Constant Aspect Ratio Tiling

Parametric Tiling is (sometimes) Polyhedral

Introduction

Tiling is a well-known effective transformation:
- Locality improvement, new level of granularity with parallelism opportunities.
- If the tile sizes are constant, polyhedral ($i = 1 + a + b$)

Parametric tiling: tiling were the tile size is a parameter
- Tile size can be picked at runtime (ex: autotuning)
- Non polyhedral ($i = a + b + c$)

Parametric tiling is usually embedded in the code generator:
- Fourier-Motzkin symbolic elimination
- Tile the bounding box of the iteration domain
- D-tiling [Kim, LPCT10] (computing the outset, inset, parallel)
- PrimeTile [Hartono, ICS10] (sequential), DynTile [Hartono, IDPDS10] and PTile [Baskaran, CGO10] (parallel)
- The transformations applied after parametric tiling must be “hard-coded”
  (ex: wavefront/rectangular parallelism [Athanasios, Kelly, LCPC13])

Constant Aspect Ratio Tiling

Parametric tiling using only one tile size parameter and a fixed aspect ratio for every dimensions.

\[
\begin{array}{c|c|c}
\alpha & \beta \\
\mid \hline
\gamma & \delta \\
\end{array}
\]

i = \delta \cdot b \cdot \alpha_1 + \beta_1 \delta_1 \text{ where } 0 \leq \delta_1 < \delta \\
\alpha_1: \text{ blocked indexes} \\
\beta_1: \text{ local indexes} \\
\delta_1: \text{ ratios} \\
Parameters: \beta_b = b, \alpha_1, \delta_1 \\
where \ 0 \leq \beta_b < b

- **Main benefit:** Under these constraints, we are polyhedral !
- **Mathematical foundation:** How do we manage polyhedral and affine function?

CART on affine function

We have two tilings: for the antecedent space and for the image space.

\[
\begin{array}{c|c|c|c|c}
\alpha & \beta & ii & jj \\
\mid \hline
\gamma & \delta & k & l \\
\end{array}
\]

- **Example:** $f : (i, j \rightarrow i + j)$, with a tiling of the antecedent domain $b \times b$ and of the image domain $b$.
- **How to obtain its CART version $\bar{\phi}(\alpha, \beta, ii, jj) = (\alpha', \beta')$ ?

Same method than for polyhedra (with equations) $\rightarrow \phi$ is a piecewise affine function

\[
d(\alpha, \beta, ii, jj) = \begin{cases} 
(\alpha + ii, \beta + jj) & \text{if } ii + jj < b \\
(\alpha + ii + 1, \beta + jj) & \text{if } ii + jj = b \\
\end{cases}
\]

In general, the piecewise affine function might have modulo constraints

- **Example:** $f : (i \rightarrow i)$ with a tiling of the antecedent domain of $b$ and of the image domain of $2b$

\[
d(\alpha, ii) = \begin{cases} 
(\alpha, ii) & \text{if } \alpha \mod 2 = 0 \\
(\alpha, ii + 1) & \text{if } \alpha \mod 2 = 1 \\
\end{cases}
\]

CART on polyhedron

Example: $D = \{ i, j \mid i + j \leq N - 1 \land j \leq M \land 0 \leq i, j \}$ with tiles of size $b \times b$.
- **How to obtain its CART version $\Delta = \{ \alpha, \beta, ii, jj \}$ ?

Let us focus on the first constraint:

\[
\begin{align*}
Nii - \alpha & = Njj - \beta \\
\Rightarrow & \alpha = \frac{Nii - Njj}{\beta} + 0 \\
Nii - \alpha & = \frac{Nii - Njj}{\beta} - 1 \geq 0 \\
& Nii - \alpha = \left\lfloor \frac{Nii - Njj}{\beta} \right\rfloor + 0 \\
& Nii - \alpha = \left\lfloor \frac{Nii - Njj}{\beta} \right\rfloor - 1 \geq 0 & \text{and } i1 = \left\lfloor \frac{Nii - Njj}{\beta} \right\rfloor \leq \left\lfloor \frac{Nii - Njj}{\beta} - 2 \right\rfloor
\end{align*}
\]

After analysing every constraints, we obtain a polyhedron per value of $k = (k, \ldots)$.

- $\Delta_1 = 0 \leq k_1 = 0$ \\
- $k_1 = -1 k_2 = 0$ \\
- $k_1 = 0 k_2 = -1$ \\
- $k_1 = -k_2 = -1$ \\
- $k_1 = -2 k_2 = -1$

We can reorganize this union of polyhedron to have only one polyhedron per tile:

- $Nii - \alpha = 0 \mid Mii - \beta = 0$ \\
- $Nii - \alpha = 0 \mid Mii - \beta \geq 1$ \\
- $Nii - \alpha = \beta \leq 1 \mid Mii - \beta = 0$ \\
- $Nii - \alpha = \beta \geq 2 \mid Mii - \beta \geq 1$

Extensions

- Tiling along non-canonical dimensions:
- **CART**
- **CoB**

- Several tile size parameters: two different tile size parameters must not interfere
  - Ex: matrix multiply with 3 tile size parameters

Conclusion and Future Work

- Standalone implementation (C++/Java): http://compasy-tools.ens-lyon.fr/
- Full CART transformation: currently being implemented in the AlphaZ compiler
- The code is still polyhedral after the CART transformation
  - Allow polyhedral analysis and optimization after parametric tiling
    (for example, we can reapply another level of tiling for free)
  - Used as the first step of the semantic tiling transformation

People Involved

Guillaume Iooss (ENS Lyon - CSU - PhD student)
guillaume.iooss[at]ens-lyon.fr

Sanjay Rajopadhye (CSU - Faculty Member)
sanjay.rajopadhye[at]colostate.edu

Christophe Alias (ENS Lyon/Inria - Faculty Member)
christophe.alias[at]ens-lyon.fr

Yun Zou (CSU - PhD student)
yun.zou[at]colostate.edu