基于MLIR的RISC-V编译优化实践
——以Buddy Compiler为例

演讲人：周旭林、张洪滨

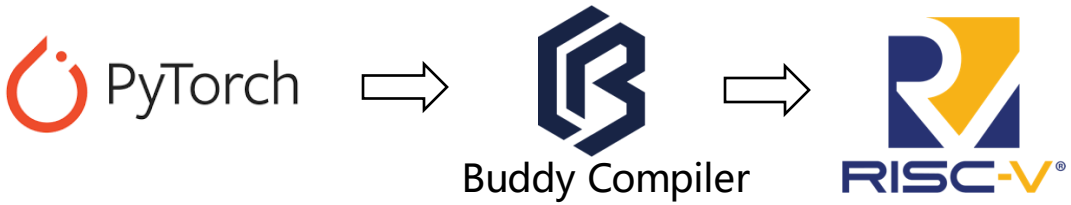
作者： 周旭林(ISCAS) 张洪滨(ISCAS) 高世豪(ISCAS) 王润(ETH) 张煦正(BJUT) 陈威威(BJUT) 刘尚昊(BUAA)

Buddy Compiler

概述：基于MLIR的RISC-V编译优化实践

Buddy Compiler

面向RISC-V CPU的AI模型向量化



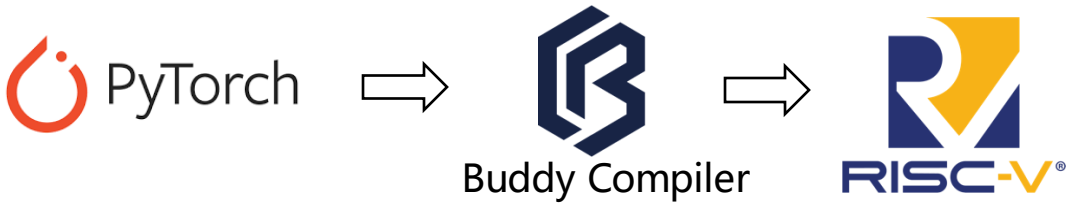
‘V’ 拓展汇编指令

```
vsetvli zero, s3, e32, m2, ta, ma
vle32.v v10, (s4), v0.t
add s4, s2, s1
slli    s4, s4, 2
add s4, a4, s4
vle32.v v12, (s4), v0.t
vmul.vx v10, v10, t6
vadd.vv v10, v10, v12
vse32.v v10, (s4), v0.t
add s1, s1, s3
sub t5, t5, s3
bgtz    t5, .LBB1_7
```

概述：基于MLIR的RISC-V编译优化实践

Buddy Compiler

面向RISC-V CPU的AI模型向量化



‘V’ 拓展汇编指令

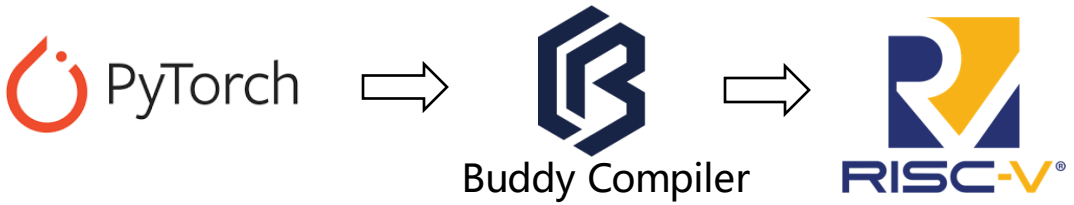
```
vsetvli zero, s3, e32, m2, ta, ma
vle32.v v10, (s4), v0.t
add s4, s2, s1
slli    s4, s4, 2
add s4, a4, s4
vle32.v v12, (s4), v0.t
vmul.vx v10, v10, t6
vadd.vv v10, v10, v12
vse32.v v10, (s4), v0.t
add s1, s1, s3
sub t5, t5, s3
bgtz    t5, .LBB1_7
```

Buddy Compiler的技术路线

概述：基于MLIR的RISC-V编译优化实践

Buddy Compiler

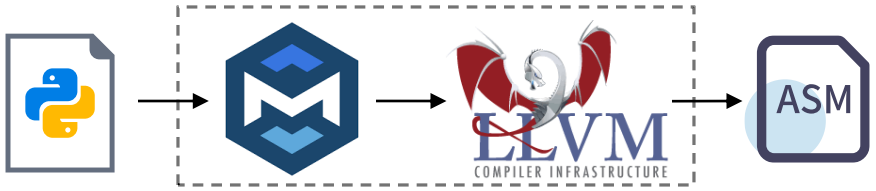
面向RISC-V CPU的AI模型向量化



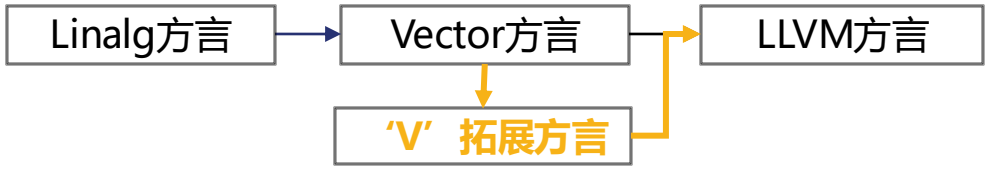
‘V’ 拓展汇编指令

```
vsetvli zero, s3, e32, m2, ta, ma
vle32.v v10, (s4), v0.t
add s4, s2, s1
slli    s4, s4, 2
add s4, a4, s4
vle32.v v12, (s4), v0.t
vmul.vx v10, v10, t6
vadd.vv v10, v10, v12
vse32.v v10, (s4), v0.t
add s1, s1, s3
sub t5, t5, s3
bgtz    t5, .LBB1_7
```

Buddy Compiler的技术路线



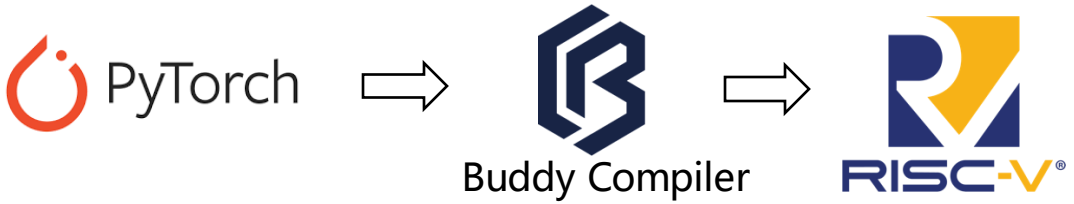
依托MLIR的多级别方言发挥 ‘V’ 拓展向量化潜力



概述：基于MLIR的RISC-V编译优化实践

Buddy Compiler

面向RISC-V CPU的AI模型向量化

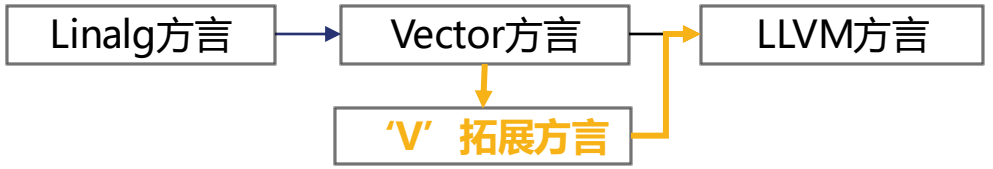


‘V’ 拓展汇编指令

```
vsetvli zero, s3, e32, m2, ta, ma
vle32.v v10, (s4), v0.t
add s4, s2, s1
slli    s4, s4, 2
add s4, a4, s4
vle32.v v12, (s4), v0.t
vmul.vx v10, v10, t6
vadd.vv v10, v10, v12
vse32.v v10, (s4), v0.t
add s1, s1, s3
sub t5, t5, s3
bgtz    t5, .LBB1_7
```

Buddy Compiler的技术路线

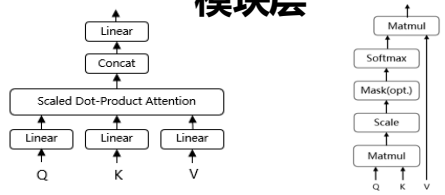
依托MLIR的多级别方言发挥 ‘V’ 拓展向量化潜力



基于 ‘V’ 拓展的多层级Benchmark

模型层

模块层



算子层

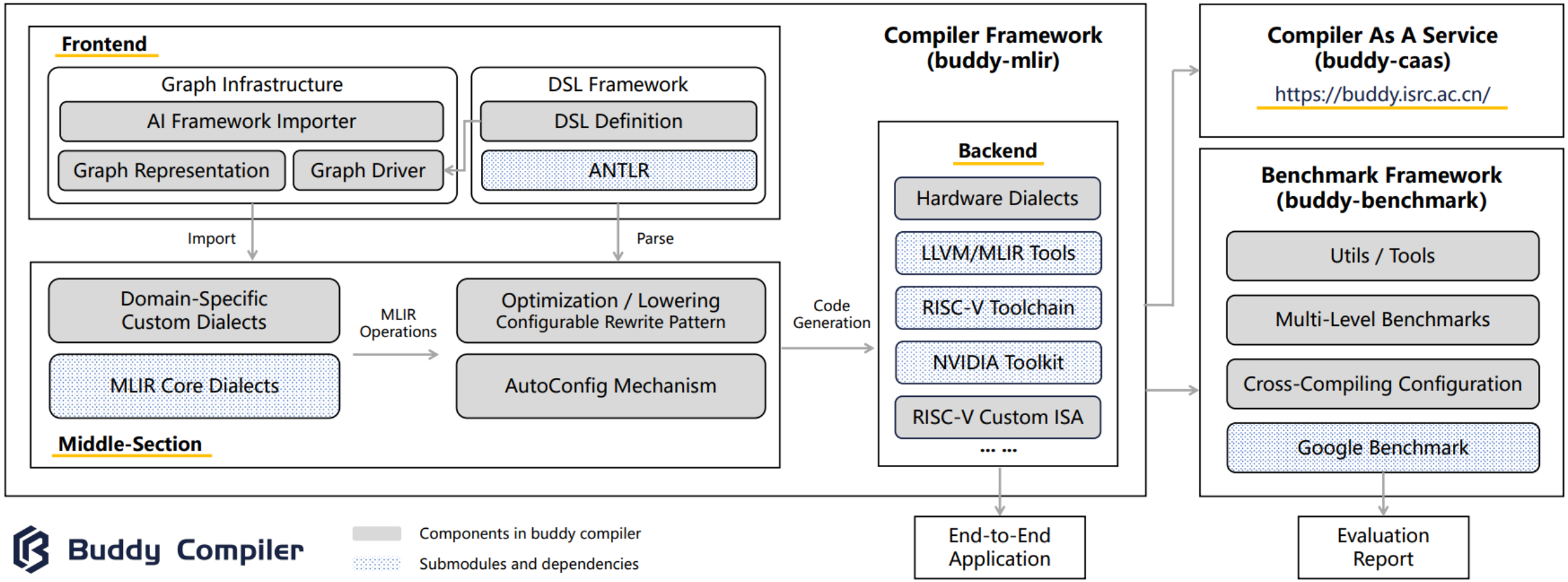
Matmul　Linear

Softmax　…

算子层（方言实现）

linalg.matmul
linalg.batch_matmul
linalg.softmax…

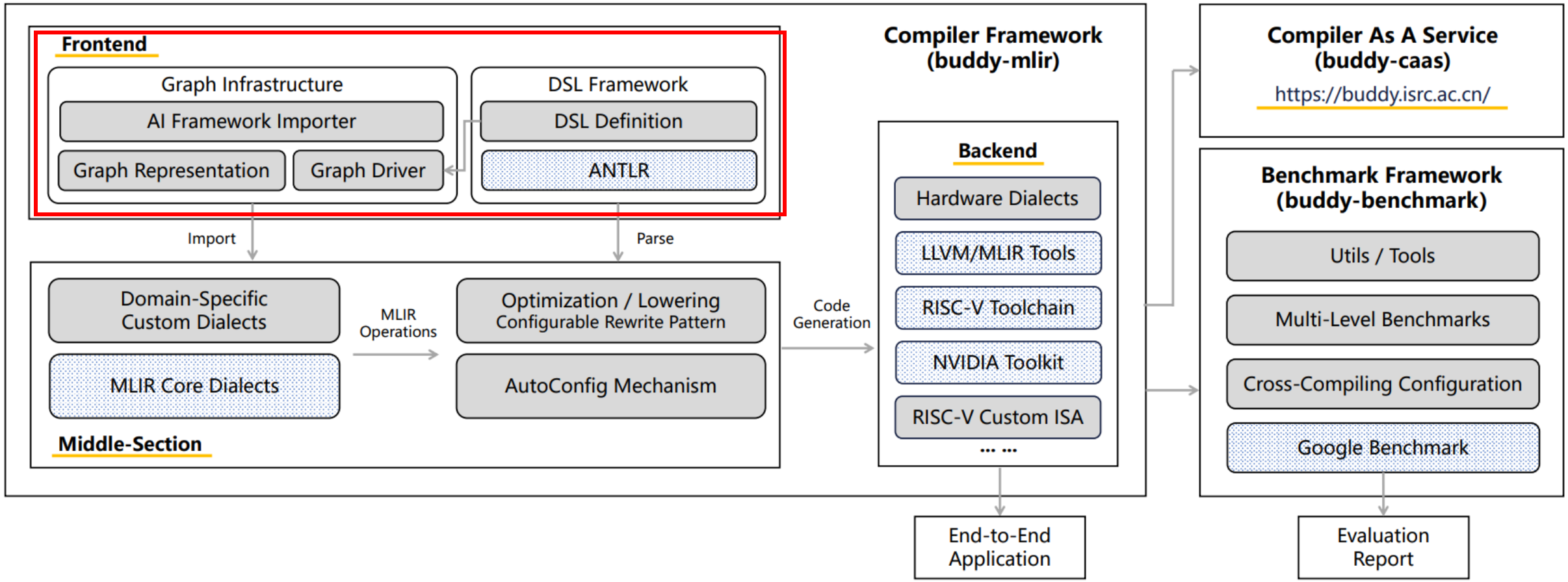
AI编译框架Buddy Compiler



AI编译框架Buddy Compiler

Buddy Compiler

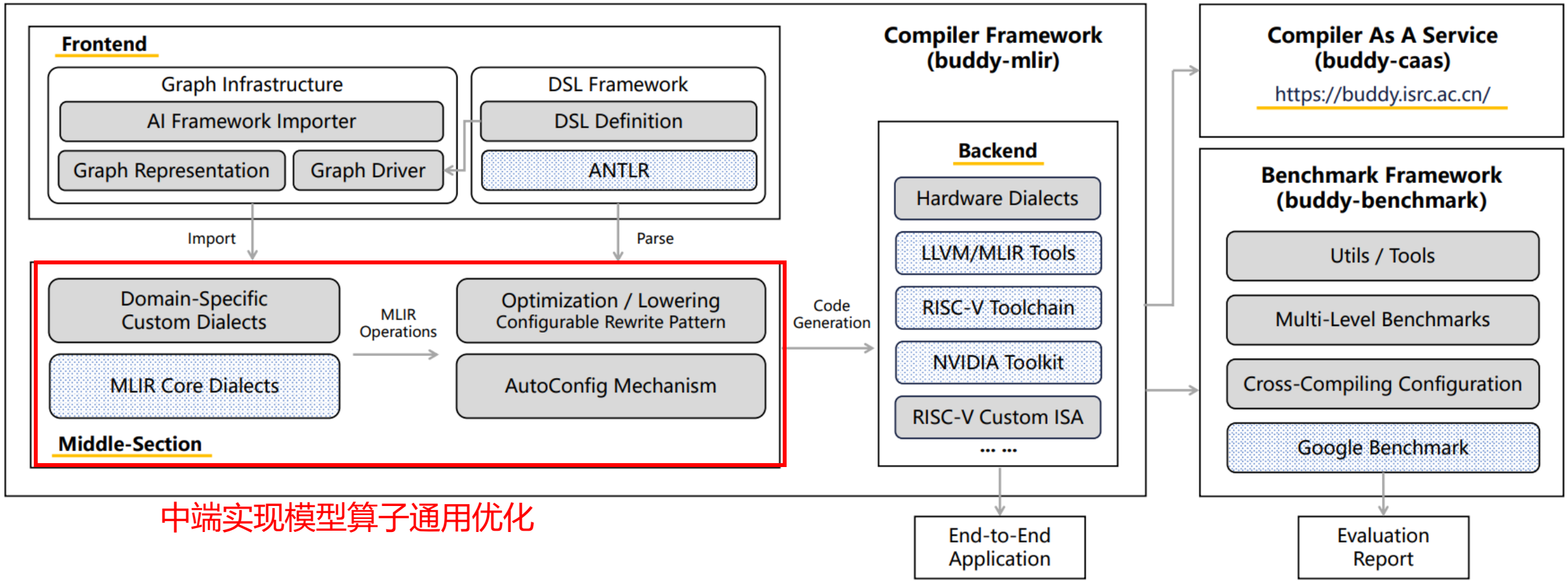
前端图基础设施支持Pytorch模型接入



AI编译框架Buddy Compiler

Buddy Compiler

前端图基础设施支持Pytorch模型接入



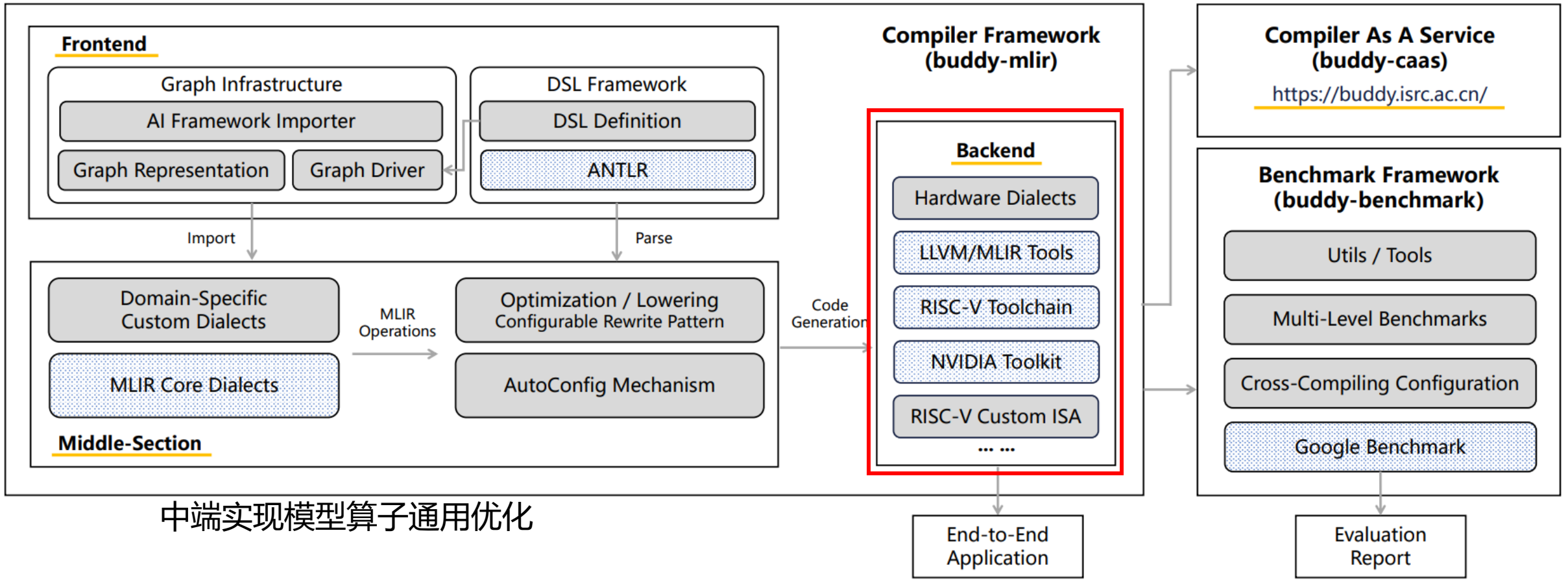
中端实现模型算子通用优化

AI编译框架Buddy Compiler

Buddy Compiler

前端图基础设施支持Pytorch模型接入

后端添加自定义方言（‘V’ 拓展优化支持的关键！）



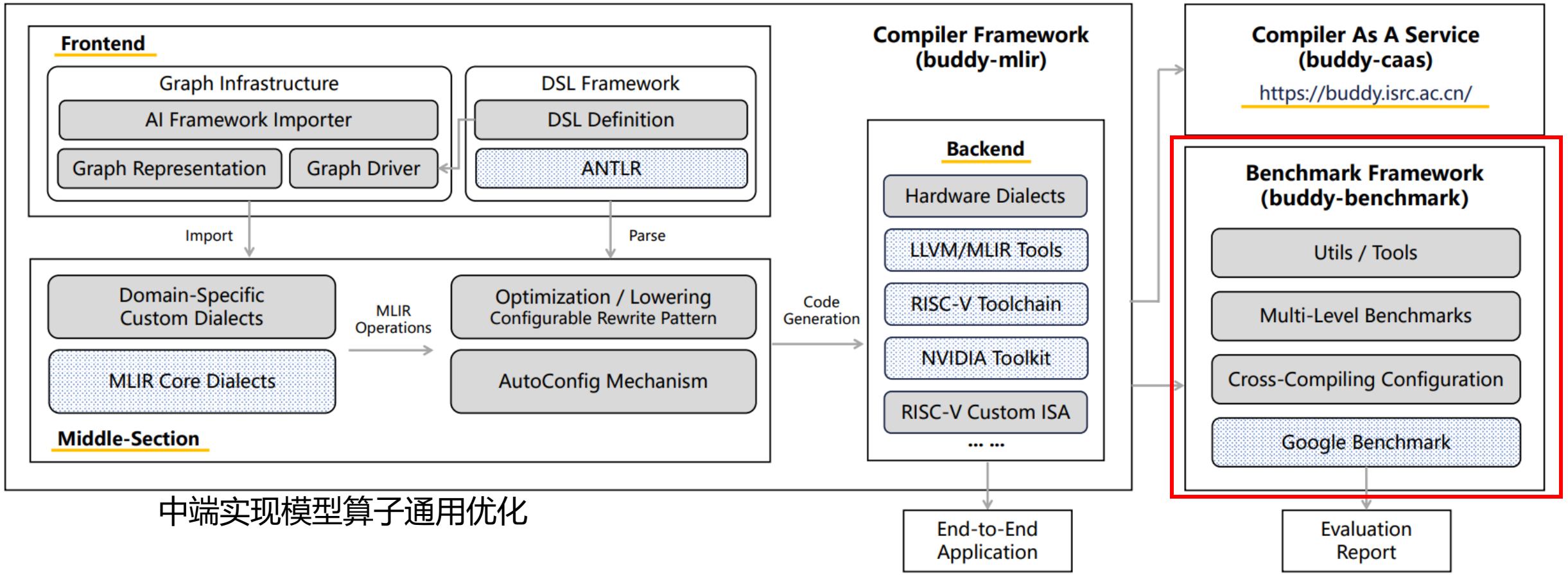
中端实现模型算子通用优化

AI编译框架Buddy Compiler

Buddy Compiler

前端图基础设施支持Pytorch模型接入

后端添加自定义方言



中端实现模型算子通用优化

多层级 ‘V’ 拓展Benchmark

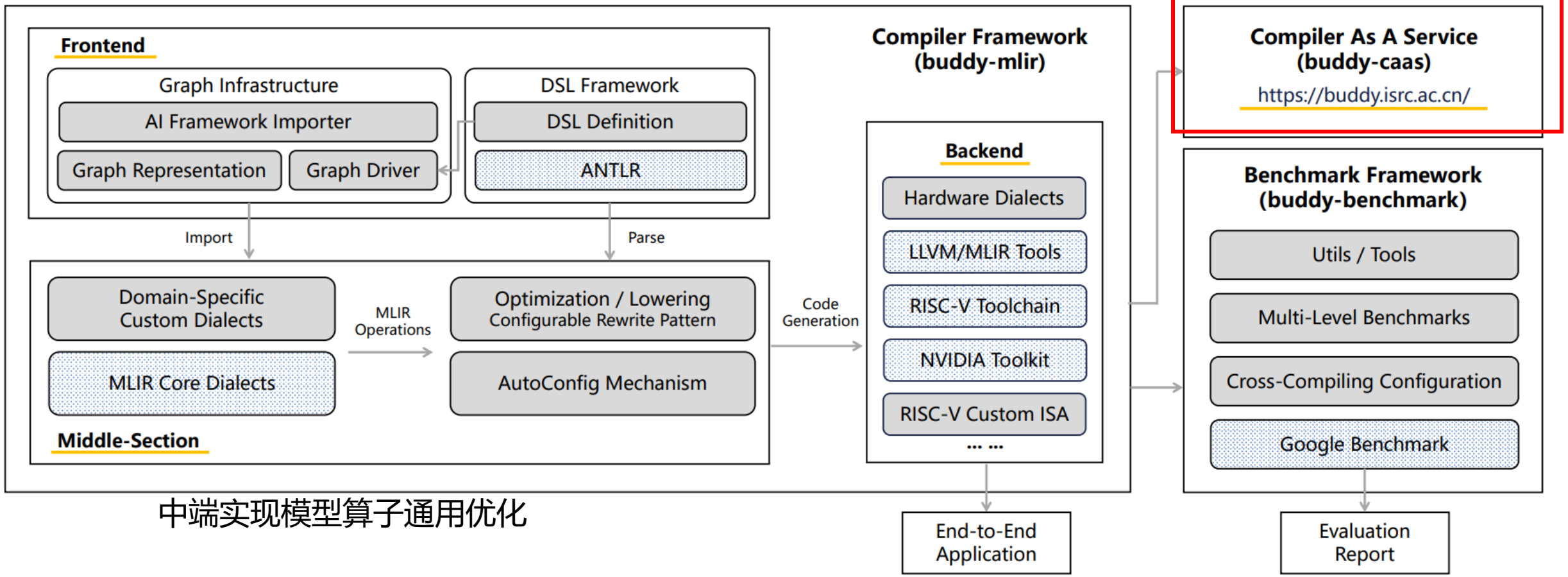
AI编译框架Buddy Compiler

Buddy Compiler

前端图基础设施支持Pytorch模型接入

后端添加自定义方言

MLIR界的“godbolt”！



中端实现模型算子通用优化

多层级 ‘V’ 拓展Benchmark

‘V’ 拓展的动态配置特性

Buddy Compiler

‘V’ 拓展中的配置指令(vsetvli / vsetivli / vsetvl)

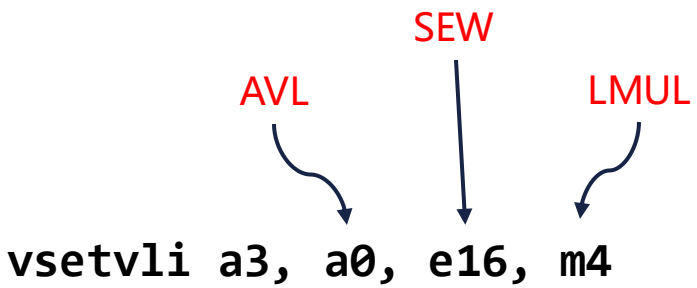
```
vsetvli rd, rs1, vtypei   # rd = new vl, rs1 = AVL, vtypei = new vtype setting
vsetivli rd, uimm, vtypei # rd = new vl, uimm = AVL, vtypei = new vtype setting
vsetvl  rd, rs1, rs2      # rd = new vl, rs1 = AVL, rs2 = new vtype value
```

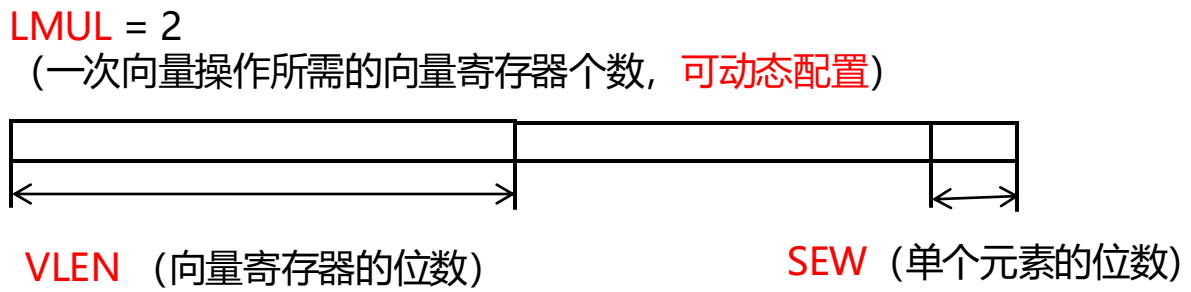
‘V’ 拓展的动态配置特性

Buddy Compiler

‘V’ 拓展中的配置指令(vsetvli / vsetivli / vsetvl)

```
vsetvli rd, rs1, vtypei   # rd = new vl, rs1 = AVL, vtypei = new vtype setting
vsetivli rd, uimm, vtypei # rd = new vl, uimm = AVL, vtypei = new vtype setting
vsetvl  rd, rs1, rs2      # rd = new vl, rs1 = AVL, rs2 = new vtype value
```



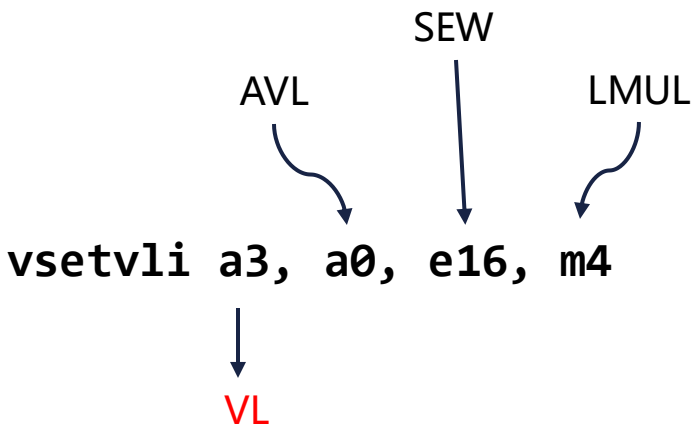


‘V’ 拓展的动态配置特性

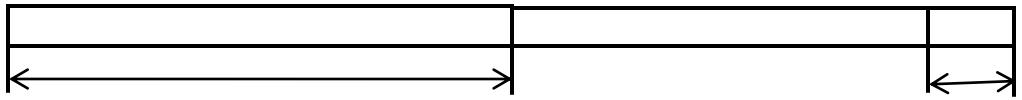
Buddy Compiler

‘V’ 拓展中的配置指令(vsetvli / vsetivli / vsetvl)

```
vsetvli rd, rs1, vtypei   # rd = new vl, rs1 = AVL, vtypei = new vtype setting
vsetivli rd, uimm, vtypei # rd = new vl, uimm = AVL, vtypei = new vtype setting
vsetvl  rd, rs1, rs2      # rd = new vl, rs1 = AVL, rs2 = new vtype value
```



LMUL = 2
（一次向量操作所需的向量寄存器个数，可动态配置）



VLEN （向量寄存器的位数）

SEW（单个元素的位数）

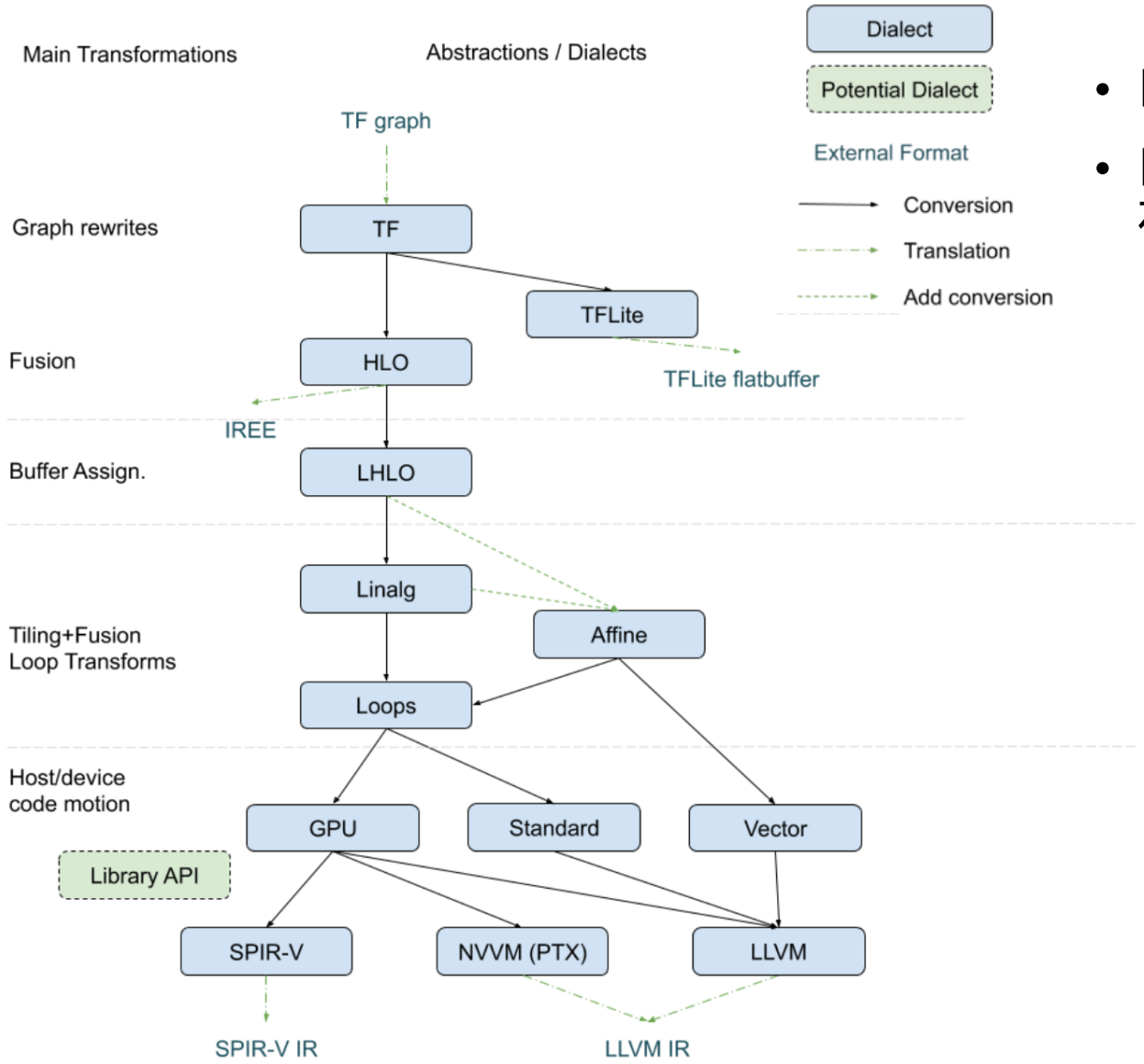
VL = min{ AVL, VLEN x LMUL / SEW }

（最终的向量长度由硬件信息、可配置参数共同决定）

合理配置向量长度可在有限硬件资源的限制下充分向量化，是性能提升的关键！

多级中间表示编译框架MLIR

Buddy Compiler

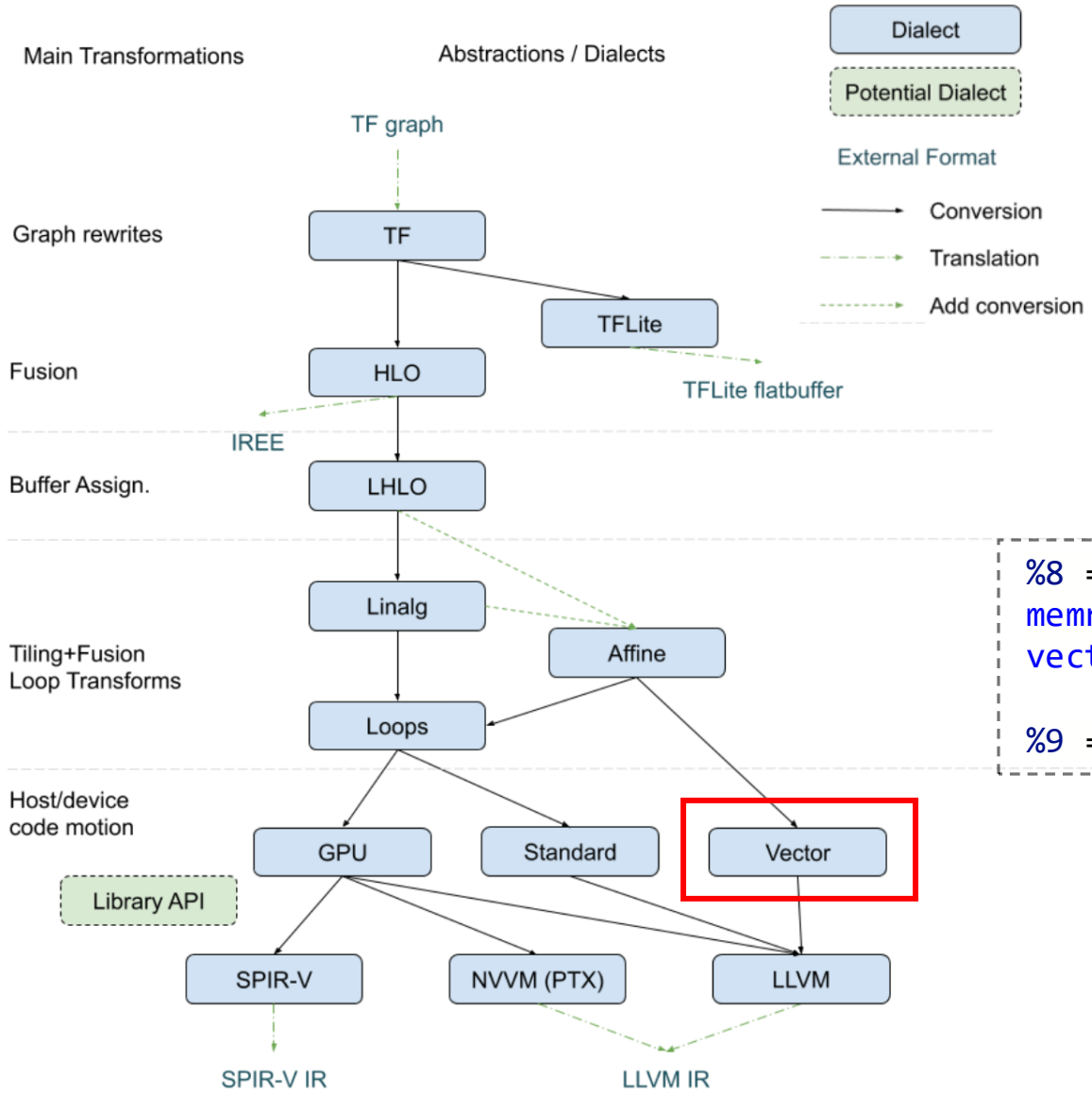


- Buddy Compiler的实现依托MLIR
- MLIR具有对应不同抽象级别的“方言”，在“方言”逐级下降的过程中实施优化

MLIR Lowering Paths - https://mlir.llvm.org/docs/Dialects/Vector/

多级中间表示编译框架MLIR

Buddy Compiler



- Buddy Compiler的实现依托MLIR
- MLIR具有对应不同抽象级别的“方言”，在“方言”逐级下降的过程中实施优化

MLIR的向量化抽象：Vector方言

```
%8 = vector.maskedload %arg2[%arg3, %arg6, %4], %2, %0 :
memref<?x?x?xi32>, vector<8xi1>, vector<8xi32> into
vector<8xi32>

%9 = arith.muli %7, %5 : vector<8xi32>
```

Vector方言的语法局限：作为定长向量类型，无法发挥‘V’拓展的动态配置特性

MLIR Lowering Paths - https://mlir.llvm.org/docs/Dialects/Vector/

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（一）添加‘V’拓展方言，支持向量长度配置和向量计算

MLIR的语法局限：读取向量前需要固定向量长度，后续无法修改

```
%5 = memref.load %A[%arg3, %arg6, %arg5] : memref<?x?x?xi32>
%6 = vector.broadcast %5 : i32 to vector<4xi32>
%7 = affine.vector_load %C[%arg3, %arg6, %arg4 * 4] : memref<?x?x?xi32>,
```

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（一）添加‘V’拓展方言，支持向量长度配置和向量计算

MLIR的语法局限：读取向量前需要固定向量长度，后续无法修改

```
%5 = memref.load %A[%arg3, %arg6, %arg5] : memref<?x?x?xi32>
%6 = vector.broadcast %5 : i32 to vector<4xi32>
%7 = affine.vector_load %C[%arg3, %arg6, %arg4 * 4] : memref<?x?x?xi32>,
```

⇩

Buddy Compiler：设计方言支持向量长度配置和动态向量运算

```
        %vl = rvv.setvl %avl, %sew, %lmul : index
%mul_vector = rvv.mul %input_vector, %aEle, %vl :
vector<[4]xi32>, i32, index
```

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（一）添加‘V’拓展方言，支持向量长度配置和向量计算

MLIR的语法局限：读取向量前需要固定向量长度，后续无法修改

```
%5 = memref.load %A[%arg3, %arg6, %arg5] : memref<?x?x?xi32>
%6 = vector.broadcast %5 : i32 to vector<4xi32>
%7 = affine.vector_load %C[%arg3, %arg6, %arg4 * 4] : memref<?x?x?xi32>,
```

⇩

Buddy Compiler：设计方言支持向量长度配置和动态向量运算

```
%vl = rvv.setvl %avl, %sew, %lmul : index
%mul_vector = rvv.mul %input_vector, %aEle, %vl :
vector<[4]xi32>, i32, index
```

[RFC] Add RISC-V Vector Extension (RVV) Dialect

MLIR

zhanghb97 Aug 2021

Hi,

I am writing to propose a RISC-V Vector extension (RVV) dialect. The RISC-V vector extension v1.0 candidate 38 has been released. Currently, LLVM supports the stable release v0.10. RVV is rapidly emerging, I think applications and optimizations will benefit from its features, but RVV is absent in MLIR architectural-specific vector dialects now. In MLIR, there are two types of vector-related dialects:

- Virtual Vector level/General vector dialect: Vector Dialect
- Hardware Vector level/Architectural-specific dialects vector dialect: amx Dialect, x86-vector Dialect, arm-neon Dialect, and arm-sve Dialect.

This RFC proposes the initial RVV Dialect. Fortunately, the SVE dialect has explored scalable vector types and operations, allowing me to refer and simplify my implementation on the RVV side.

https://discourse.llvm.org/t/rfc-add-risc-v-vector-extension-rvv-dialect/4146

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（二） Vector方言支持动态语法，实现尾端处理向量化

MLIR的语法局限：循环中的尾端处理采用掩码方式，无法向量化

```
%8 = vector.maskedload %C[%arg3, %arg6, %4], %2, %0 : memref<?x?x?xi32>,
%tmp9 = arith.muli %7,%5: vector<4xi32>
%9 =arith.addi %tmp9,%8 : vector<4xi32>
vector.maskedstore %C[%arg3, %arg6, %4], %2, %9 : memref<?x?x?xi32>, vec
```

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（二） Vector方言支持动态语法，实现尾端处理向量化

MLIR的语法局限：循环中的尾端处理采用掩码方式，无法向量化

```
%8 = vector.maskedload %C[%arg3, %arg6, %4], %2, %0 : memref<?x?x?xi32>,
%tmp9 = arith.muli %7,%5: vector<4xi32>
%9 =arith.addi %tmp9,%8 : vector<4xi32>
vector.maskedstore %C[%arg3, %arg6, %4], %2, %9 : memref<?x?x?xi32>, vec
```

⇩

Buddy Compiler：采用predication intrinsic实现尾端向量化

```
%vec = vector_exp.predication %mask, %vl : vector<[4]xi1>, i32 {
  %ele = vector.load %m[%c0, %c0]: memref<8x8xi32>, vector<[4]xi32>
  vector.yield %ele : vector<[4]xi32>
} : vector<[4]xi32>
```

Buddy Compiler对‘V’拓展的支持

Buddy Compiler

（二） Vector方言支持动态语法，实现尾端处理向量化

MLIR的语法局限：循环中的尾端处理采用掩码方式，无法向量化

```
%8 = vector.maskedload %C[%arg3, %arg6, %4], %2, %0 : memref<?x?x?xi32>,
%tmp9 = arith.muli %7,%5: vector<4xi32>
%9 =arith.addi %tmp9,%8 : vector<4xi32>
vector.maskedstore %C[%arg3, %arg6, %4], %2, %9 : memref<?x?x?xi32>, vec
```

⇩

Buddy Compiler：采用predication intrinsic实现尾端向量化

```
%vec = vector_exp.predication %mask, %vl : vector<[4]xi1>, i32 {
  %ele = vector.load %m[%c0, %c0]: memref<8x8xi32>, vector<[4]xi32>
  vector.yield %ele : vector<[4]xi32>
} : vector<[4]xi32>
```

[RFC] Dynamic Vector Semantics for the MLIR Vector Dialect

MLIR

zhanghb97 Dec 2023

Authors: Hongbin Zhang (ISCAS PLCT Lab) & Diego Caballero (Google)

Brief Summary

This proposal extends the Vector dialect with the concept of dynamic vectors (i.e., vectors whose length may arbitrarily vary at runtime). It defines a dynamic vector type (e.g., `vector<?xf32>`) and two operations (`vector.get_vl` and `vector.set_vl`) to manipulate dynamic vectors.

The main focus of our proposal is to properly define the semantics of dynamic vectors. We present three generic use cases as an example of applicability but they shouldn't prescribe or limit their usage. We also showcase RVV (RISC-V Vector Extensions) and its vector-length agnostic (VLA) model as a specific end-to-end application example. However, we envision further applicability of dynamic vectors and custom lowerings to other targets that we may explore in the future.

The dynamic vector representation seamlessly integrates with existing Vector dialect features like scalable vectors, vector masking and Linalg tiling-based vectorization.

Furthermore, we introduce an initial RVV Dialect that interfaces with the `vector.get_vl` and `vector.set_vl` operations to facilitate the lowering to RVV in LLVM. We leverage existing LLVM Vector Predication (VP) operations to model specific functionality for RVV but also reusability for future targets.

https://discourse.llvm.org/t/rfc-dynamic-vector-semantics-for-the-mlir-vector-dialect/75704

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成’ V’ 拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成‘V’拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

（一）优化场景来自实际AI模型

- ✓ MobileNetV3
- ✓ Meta/Llama2
- ✓ OpenAI/Whisper

…

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成‘V’拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

（一）优化场景来自实际AI模型

- ✓ MobileNetV3
- ✓ Meta/Llama2
- ✓ OpenAI/Whisper

…

（二）优化对象覆盖不同层级

- ✓ 模型层：BabyLlama…
- ✓ 模块层：Attention Layer…
- ✓ 算子层：batch_matmul…

…

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成‘V’拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

（一）优化场景来自实际AI模型

- ✓ MobileNetV3
- ✓ Meta/Llama2
- ✓ OpenAI/Whisper

…

（二）优化对象覆盖不同层级

- ✓ 模型层：BabyLlama…
- ✓ 模块层：Attention Layer…
- ✓ 算子层：batch_matmul…

…

（三）优化方式集成多种策略

- ✓ ‘V’拓展向量优化
- ✓ 通用张量优化
- ✓ 图级别优化
- ✓ …

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成‘V’拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

（一）优化场景来自实际AI模型

- ✓ MobileNetV3
- ✓ Meta/Llama2
- ✓ OpenAI/Whisper

…

（二）优化对象覆盖不同层级

- ✓ 模型层：BabyLlama…
- ✓ 模块层：Attention Layer…
- ✓ 算子层：batch_matmul…

…

（三）优化方式集成多种策略

- ✓ ‘V’拓展向量优化
- ✓ 通用张量优化
- ✓ 图级别优化
- ✓ …

（四）优化平台包括多种硬件

- ✓ CanMV-K230
- ✓ SpacemiT K1
- ✓ SiFive X280
- ✓ CHIPS Alliance T1 VPU

…

Buddy Compiler的‘V’拓展Benchmark

Buddy Compiler

构建可生成‘V’拓展的MLIR示例Benchmark，面向多种RISC-V硬件平台

PyTorch

（一）优化场景来自实际AI模型

- ✓ MobileNetV3
- ✓ Meta/Llama2
- ✓ OpenAI/Whisper

…

（二）优化对象覆盖不同层级

- ✓ 模型层：BabyLlama…
- ✓ 模块层：Attention Layer…
- ✓ 算子层：batch_matmul…

…

（三）优化方式集成多种策略

- ✓ ‘V’拓展向量优化
- ✓ 通用张量优化
- ✓ 图级别优化
- ✓ …

（四）优化平台包括多种硬件

- ✓ CanMV-K230
- ✓ SpacemiT K1
- ✓ SiFive X280
- ✓ CHIPS Alliance T1 VPU

…

算子层参数最佳配置示例

```
[root@k231 /sharefs ]#./conv2d-nchw-fchw-benchmark
1970-01-30T23:48:07+00:00
Running ./conv2d-nchw-fchw-benchmark
Run on (1 X 39.0224 MHz CPU )
Load Average: 0.50, 0.16, 0.06
-------------------------------------------------------------------
Benchmark                         Time             CPU   Iterations
-------------------------------------------------------------------
Conv2DNchwFchw                 6585 ms         6584 ms            1
Conv2DNchwFchwVector8          1466 ms         1465 ms            1
Conv2DNchwFchwVector16         1014 ms         1014 ms            1
Conv2DNchwFchwVector32          735 ms          735 ms            1
Conv2DNchwFchwVector64         1059 ms         1059 ms            1
Conv2DNchwFchwVector128        2338 ms         2337 ms            1
```

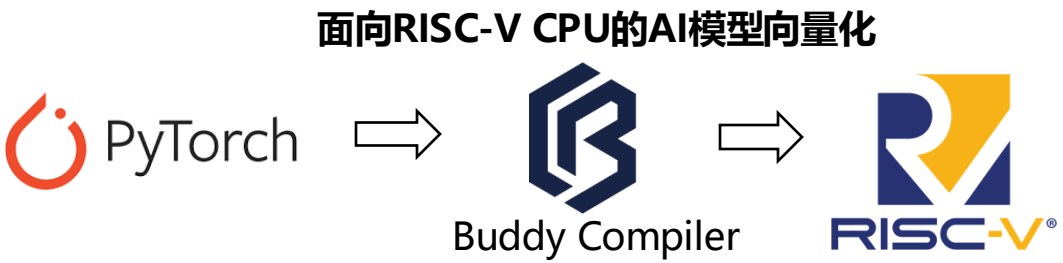
模型层TinyLlama-B1.1端到端推理示例

```
702 ○  ./dl-model-tinyllama-benchmark
2024-08-21T17:43:40+08:00
Running ./dl-model-tinyllama-benchmark
Run on (8 X 1600 MHz CPU s)
CPU Caches:
  L1 Instruction 32 KiB (x8)
  L1 Data 32 KiB (x8)
  L2 Unified 512 KiB (x2)
Load Average: 3.77, 5.80, 6.60
-------------------------------------------------------------------------------------------
Benchmark                                                 Time             CPU   Iterations
-------------------------------------------------------------------------------------------
BM_TinyLlama_V3/BM_TinyLlama_V3_Auto_Vectorization     367386 ms       367189 ms            1
BM_TinyLlama_V3/BM_TinyLlama_V3_Vectorization          371202 ms       371005 ms            1
```

总结： Buddy Compiler的优化实践

Buddy Compiler

✓支持AI模型面向RISC-V CPU的向量化推理

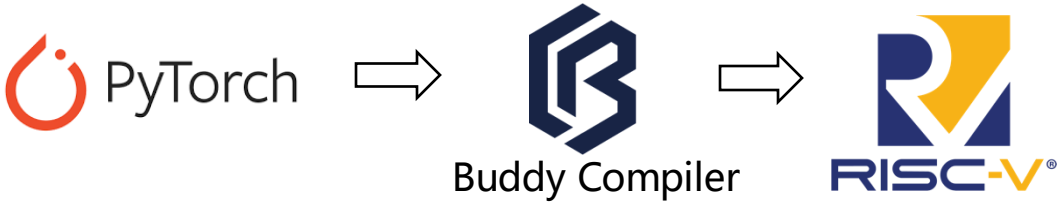


总结： Buddy Compiler的优化实践

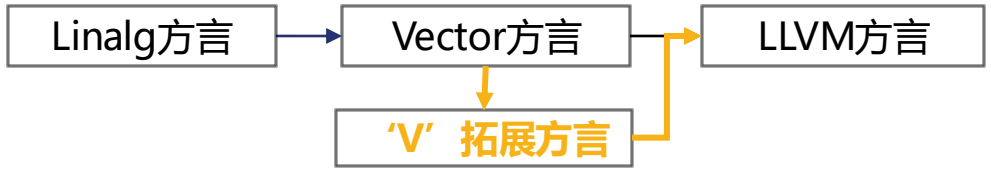
Buddy Compiler

- ✓支持AI模型面向RISC-V CPU的向量化推理
- ✓添加‘V’拓展自定义方言和语法支持

面向RISC-V CPU的AI模型向量化



依托MLIR的多级别方言发挥‘V’拓展向量化潜力

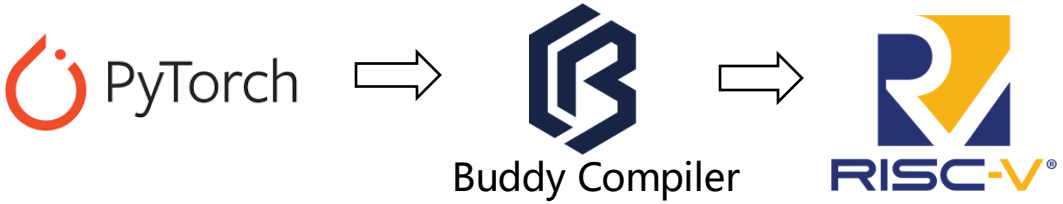


总结： Buddy Compiler的优化实践

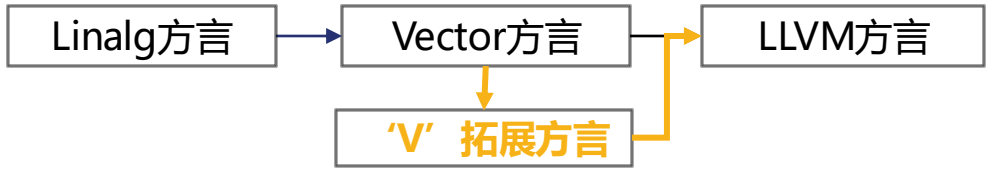
Buddy Compiler

- ✓支持AI模型面向RISC-V CPU的向量化推理
- ✓添加‘V’拓展自定义方言和语法支持
- ✓构建基于‘V’拓展的多层级Benchmark

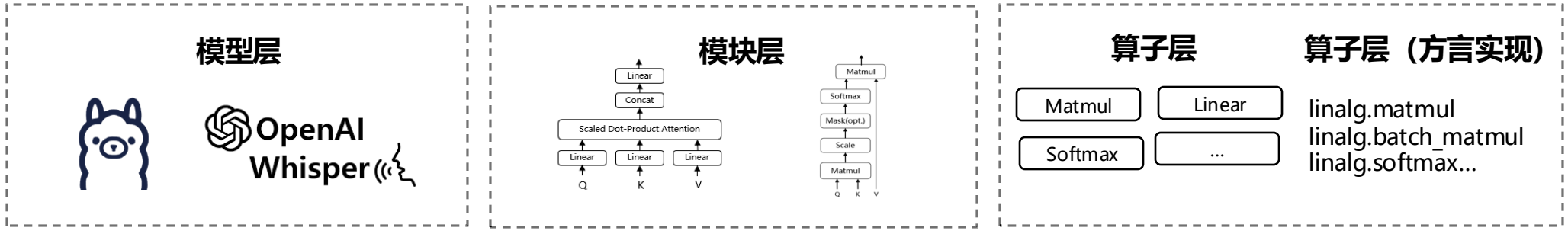
面向RISC-V CPU的AI模型向量化



依托MLIR的多级别方言发挥‘V’拓展向量化潜力



基于‘V’拓展的多层级Benchmark



“

Buddy Compiler致力于建设基于MLIR和RISC-V的软件生态，我们的愿景是解锁AI软件栈协同设计的无限可能！

”

Buddy Compiler相关链接

Buddy Compiler

主页：https://buddy-compiler.github.io/

GitHub：https://github.com/buddy-compiler

Slack：https://buddycompiler.slack.com/

EuroLLVM 2023报告：https://www.youtube.com/watch?v=EELBpBA-XCE

CGO C4ML Workshop 2024报告：https://c4ml.org/c4ml-2024

[MLIR论坛RFC] 添加‘V’拓展方言：

https://discourse.llvm.org/t/rfc-add-risc-v-vector-extension-rvv-dialect/4146

[MLIR论坛RFC] Vector方言支持动态语法：

https://discourse.llvm.org/t/rfc-dynamic-vector-semantics-for-the-mlir-vector-dialect/75704

Buddy Compiler

谢谢！