INSTITUTO TECNOLOGICO DE IZTAPALAPA

MATERIA: LENGUAJES Y AUTOMATAS II

TEMA: MLIR

NOMBRE DEL EQUIPO: RIÑON

NTEGRANTES DEL EQUIPO:
CORTEZ GARCIA JESSICA GABRIELA
GUERRERO MENDEZ FABIOLA GUADALUPE
MARTINEZ ORTIZ ARELI SUSANA
MARTINEZ RODRIGUEZ GERARDO EMMANUEL

MLIR

¿QUE ES MLIR?

MLIR es un IR común que también admite operaciones específicas de hardware. Por lo tanto, cualquier inversión en la infraestructura que rodea a MLIR (por ejemplo, el compilador transmite ese trabajo) debería producir buenos rendimientos; muchos objetivos pueden utilizar esa infraestructura y se beneficiarán de ella.

MLIR es una representación poderosa, pero tampoco tiene objetivos. No intentamos admitir algoritmos de generación de código de máquina de bajo nivel (como la asignación de registros y la programación de instrucciones). Se adaptan mejor a optimizadores de nivel inferior (como LLVM). Además, no pretendemos que MLIR sea un lenguaje fuente en el que los propios usuarios finales escribirían núcleos (análogo a CUDA C ++). Por otro lado, MLIR proporciona la columna vertebral para representar cualquier DSL e integrarlo en el ecosistema.

¿PARA QUE SIRVE MLIR?

MLIR está diseñado para ser un IR híbrido que puede soportar múltiples requisitos diferentes en una infraestructura unificada. Por ejemplo, esto incluye:

- La capacidad de representar gráficos de flujo de datos (como en TensorFlow), incluidas las formas dinámicas, el ecosistema operativo extensible por el usuario, las variables de TensorFlow, etc.
- Las optimizaciones y transformaciones se realizan normalmente en tales gráficos (por ejemplo, en Grappler).
- Capacidad para albergar optimizaciones de bucle de estilo informático de alto rendimiento en los núcleos (fusión, intercambio de bucle, ordenamiento en teselas, etc.) y para transformar diseños de memoria de datos.
- Transformaciones de "reducción" de generación de código, como inserción DMA, gestión explícita de caché, ordenamiento en teselas de memoria y vectorización para arquitecturas de registro 1D y 2D.
- Capacidad para representar operaciones específicas de un objetivo, por ejemplo, operaciones de alto nivel específicas de un acelerador.
- Cuantización y otras transformaciones de gráficos realizadas en un gráfico de aprendizaje profundo.
- Primitivas poliédricas .
- Herramientas de síntesis de hardware / HLS .

INFRAESTRUCTURA DEL COMPILADOR

El marco MLIR fomenta las mejores prácticas existentes, por ejemplo, escribir y mantener una especificación de IR, construir un verificador de IR, brindar la capacidad de volcar y analizar archivos MLIR en texto, escribir pruebas unitarias extensas con la herramienta FileCheck y construir la infraestructura como un conjunto de bibliotecas modulares que se pueden combinar de nuevas formas.

Otras lecciones se han incorporado e integrado en el diseño de manera sutil. Por ejemplo, LLVM tiene errores de diseño no obvios que impiden que un compilador multiproceso funcione en múltiples funciones en un módulo LLVM al mismo tiempo. MLIR resuelve estos problemas al tener un alcance SSA limitado para reducir las cadenas de definición de uso y al reemplazar las referencias de función cruzada con explícitas symbol reference .

Para citar MLIR, utilice esta publicación .

```
@INPROCEEDINGS{9370308,
  author={C. {Lattner} and M. {Amini} and U. {Bondhugula} and A. {Cohen} and A. {Davis} and J. {Pienaar}
and R. {Riddle} and T. {Shpeisman} and N. {Vasilache} and O. {Zinenko}},
  booktitle={2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)},
  title={MLIR: Scaling Compiler Infrastructure for Domain Specific Computation},
  year={2021},
  volume={},
  number={},
  pages={2-14},
  doi={10.1109/CGO51591.2021.9370308}}
```