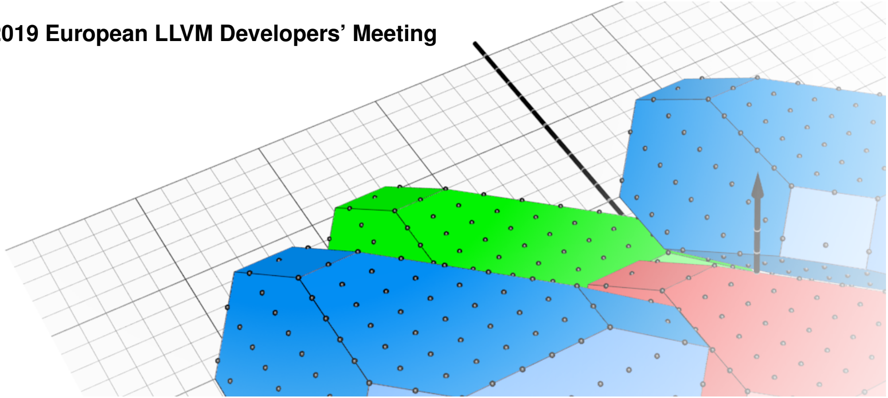
An alternative OpenMP Backend for Polly

TECHNISCHE UNIVERSITÄT DARMSTADT

Michael Halkenhäuser

2019 European LLVM Developers' Meeting



Embedded Systems & Applications

Polly

TECHNISCHE UNIVERSITÄT DARMSTADT

- Polyhedral framework on LLVM-IR

Embedded Systems & Applications

Polly

TECHNISCHE UNIVERSITÄT DARMSTADT

- Polyhedral framework on LLVM-IR
  - Efficient analyses and transformations
  - Code generation

Embedded Systems & Applications

Polly

TECHNISCHE UNIVERSITÄT DARMSTADT

- Polyhedral framework on LLVM-IR
  - Efficient analyses and transformations
  - Code generation
- Example transformations
  - Loop interchange / fission / fusion
  - Strip mining (Vectorization)

Polly

TECHNISCHE UNIVERSITÄT DARMSTADT

- Polyhedral framework on LLVM-IR
  - Efficient analyses and transformations
  - Code generation
- Example transformations
  - Loop interchange / fission / fusion
  - Strip mining (Vectorization)
  - Automatic parallelization

Polly – Sample Parallelization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Automatic parallelization
  - No need for manual annotation

Polly – Sample Parallelization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Automatic parallelization
  - No need for manual annotation

```
// "matvect" -- Sequential
// (Simplified dependencies)



for (i = 0; i <= n; i++) {
  for (j = 0; j <= n; j++)
    s[i] = s[i] + a[i][j] * x[j];
}
```

Input

Embedded Systems & Applications

Polly – Sample Parallelization

- Automatic parallelization
  - No need for manual annotation

```
// "matvect" -- Sequential
// (Simplified dependencies)



for (i = 0; i <= n; i++) {
  for (j = 0; j <= n; j++)
    s[i] = s[i] + a[i][j] * x[j];
}
```

Input

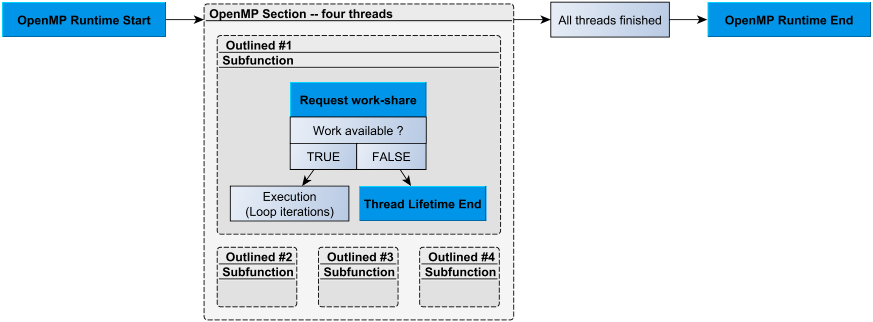
```
// "matvect" -- OpenMP parallelized
// Equivalent to the LLVM-IR output

#pragma omp parallel for [...] \
schedule (dynamic, 1) num_threads(N)
for (i = 0; i <= n; i++) {
  for (j = 0; j <= n; j++)
    s[i] = s[i] + a[i][j] * x[j];
}
```

Output

Polly – Parallelization Scheme

- Polly detects parallelizable code regions
  - Moved into an *outlined function*
  - Executed using OpenMP API



Motivation for an alternative OpenMP Backend

- Limited influence on OpenMP execution
  - Increase number of user options
  - Improve fine-tuning possibilities

Motivation for an alternative OpenMP Backend

- Limited influence on OpenMP execution
  - Increase number of user options
  - Improve fine-tuning possibilities

- Dependent on GNU OpenMP API
  - Expand the scope of application

Motivation for an alternative OpenMP Backend

TECHNISCHE UNIVERSITÄT DARMSTADT

- Limited influence on OpenMP execution
  - Increase number of user options
  - Improve fine-tuning possibilities
- Dependent on GNU OpenMP API
  - Expand the scope of application
- LLVM OpenMP implementation available
  - Enable direct use of LLVM's OpenMP runtime
  - Support automated testing

LLVM OpenMP Backend

TECHNISCHE UNIVERSITÄT DARMSTADT

- Extension of the preexisting backend
  - Reused common functionalities
    - Moved into abstract base class

Embedded Systems & Applications

LLVM OpenMP Backend

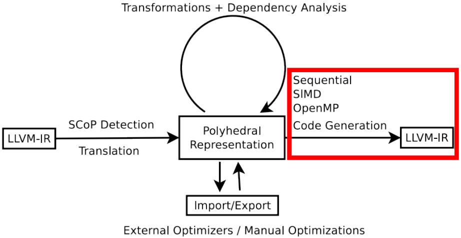
TECHNISCHE UNIVERSITÄT DARMSTADT

- Extension of the preexisting backend
  - Reused common functionalities
    - Moved into abstract base class
  - API-specific call creation and placement
    - Implemented in derived class per backend

LLVM OpenMP Backend

- Extension of the preexisting backend
  - Reused common functionalities
    - Moved into abstract base class
  - API-specific call creation and placement
    - Implemented in derived class per backend

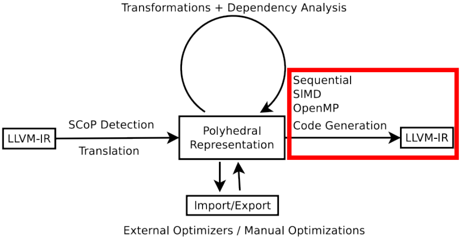
- User may **choose** **backend**
  - Via CL switch, similar to
    - *Number of threads*



LLVM OpenMP Backend

- Extension of the preexisting backend
  - Reused common functionalities
    - Moved into abstract base class
  - API-specific call creation and placement
    - Implemented in derived class per backend

- User may **choose** **backend**
  - Via CL switch, similar to
    - *Number of threads*
  - Additional options
    - *Scheduling type*
    - *Chunk size*



LLVM OpenMP Backend – Options

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Scheduling type* determines work distribution

static	***dynamic***	***guided***
Predetermined, uniform distribution of iterations	Threads request work shares of *chunk size*	Hybrid scheduling of *static* and *dynamic*, using a *minimum chunk size*

LLVM OpenMP Backend – Options

- *Scheduling type* determines work distribution

	static	***dynamic***	***guided***
Load Balancing	–	+	○
Organization Overhead	+	–	○

LLVM OpenMP Backend – Options

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Scheduling type* determines work distribution

	static	***dynamic***	***guided***
Load Balancing	–	+	○
Organization Overhead	+	–	○

- ***static*** suited for constant computational demands
- ***dynamic*** suited for shifting computational demands
- ***guided*** suited for "both"

Experimental Methodology

TECHNISCHE UNIVERSITÄT DARMSTADT

- PolyBench[1]
  - Provides multiple datasets
  - Triggers auto-parallelization in 18 benchmarks

[1] https://sourceforge.net/projects/polybench/

Experimental Methodology

TECHNISCHE UNIVERSITÄT DARMSTADT

- PolyBench[1]
  - Provides multiple datasets
  - Triggers auto-parallelization in 18 benchmarks
- Runtime results
  - Average from 50 out of 60 runs (10% trimmed-mean)
  - Utilized CPU: AMD R5 1600X

[1] https://sourceforge.net/projects/polybench/

Embedded Systems & Applications

Experimental Methodology

TECHNISCHE UNIVERSITÄT DARMSTADT

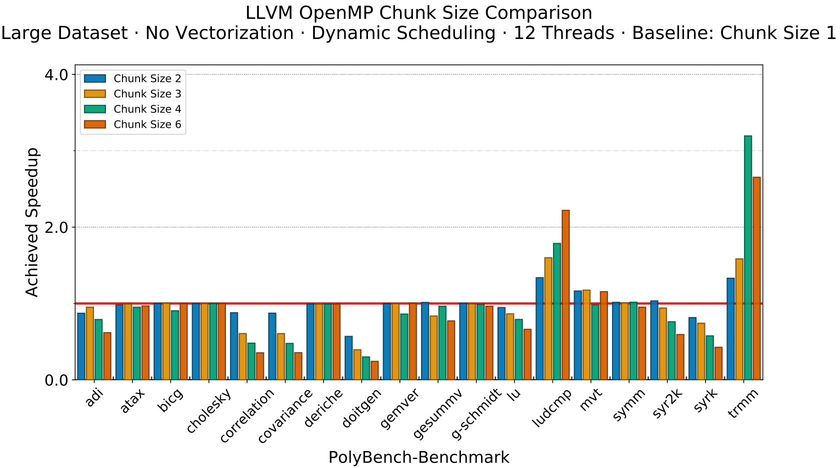
- PolyBench[1]
  - Provides multiple datasets
  - Triggers auto-parallelization in 18 benchmarks
- Runtime results
  - Average from 50 out of 60 runs (10% trimmed-mean)
  - Utilized CPU: AMD R5 1600X
- Plots show *relative speedup*
  - $speedup = \frac{runtime\ of\ baseline}{runtime\ of\ competitor}$

[1] https://sourceforge.net/projects/polybench/

Embedded Systems & Applications

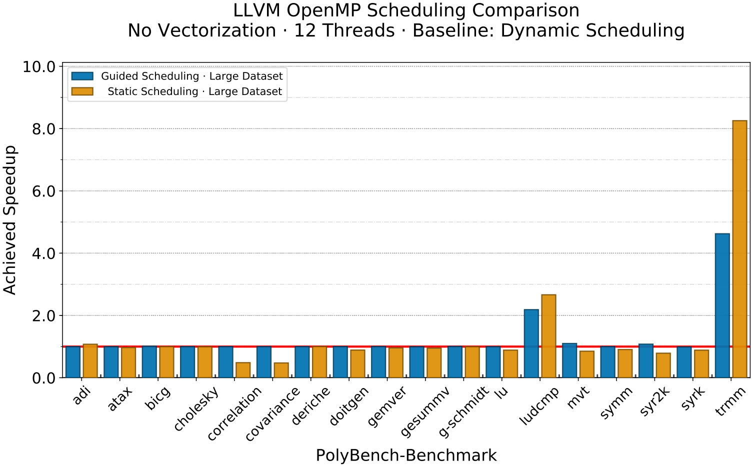
Performance Impact of *chunk size*

TECHNISCHE UNIVERSITÄT DARMSTADT



Performance Impact of *scheduling type*

TECHNISCHE
UNIVERSITÄT
DARMSTADT



Intermezzo – Customization Options

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Chunk size*
  - 1 is usually a reasonable choice
  - Very beneficial in particular cases
    - More than 3$\times$ speedup possible

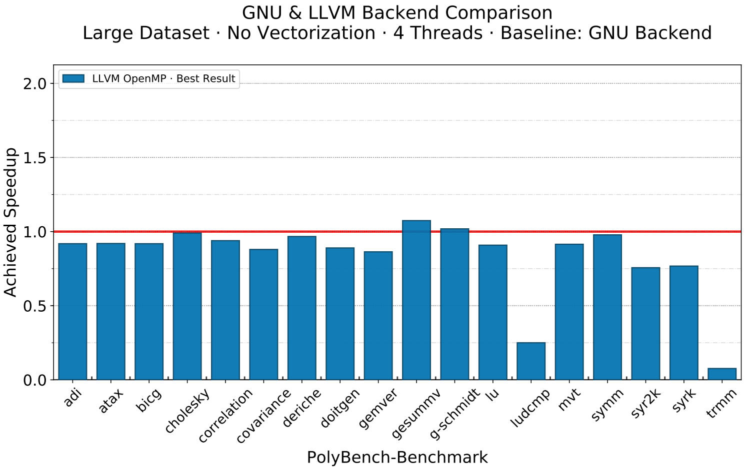
Intermezzo – Customization Options

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Chunk size*
  - 1 is usually a reasonable choice
  - Very beneficial in particular cases
    - More than $3\times$ speedup possible
- *Scheduling type*
  - *Dynamic:* Good overall performance
  - *Guided:* Performs at least as good as *dynamic*
  - *Static:* Problem-dependent
    - May achieve $8\times$ speedup compared to *dynamic*

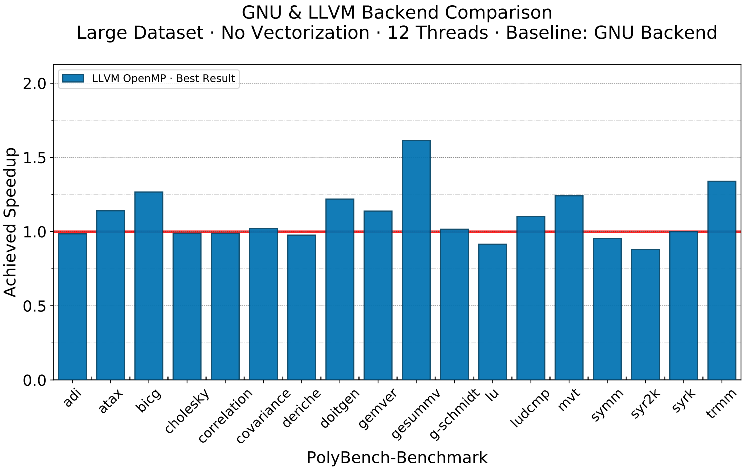
Backend Comparison
LLVM versus GNU OpenMP Backend

TECHNISCHE UNIVERSITÄT DARMSTADT



Backend Comparison
LLVM versus GNU OpenMP Backend

TECHNISCHE UNIVERSITÄT DARMSTADT



Intermezzo – Backend Comparison

- Using the maximum number of available threads
- Our "LLVM" backend
  - Achieves comparable performance
  - Performs significantly faster than "GNU" in seven cases
  - Reaches up to 1.6$\times$ speedup

Intermezzo – Backend Comparison

TECHNISCHE UNIVERSITÄT DARMSTADT

- Using the maximum number of available threads
- Our "LLVM" backend
  - Achieves comparable performance
  - Performs significantly faster than "GNU" in seven cases
  - Reaches up to 1.6$\times$ speedup
- GNU backend
  - Only a single, considerable lead

Embedded Systems & Applications

Intermezzo – Backend Comparison

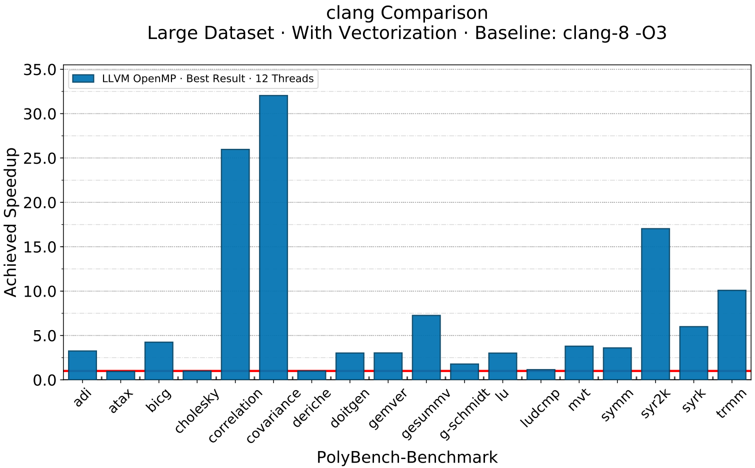
TECHNISCHE UNIVERSITÄT DARMSTADT

- Using the maximum number of available threads
- Our "LLVM" backend
  - Achieves comparable performance
  - Performs significantly faster than "GNU" in seven cases
  - Reaches up to 1.6$\times$ speedup
- GNU backend
  - Only a single, considerable lead

- Additional switches
  - Allow problem-specific adjustments
  - ... without depending on env. variable

General Comparison
LLVM OpenMP Backend versus clang

TECHNISCHE
UNIVERSITÄT
DARMSTADT



Embedded Systems & Applications

Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

- Our "LLVM" OpenMP backend for Polly
  - Represents a superior alternative

Embedded Systems & Applications

Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

- Our "LLVM" OpenMP backend for Polly
  - Represents a superior alternative
  - Acts as drop-in replacement

Conclusion

- Our "LLVM" OpenMP backend for Polly
  - Represents a superior alternative
  - Acts as drop-in replacement
  - Provides more customization options

Conclusion

- Our "LLVM" OpenMP backend for Polly
  - Represents a superior alternative
  - Acts as drop-in replacement
  - Provides more customization options
  - Carries no clear drawbacks, but instead ...
    - Reaches up to 1.6$\times$ speedup

Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

- Our "LLVM" OpenMP backend for Polly
  - Is publicly available
    - Review accepted on March 19th
      https://reviews.llvm.org/D59100
    - Currently on Polly‘s master branch
      https://github.com/llvm/llvm-project/commit/89251ed
- References:

  - Title graphic: https://polly.llvm.org/images/header-background.png
  - T. Grosser, H. Zheng, R. Aloor, A. Simbürger, A. Größlinger, and L. - N. Pouchet, “Polly - Polyhedral optimization in LLVM,” in Proceedings of the First International Workshop on Polyhedral Compilation Techniques (IMPACT), vol. 2011, 2011, p. 1.

Questions ?

- Ask them now, or ...
- Find me tomorrow, at the poster session
  - 09:00 am - 10:00 am (Foyer)