RFC: A new divergence analysis for LLVM

Simon Moll, Thorsten Klößner and Sebastian Hack

http://compilers.cs.uni-saarland.de

Compiler Design Lab
Saarland University
Saarland Informatics Campus

Recap: VPlan+RV

- VPlan: new vectorization infrastructure for LLVM.
  $\rightarrow$ under development.

Recap: VPlan+RV

- VPlan: new vectorization infrastructure for LLVM.
  $\rightarrow$ under development.
- RV: The Region Vectorizer `github.com/uni-saarland/rv`
  $\rightarrow$ Vectorizer for outer loops and whole functions.
  $\rightarrow$ available today!

Recap: VPlan+RV

- VPlan: new vectorization infrastructure for LLVM.
  → under development.
- RV: The Region Vectorizer `github.com/uni-saarland/rv`
  → Vectorizer for outer loops and whole functions.
  → available today!
- VPlan+RV: Bring RV's analyses and transformations to VPlan.
  → Today: *Divergence Analysis*
  → Coming up: *Partial Control-Flow Linearization* (PLDI '18).

DivergenceAnalysis

```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

DivergenceAnalysis

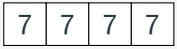
```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized

DivergenceAnalysis

```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized



DivergenceAnalysis

```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized

7	7	7	7

-1	1	6	2

DivergenceAnalysis

```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized

7	7	7	7
-1	1	6	2

- Integrated with LoopVectorizer (vplan-rv fork).
  → Not much to do: only single block loops with LLVM's LV
  → unit tests show what's possible.

DivergenceAnalysis

```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized

7	7	7	7
-1	1	6	2

- Integrated with LoopVectorizer (vplan-rv fork).
  → Not much to do: only single block loops with LLVM's LV
  → unit tests show what's possible.
- Won't be required by VPlan before patch series #3.

DivergenceAnalysis

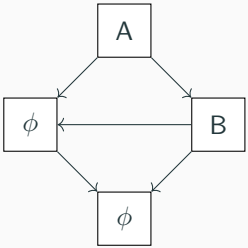
```
for (int i = 0; i < n; ++i) {
  for (int j = 0; j < m; ++j) {
    uni_var = f(i);
    varying_var = foo(i) + bar(j);
  }
}
```

vectorized

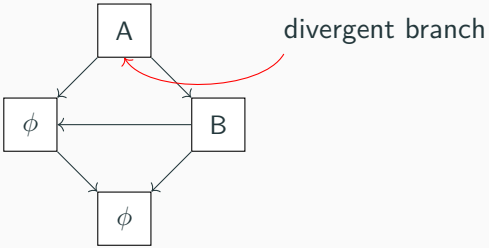
7	7	7	7
-1	1	6	2

- Integrated with LoopVectorizer (vplan-rv fork).
  → Not much to do: only single block loops with LLVM's LV
  → unit tests show what's possible.
- Won't be required by VPlan before patch series #3.
  → until then, let's fix LLVM's DivergenceAnalysis for GPUs.

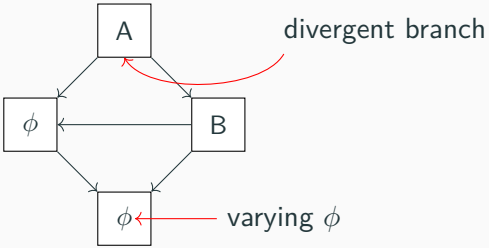
LLVM's DivergenceAnalysis (NVPTX/AMDGPU)



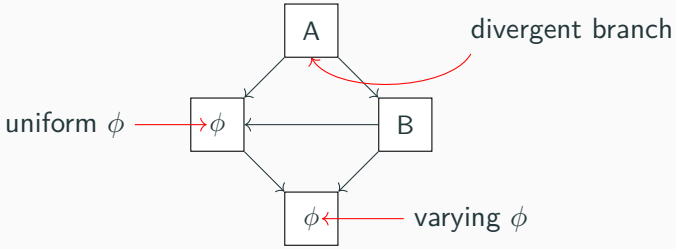
LLVM's DivergenceAnalysis (NVPTX/AMDGPU)



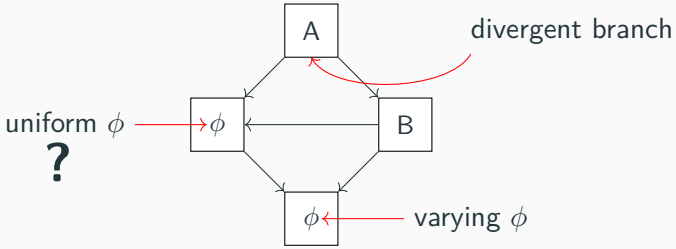
LLVM's DivergenceAnalysis (NVPTX/AMDGPU)



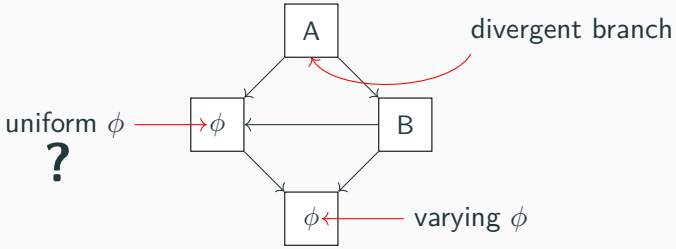
LLVM's DivergenceAnalysis (NVPTX/AMDGPU)



LLVM's DivergenceAnalysis (NVPTX/AMDGPU)

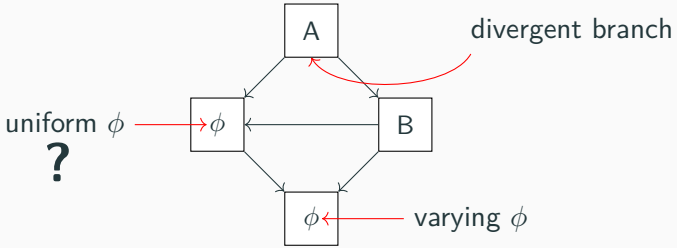


LLVM's DivergenceAnalysis (NVPTX/AMDGPU)

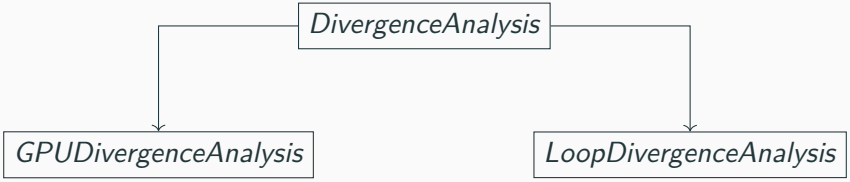


- LLVM's DivergenceAnalysis invalid for unstructured CFGs.

LLVM's DivergenceAnalysis (NVPTX/AMDGPU)



- LLVM's DivergenceAnalysis invalid for unstructured CFGs.
- Our analysis supports unstructured control.



NVPTX/AMDGPU
StructurizeCFG
`-use-rv-da`

LoopVectorizer
`-vectorizer-use-da`

Available at `github.com/cdl-saarland/vplan-rv`