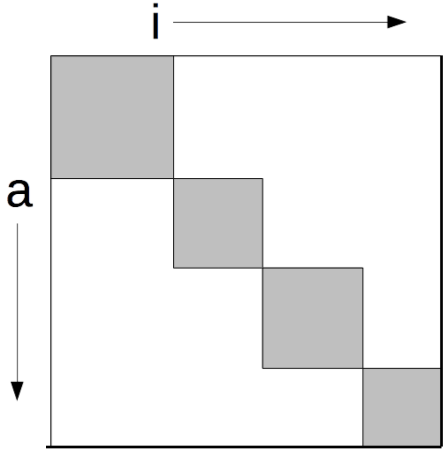
SCAN ME

The Direct Product Decomposition



$$T_i^a \longrightarrow$$

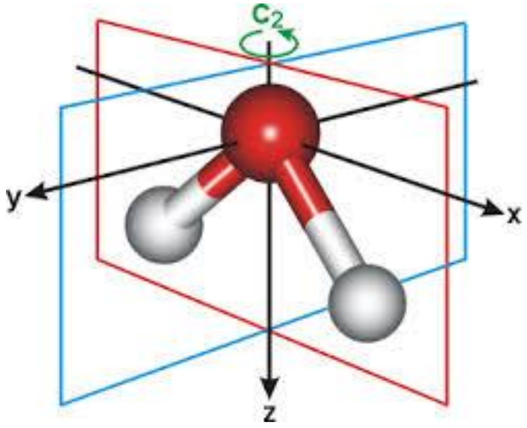
$$\Gamma_a \otimes \Gamma_i = \Gamma_T$$



Point Group Symmetry

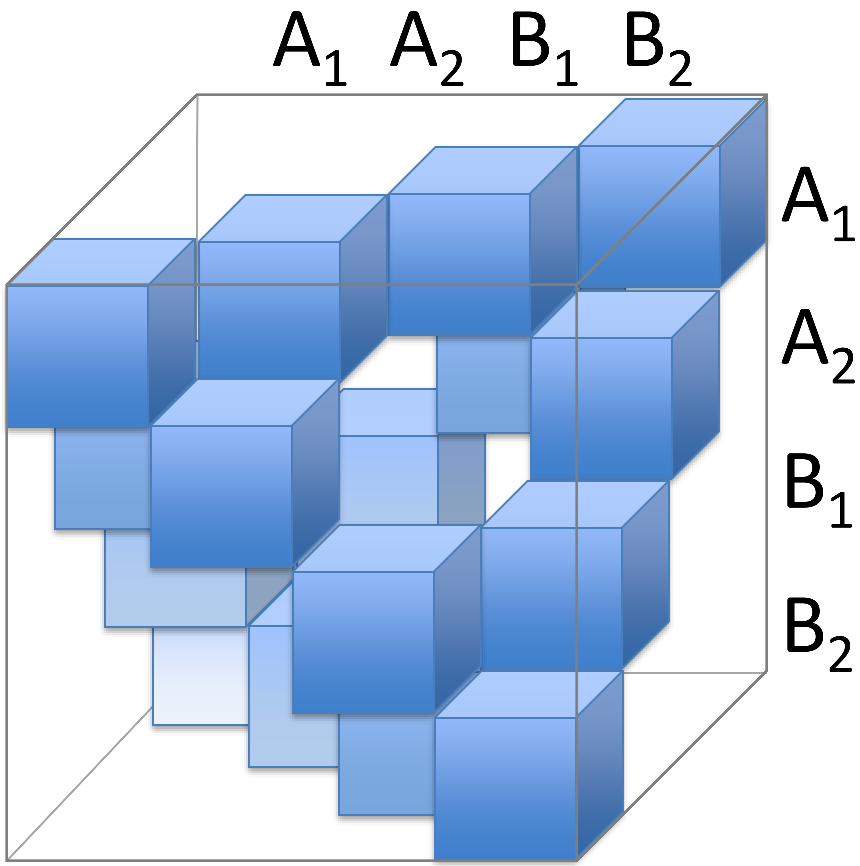
Cost savings proportional to g^2 (g = number of irreducible representations/blocks).

The Direct Product Decomposition

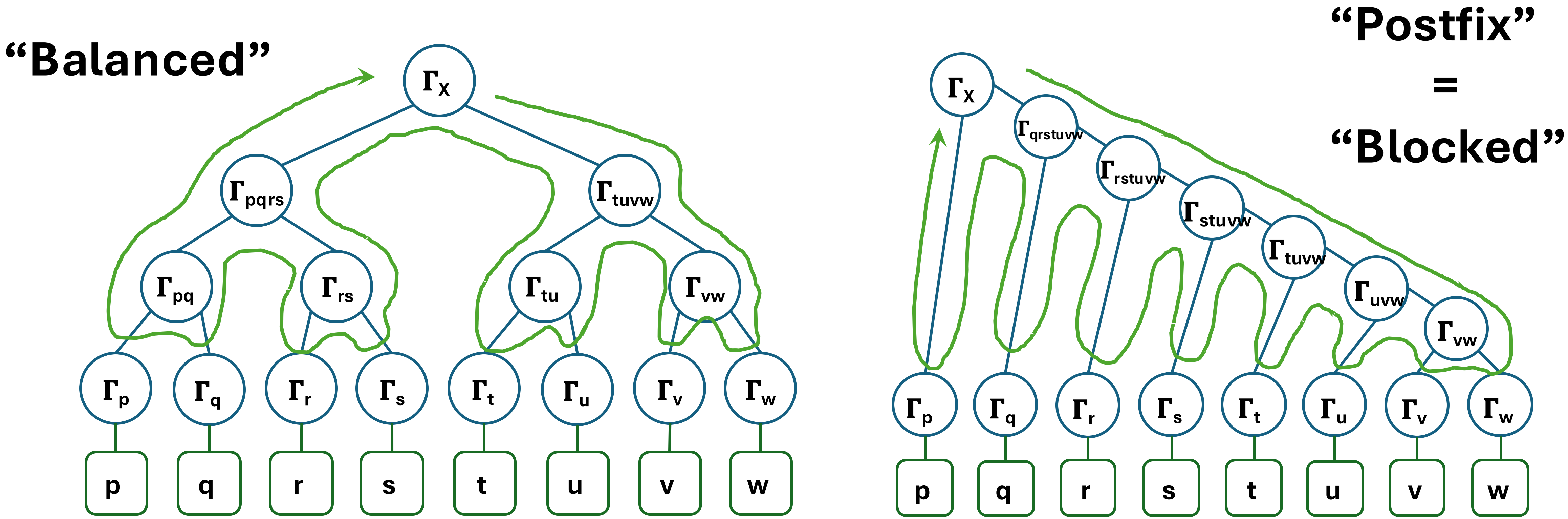


$$Z_{abc} \longrightarrow$$

$$\Gamma_a \otimes \Gamma_b \otimes \Gamma_c = \Gamma_Z$$

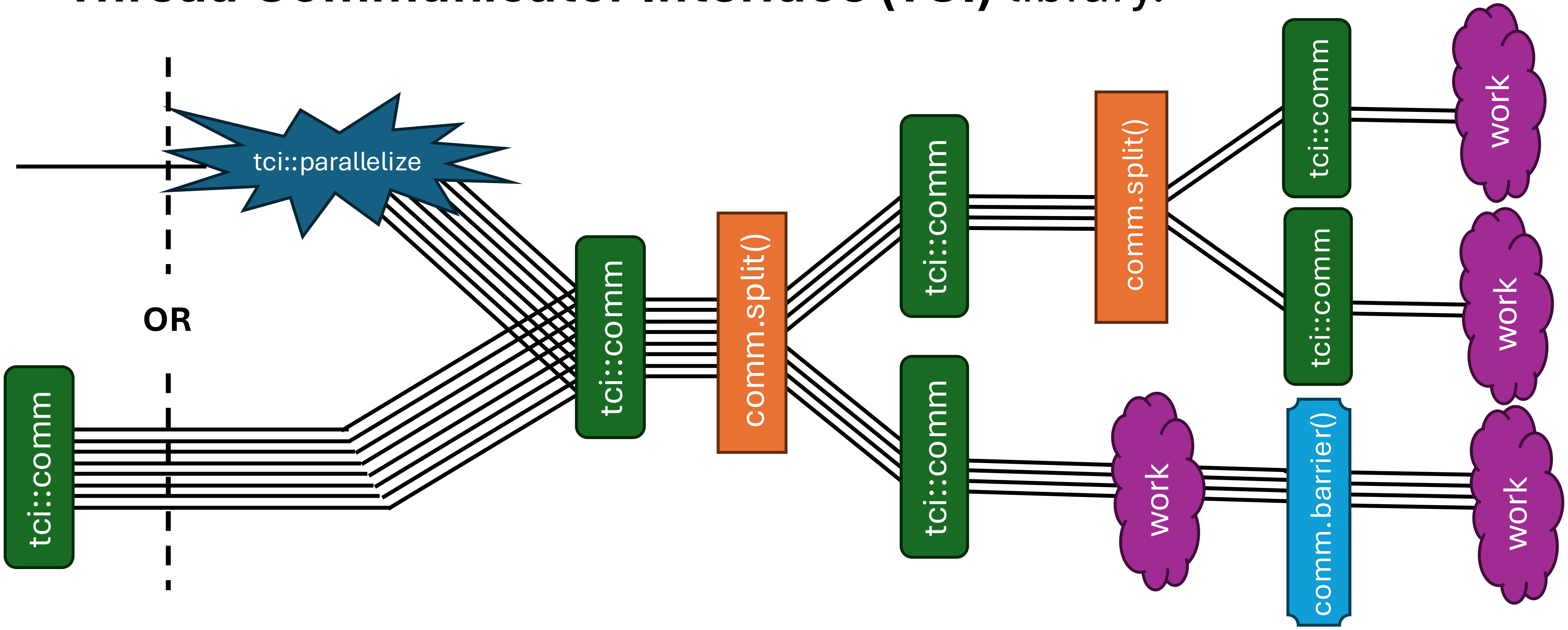


The Direct Product Decomposition



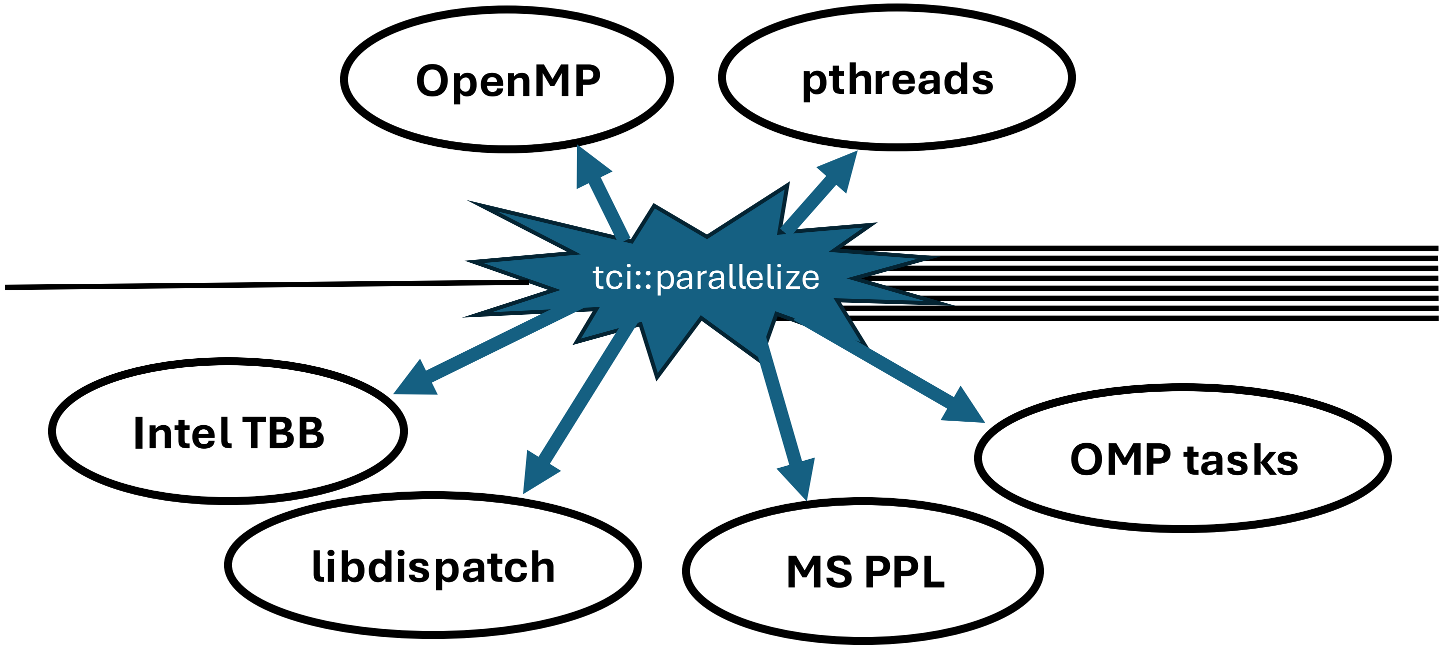
Multi-threading

- Threading over multiple cores (up to 100s) uses the external **Thread Communicator Interface (TCI)** library.



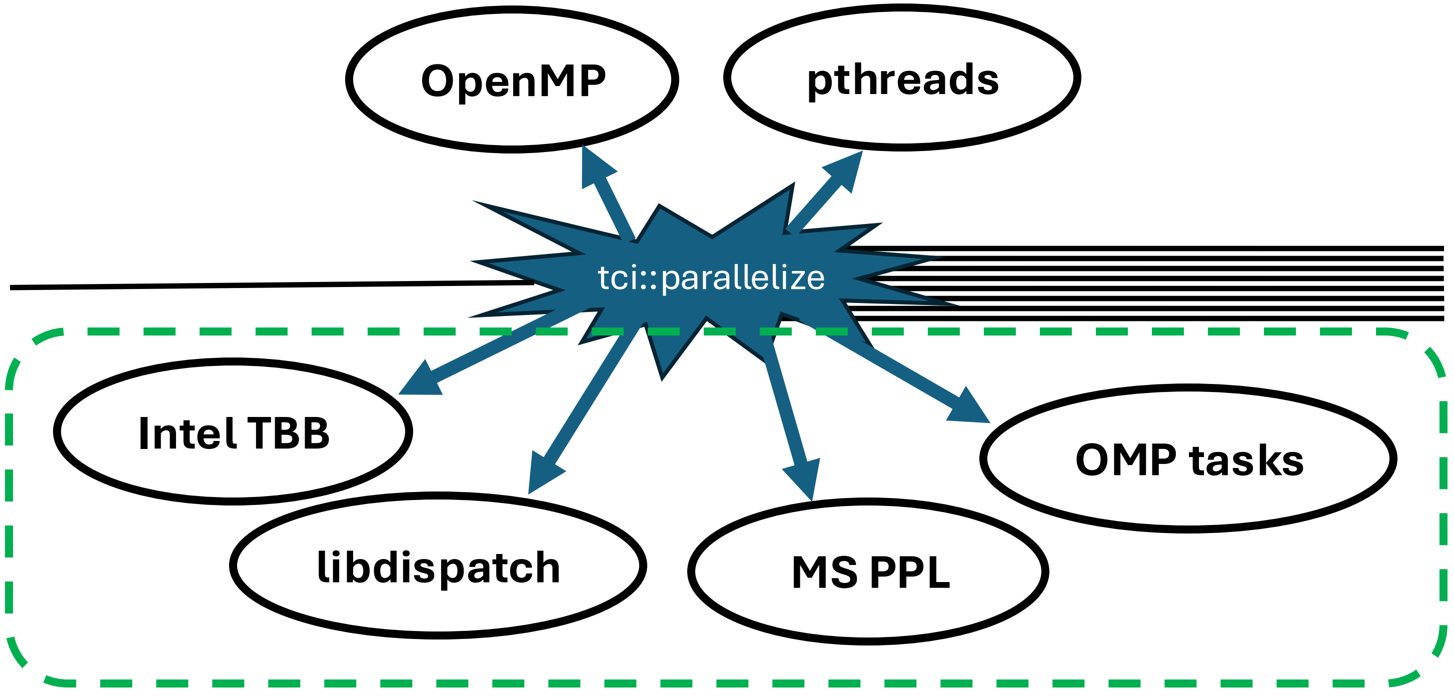
Multi-threading

- Threading over multiple cores (up to 100s) uses the external **Thread Communicator Interface (TCI)** library.



Multi-threading

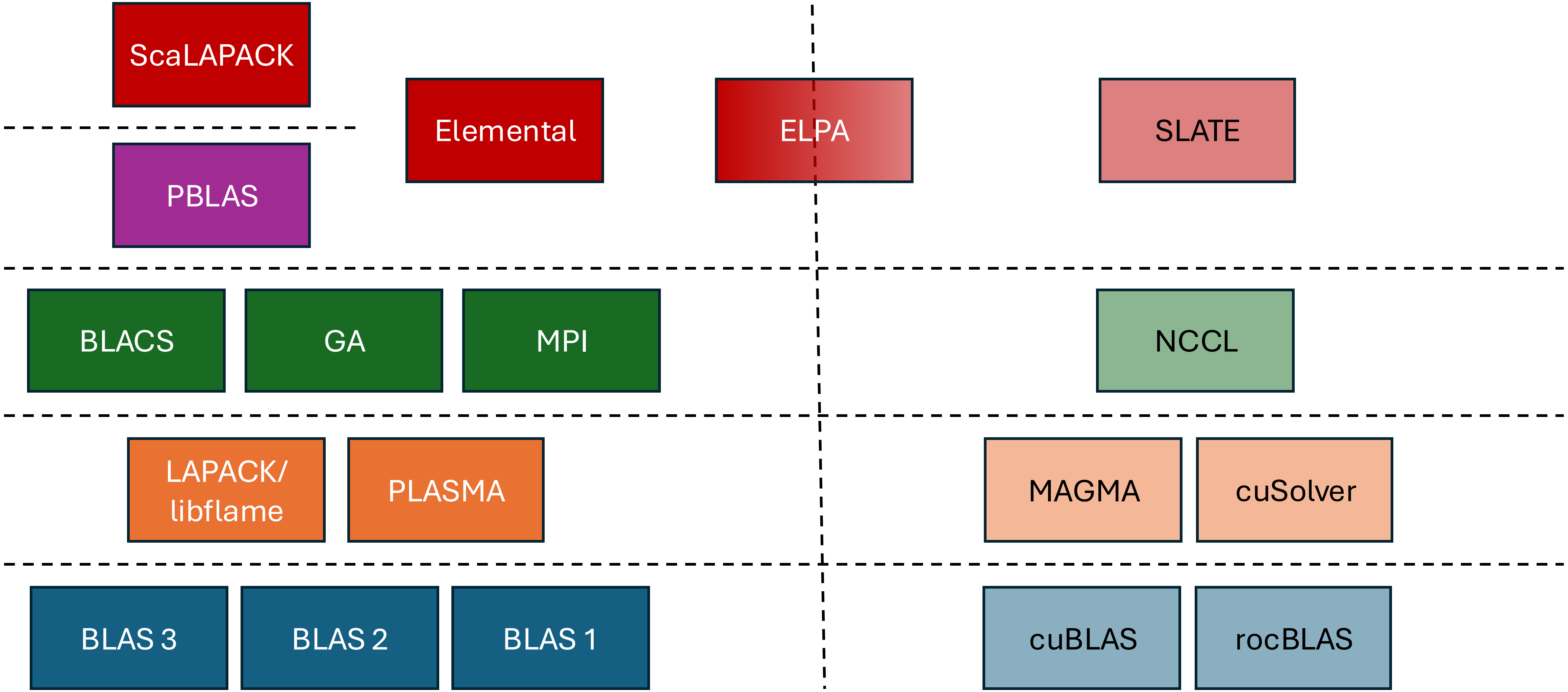
- Threading over multiple cores (up to 100s) uses the external **Thread Communicator Interface (TCI)** library.



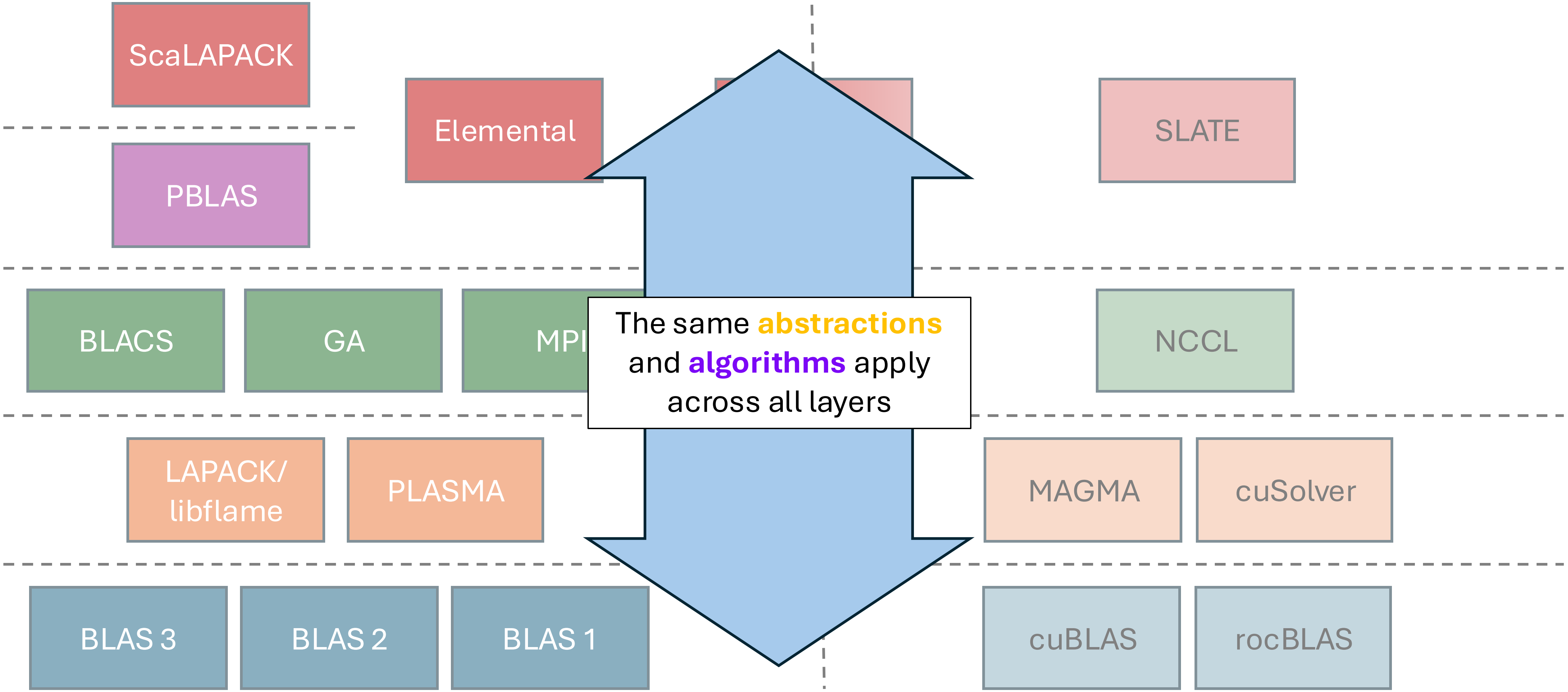
Revisiting libflame et al.:

All in the FAMLIES

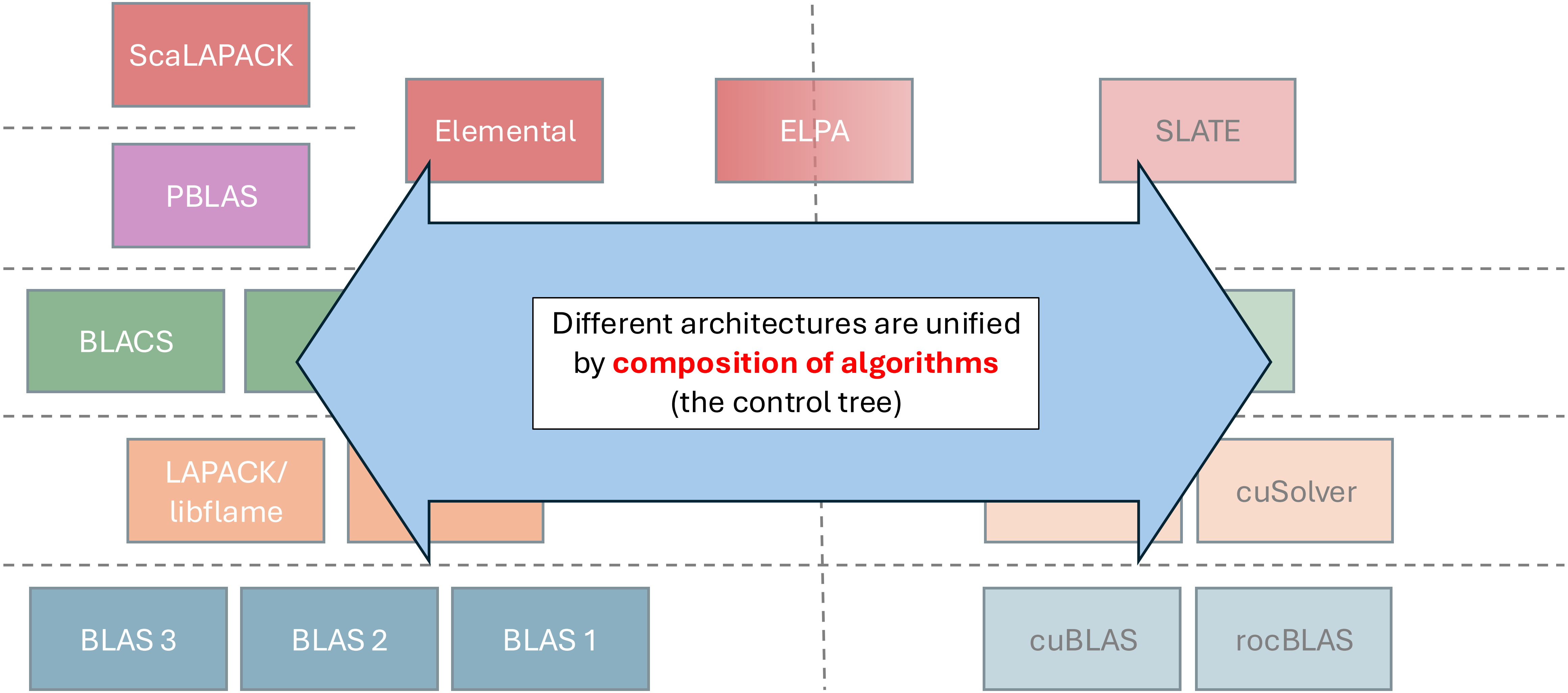
The Traditional DLA Stack



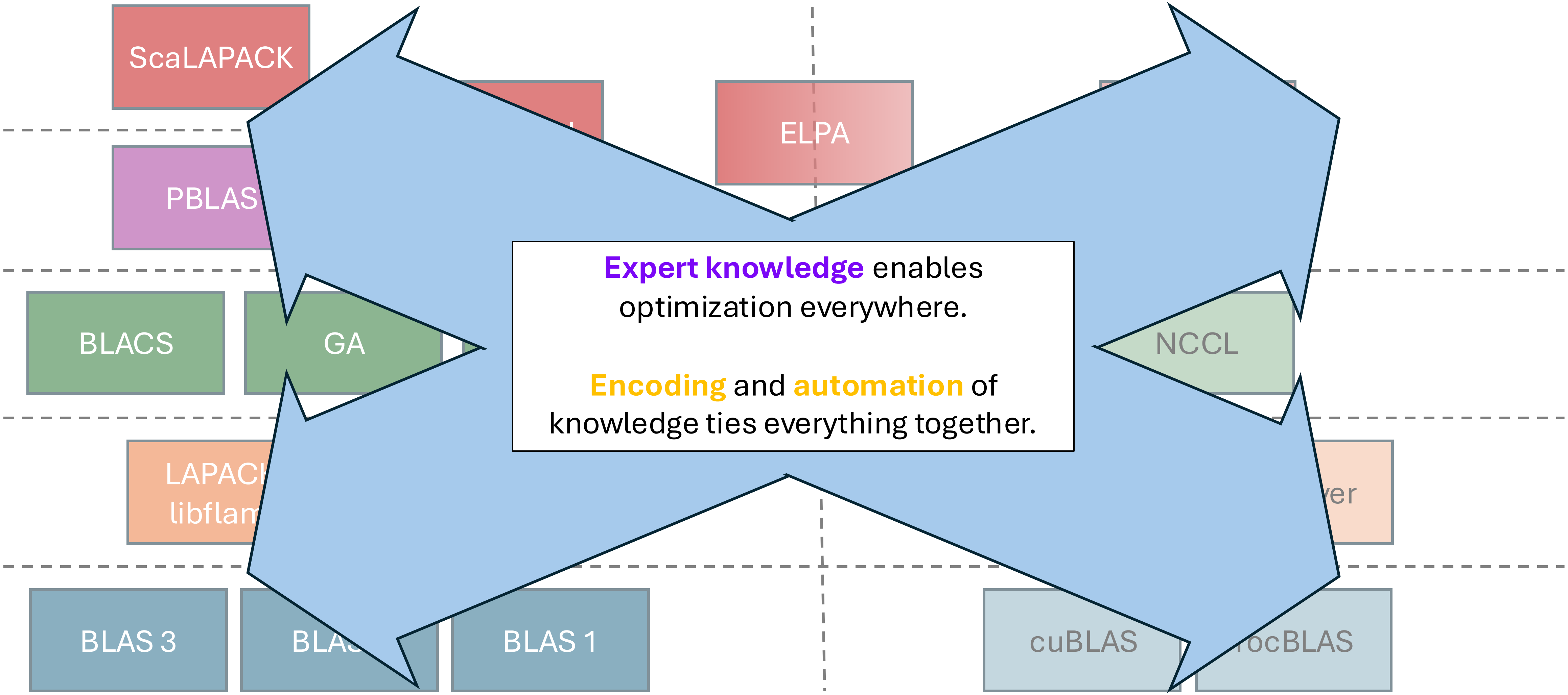
The Traditional DLA Stack



The Traditional DLA Stack



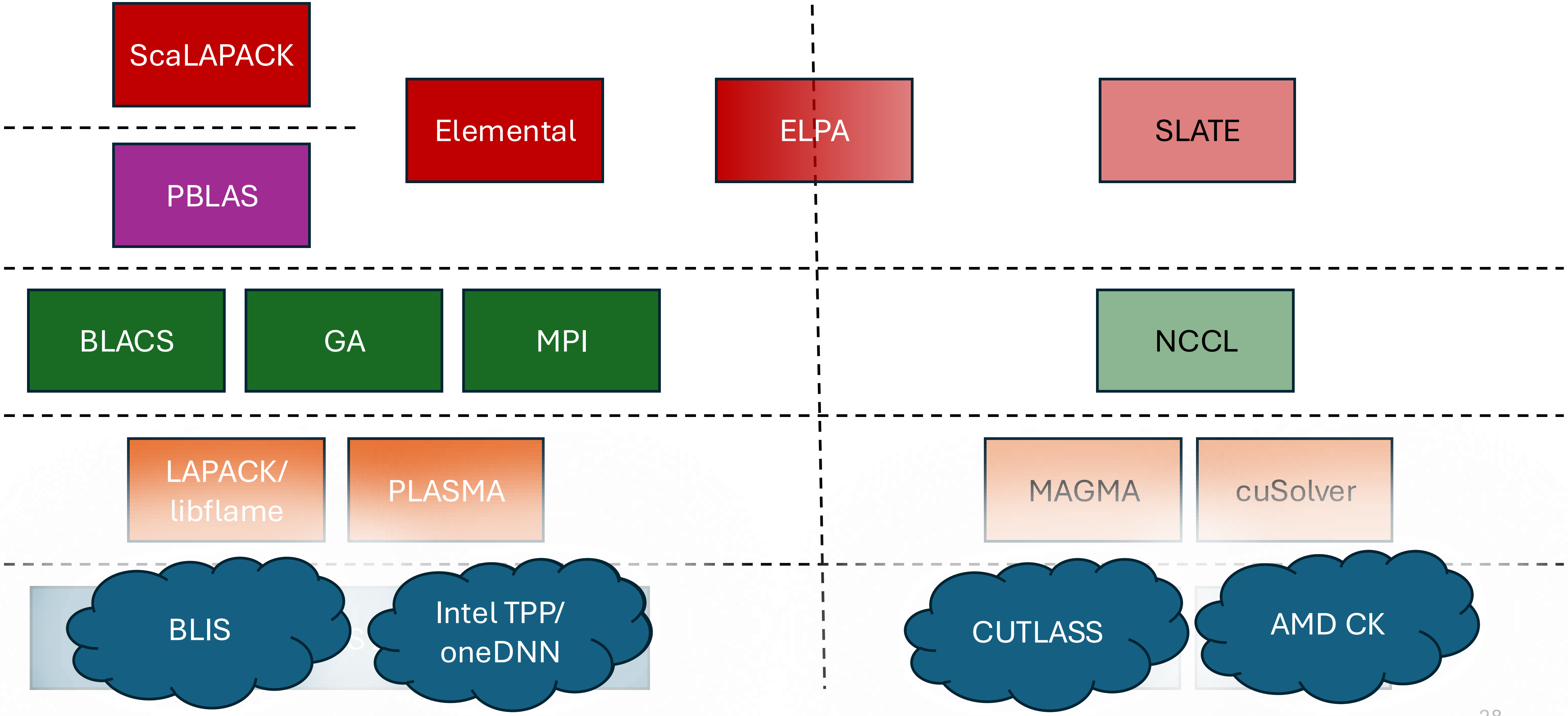
The Traditional DLA Stack

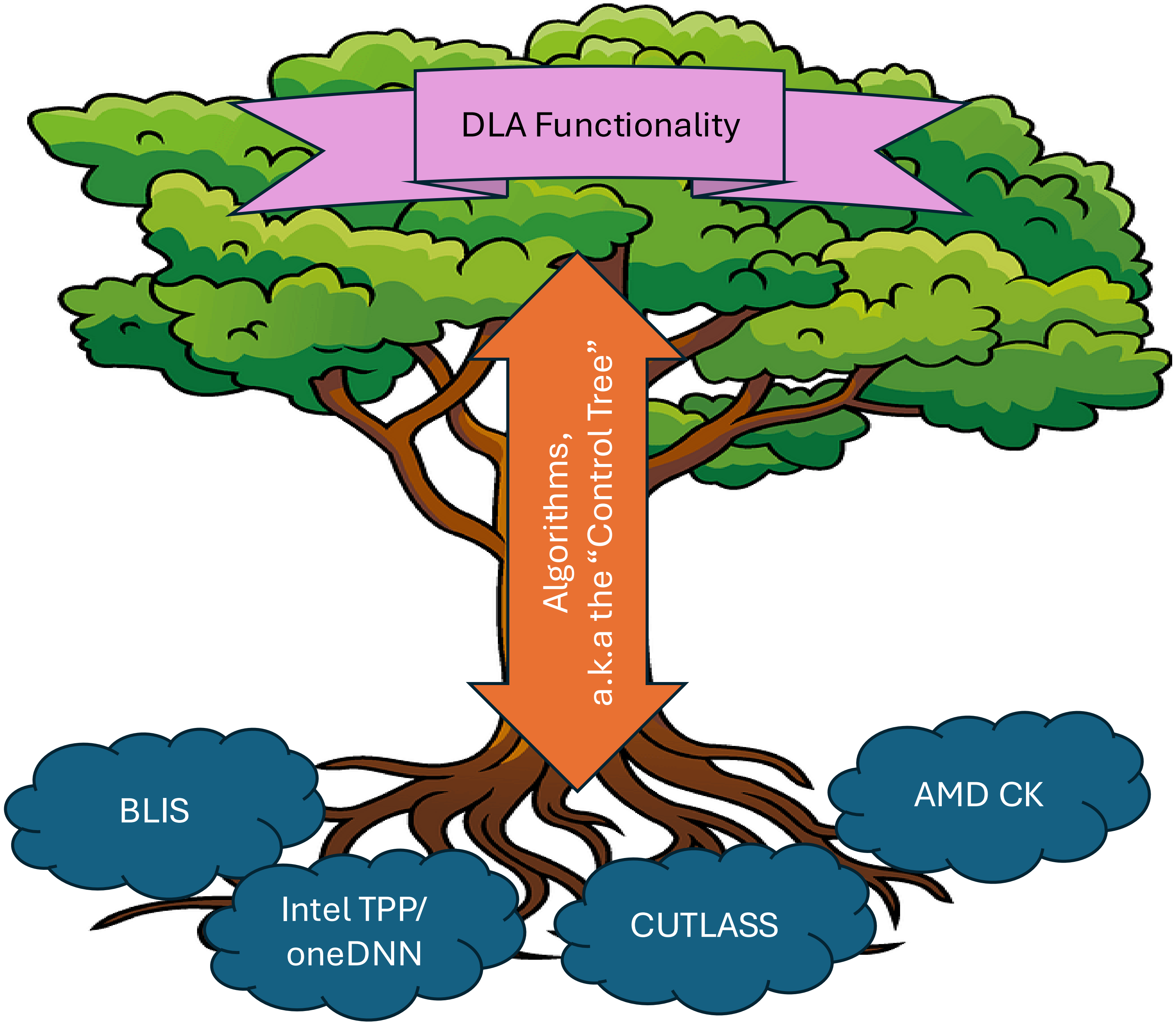


Expert knowledge enables optimization everywhere.

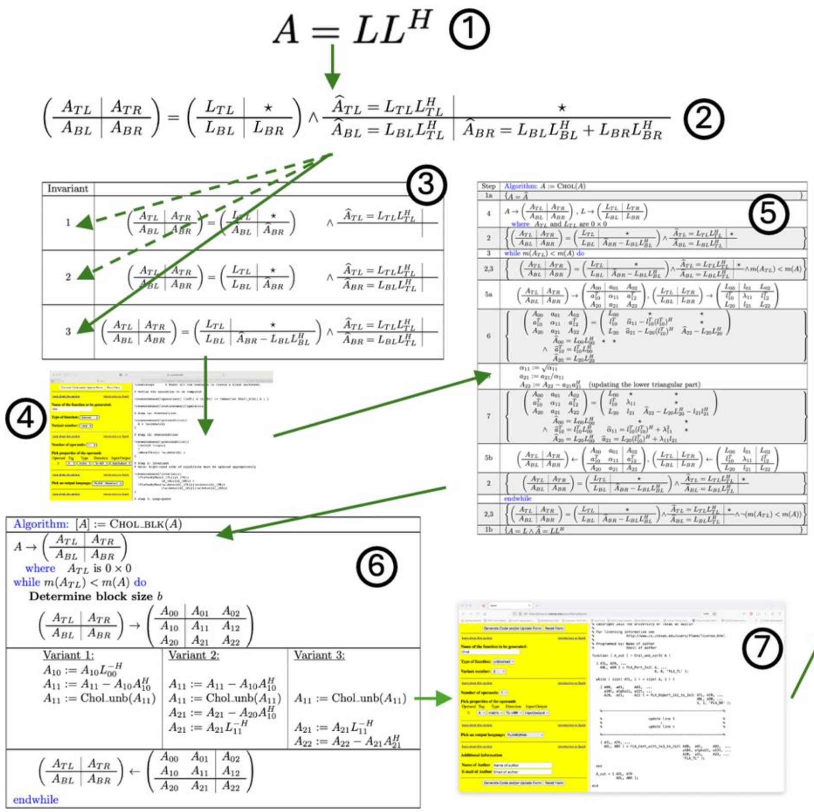
Encoding and **automation** of knowledge ties everything together.

Recent Developments




DLA Functionality
Algorithms,
a.k.a the “Control Tree”
BLIS
Intel TPP/
oneDNN
CUTLASS
AMD CK

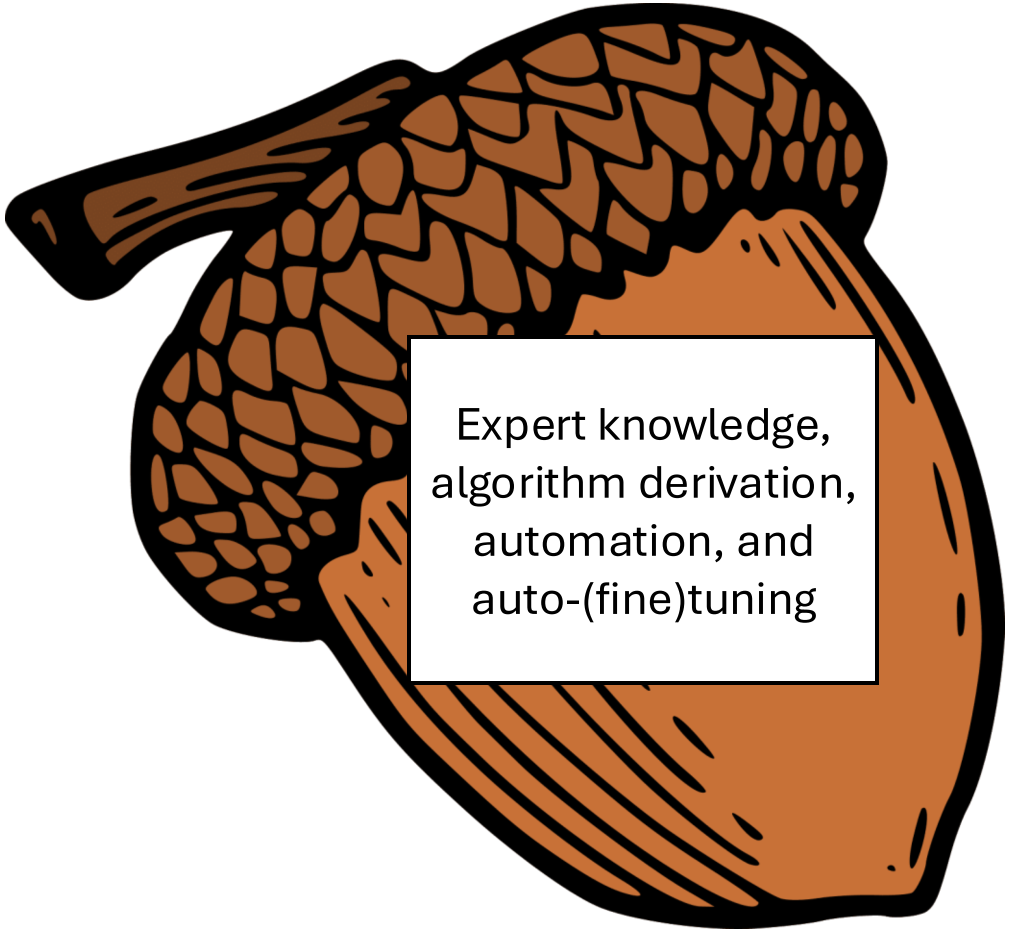
FAMLIES



⑧

```
FLA_Part_2x2( A,    &ATL, &ATR,
                    &ABL, &ABR,     0, 0, FLA_TL );

while ( FLA_Obj_length( ATL ) < FLA_Obj_length( A ) ){

  b = FLA_Determine_blocksize( ABR, FLA_BR, FLA_Cntl_blocksize( cntl ) );

  FLA_Repart_2x2_to_3x3( ATL, /**/ ATR,       &A00, /**/ &A01, &A02,
                      /* ************* */   /* ******************** */
                                              &A10, /**/ &A11, &A12,
                         ABL, /**/ ABR,       &A20, /**/ &A21, &A22,
                         b, b, FLA_BR );

  /*------------------------------------------------------------*/

  // A11 = chol( A11 )
  r_val = FLA_Chol_internal( FLA_LOWER_TRIANGULAR, A11,
                             FLA_Cntl_sub_chol( cntl ) );

  if ( r_val != FLA_SUCCESS )
    return ( FLA_Obj_length( A00 ) + r_val );

  // A21 = A21 * inv( tril( A11 )' )
  FLA_Trsm_internal( FLA_RIGHT, FLA_LOWER_TRIANGULAR,
                     FLA_CONJ_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A21,
                     FLA_Cntl_sub_trsm( cntl ) );

  // A22 = A22 - A21 * A21'
  FLA_Herk_internal( FLA_LOWER_TRIANGULAR, FLA_NO_TRANSPOSE,
                     FLA_MINUS_ONE, A21, FLA_ONE, A22,
                     FLA_Cntl_sub_herk( cntl ) );

  /*------------------------------------------------------------*/

  FLA_Cont_with_3x3_to_2x2( &ATL, /**/ &ATR,       A00, A01, /**/ A02,
                                                   A10, A11, /**/ A12,
                          /* ************** */  /* ****************** */
                            &ABL, /**/ &ABR,       A20, A21, /**/ A22,
                            FLA_TL );
}
```

The Future DLA Stack?



Hopefully we'll see you at the FAMLIES Retreat 2026!