Abstraction Layer
For Standardizing APIs of Task-Based Engines

AL4SAN

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology
Extreme Computing Research Center

The abstraction layer for standardizing APIs of task-based engines (AL4SAN) is designed as a lightweight software library, which provides a collection of APIs to unify the expression of tasks and their data dependencies from existing dynamic engines. AL4SAN supports various dynamic runtime systems relying on compiler infrastructure technology or on library-defined APIs. It features an abstraction of task-based engines and, therefore, enables a single-code application to assess various runtimes and their respective scheduling components. The goal of AL4SAN is not to create yet another runtime system, but to further leverage the user-obliviousness of the underlying complex hardware architectures at the dawn of the Exascale age.

AL4SAN v1.0 Features

- Standardizing task-based runtime systems
- Using a lightweight abstraction layer
- Improving user productivity
- Supporting different hardware architectures
- Performing with a relatively limited overhead **(up to 10%)**

Cholesky Pseudo-Code

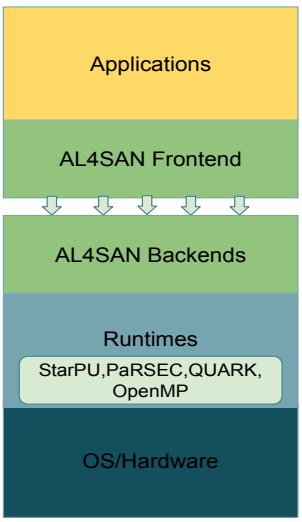
```
AL4SAN_Init(ncpus, ngpus);
sequence=AL4SAN_Sequence_Create();
for k ← 0 to nt do
    AL4SAN_Insert_Task(AL4SAN_TASK(POTRF),
      sequence, A[k][k]...);
    for m ← k+1 to nt do
        AL4SAN_Insert_Task(AL4SAN_TASK(TRSM),
          sequence, A[k][k], A[m][k], ...);
    for n ← k+1 to nt do
        AL4SAN_Insert_Task(AL4SAN_TASK(SYRK),
          sequence, A[n][k], A[n][n],...);
        for m ← n+1 to nt do
            AL4SAN_Insert_Task(AL4SAN_TASK(GEMM),
              sequence, A[m][k], A[n][k], A[m][n],...);
AL4SAN_Sequence_Wait( sequence);
AL4SAN_Sequence_Destroy(sequence);
AL4SAN_Finalize();
```

Task Interface

```
AL4SAN_Task_CPU(POTRF, POTRF_CPU)
AL4SAN_Insert_Task(AL4SAN_TASK(POTRF),
        sequence,
        AL4SAN_VALUE,   &uplo,   sizeof(int),
        AL4SAN_VALUE,   &n,      sizeof(int),
        AL4SAN_INOUT,   A[k][k], AL4SAN_DEP,
        AL4SAN_VALUE,   &lda,    sizeof(int),
        AL4SAN_VALUE,   &info,   sizeof(int),
        ..., ARG_END);

void POTRF_CPU (AL4SAN_arg_list *al4san_arg) {
    AL4SAN_Unpack_Arg(al4san_arg, &uplo, &nb,
      &A, &lda, &info );
    potrf_(&uplo, &nb, A, &lda, &info); }
```

Software Infrastructure



AL4SAN Roadmap

- Extending to more engines
- Leveraging data abstraction
- Integrating C++ constructs
- Composing across dynamic runtime systems
- Adding support to more algorithms and applications

Runtime Support

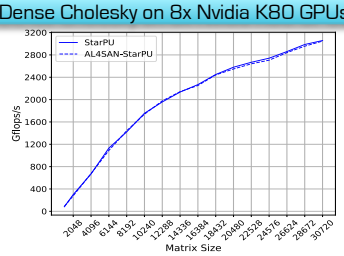
- OpenMP-LLVM
- PaRSEC
- StarPU
- QUARK

Main Reference

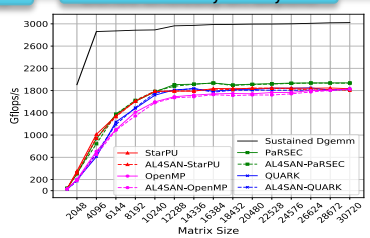
AL4SAN: Abstraction Layer For Standardizing APIs of Task-Based Engines, **R. Alomairy, H. Ltaief, M. Abduljabbar, and D. Keyes**,
Submitted to IPDPS'19,
Available at http://hdl.handle.net/10754/629718

Performance Assessment

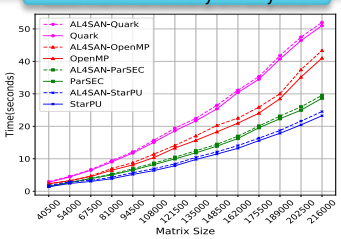
Dense Cholesky on 8x Nvidia K80 GPUs



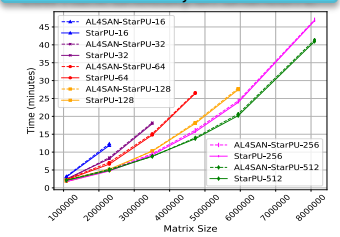
Dense Cholesky on Skylake



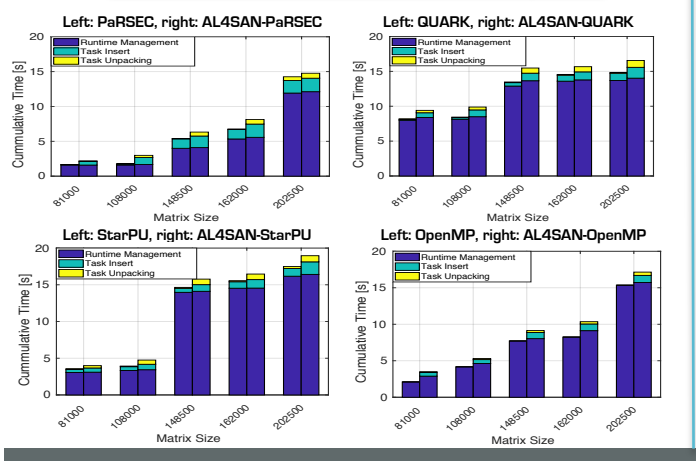
Low Rank Cholesky on Skylake



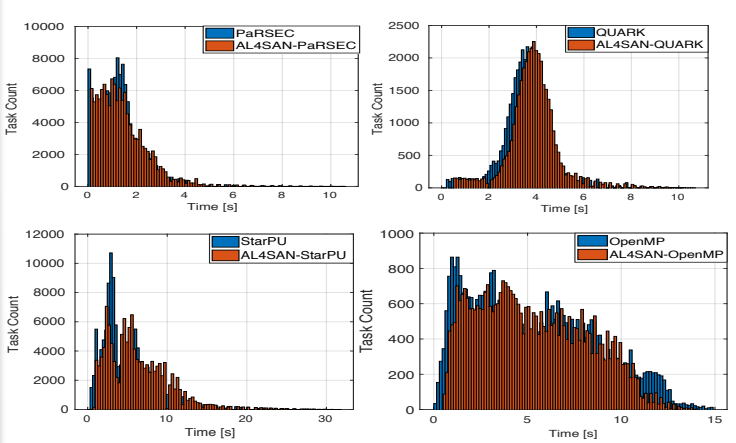
Low Rank Cholesky on Shaheen-2



Overhead Breakdown Across Runtimes on Skylake



Task Scheduling Distribution on Skylake



A collaboration of
With support from
Sponsored by