ParameTrick: Coefficient Generalization for Faster Polyhedral Scheduling

Gianpietro Consolaro, Harenome Razanajato, Nelson Lossing, Denis Barthou, Zhen Zhang, Corinne Ancourt, Cedric Bastoul

MINES PARIS | PSL

HUAWEI

Contents

Polyhedral Optimization: Introduction

Objective: Loop based optimization

Interchange

Initial Code

```
for(i=0; i<N; i++){
 for(j=0; j<M; j++){
  A[j][i] = 0;
 }
}
```

Schedule Θ = $[i \quad j]$

Final Code

```
#pragma omp parallel for
for(j=0; j<M; j++){
 for(i=0; i<N; i++){
  A[j][i] = 0;
 }
}
```

Schedule Θ = $[j \quad i]$

Skewing

Initial Code

```
for(i=2; i<N; i++){
 for(j=2; j<N; j++){
  A[i][j] = A[i-1][j]+A[i][j-1];
 }
}
```

Schedule Θ = $[i \quad j]$

Final Code

```
for(i=2; i<N; i++){
 #pragma omp parallel for
 for(j=i; j<i+N-2){
  B[i][j]=B[i-1][j-1]+B[i][j-1];
 }
}
```

Schedule Θ = $[i \quad i+j-2]$

Fusion/Distribution

Initial Code

```
for(i=0; i<N; i++){
 for(j=0; j<N; j++){
  A[i][j]= A[i+1][j];
  B[i][j]=0;
 }
}
```

Schedule Θ(S0) = $[i \quad j \quad 0]$

Θ(S1) = $[i \quad j \quad 1]$

Final Code

```
for(i=0; i<N; i++)
 #pragma omp parallel for
 for(j=0; j<N; j++)
  A[i][j]= A[i+1][j];

#pragma omp parallel for
for(i=0; i<N; i++)
 for(j=0; j<N; j++)
  B[i][j]=0;
```

Schedule Θ(S0) = $[0 \quad i \quad j]$

Θ(S1) = $[1 \quad i \quad j]$

And much more…

MINES PARIS | PSL HUAWEI

Polyhedral Optimization: Automatic Optimization Pipeline

Input Code

```
for(i=0; i<10; i++){
 for(j=0; j<20; j++){
  A[j][i] = 0;
 }
}
```

Algebraic Abstraction

Domain

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 9 \\ 0 & 1 & 0 \\ 0 & -1 & 19 \end{bmatrix} \cdot \begin{bmatrix} i \\ j \\ 1 \end{bmatrix} \geq 0$$

Initial Schedule

$\Theta = [i \quad j]$

Dependencies

$\emptyset$

Polyhedral Scheduler

Output Schedule

$\Theta = [j// \quad i//]$

ILP

Code Generation

Final Code

```
#pragma omp parallel for
for(j=0; j<20; j++){
 for(i=0; i<10; i++){
  A[j][i] = 0;
 }
}
```

1. **Algebraic Abstraction:** from for loop based kernels, we extract the semantically important information
2. **Polyhedral Scheduler:** automatically finds a scheduling transformation (complete order of the iterations)
3. **Code generation:** generating the code corresponding to the scheduling transformation

Polyhedral Scheduler: Scheduling Space

Objective: finding an optimal (execution time) scheduling function

Scheduling function for the statement S:

$\Theta(S) = [\varphi_0(S) \ldots \varphi_k(S)]$ where $\varphi_i(S) = \overrightarrow{T_i(S)} \cdot \begin{bmatrix} \overrightarrow{it^S} \\ \vec{N} \\ 1 \end{bmatrix}$

The scheduler looks for the optimal $\overrightarrow{T_i(S)}$ *vector*

Example:

```
for(i=0; i<N; i++){
 for(j=0; j<N; j++){
  A[j][i] = 0; //S
 }
}
```

Schedule: $\Theta(S) = [\overset{\varphi_0(S)}{\boxed{i}}\ \overset{\varphi_1(S)}{\boxed{j}}]$

$$\varphi_0(S) = \overset{\overrightarrow{T_0(S)}}{\boxed{\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}}} \cdot \begin{bmatrix} i \\ j \\ N \\ 1 \end{bmatrix}$$

$$\varphi_1(S) = \overset{\overrightarrow{T_1(S)}}{\boxed{\begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}}} \cdot \begin{bmatrix} i \\ j \\ N \\ 1 \end{bmatrix}$$

It builds 1 ILP (Integer Linear Programming) problem for each scheduling dimension $[T_i(S_0) \ldots T_i(S_M)]$

MINES PARIS | PSL HUAWEI

Polyhedral Scheduler: Constraints

Legality Constraints: allows only transformations preserving the semantics

For each data-dependency $\delta_{S \to R}$

Legality: $\boldsymbol{\varphi_i}$**(R) -** $\boldsymbol{\varphi_i}$**(S)** >= 0

$$\overrightarrow{T_i(R)} \cdot \begin{bmatrix} \overrightarrow{it^R} \\ \vec{N} \\ 1 \end{bmatrix} - \overrightarrow{T_i(S)} \cdot \begin{bmatrix} \overrightarrow{it^S} \\ \vec{N} \\ 1 \end{bmatrix} >= 0$$

Not Affine!!

We apply Farkas Lemma + Fourier Motzkin Elimination to obtain constraints only on the $[\overrightarrow{T_i(R)}\ \overrightarrow{T_i(S)}]$ space

MINES PARIS | PSL HUAWEI

ParameTrick: Idea

Objective: simplifying constraints' numerical complexity

Input Code

```
for(i=0; i<=537; i++){
 c[i] = b;                //S
 for(j=0; j<537; j++){
  d[i][j] = c[i];         //R
 }
}
```

Original Domains

$$D_S = \begin{pmatrix} 1 & 0 \\ -1 & 537 \end{pmatrix} \begin{pmatrix} i^S \\ 1 \end{pmatrix} \geq 0$$

$$D_R = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 0 & 537 \\ 0 & 1 & 0 \\ 0 & -1 & 537 \end{pmatrix} \begin{pmatrix} i^R \\ j^R \\ 1 \end{pmatrix} \geq 0$$

Original Dependency

$$\delta_{S \to R} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 537 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 537 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 537 \\ 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} i^S \\ i^R \\ j^R \\ 1 \end{pmatrix} \geq 0$$

Legality Constraint

$$\begin{pmatrix} 0 & 0 & 537 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ -537 & 537 & 537 & -1 & 1 & 0 \\ -537 & 537 & 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} t\_i^S \\ t\_i^R \\ t\_j^R \\ t\_1^S \\ t\_1^R \\ 1 \end{pmatrix} \geq 0$$

ParameTrick

Substitute big coefficients with parametric constants

Let's substitute **537** by **N:**

Simplified Dependency

$$\delta_{S \to R} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} i^S \\ i^R \\ j^R \\ N \\ 1 \end{pmatrix} \geq 0$$

Legality Constraint

$$\begin{pmatrix} 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} t\_i^S \\ t\_i^R \\ t\_j^R \\ t\_N^S \\ t\_N^R \\ t\_1^S \\ t\_1^R \\ 1 \end{pmatrix} \geq 0$$

MINES PARIS | PSL

HUAWEI

ParameTrick: Idea

Not convinced?

2mm

```
for(i=0;i<3200;i++)
 for(j=0;j<3600;j++){
  tmp[i][j] = 0;
   for(k=0;k<4400;++k)
    tmp[i][j] += alpha*A[i][k]*B[k][j]; //S
 }

for(i=0;i<3200;i++)
 for(j=0;j<4800;j++) {
  D[i][j] *= beta;
  for(k=0;k<3600;++k)
   D[i][j] += tmp[i][k]*C[k][j];         //R
 }
```

Original

$$\text{Legality}(dep_{S\text{->}R}) \quad \begin{pmatrix} 0 & 0 & -4399 & 0 & 0 & 0 & -1 & 1 & 0 \\ -3199 & 0 & -4399 & 3199 & 0 & 0 & -1 & 1 & 0 \\ 0 & -3599 & -4399 & 0 & 0 & 3599 & -1 & 1 & 0 \\ -3199 & -3599 & -4399 & 3199 & 0 & 3599 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ -3199 & 0 & 0 & 3199 & 0 & 0 & -1 & 1 & 0 \\ 0 & -3599 & 0 & 0 & 0 & 3599 & -1 & 1 & 0 \\ -3199 & -3599 & 0 & 3199 & 0 & 3599 & -1 & 1 & 0 \\ 0 & 0 & -4399 & 0 & 4799 & 0 & -1 & 1 & 0 \\ -3199 & 0 & -4399 & 3199 & 4799 & 0 & -1 & 1 & 0 \\ 0 & -3599 & -4399 & 0 & 4799 & 3599 & -1 & 1 & 0 \\ -3199 & -3599 & -4399 & 3199 & 4799 & 3599 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 4799 & 0 & -1 & 1 & 0 \\ -3199 & 0 & 0 & 3199 & 4799 & 0 & -1 & 1 & 0 \\ 0 & -3599 & 0 & 0 & 4799 & 3599 & -1 & 1 & 0 \\ -3199 & -3599 & 0 & 3199 & 4799 & 3599 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} t\_i^S \\ t\_j^S \\ t\_k^S \\ t\_i^R \\ t\_j^R \\ t\_k^R \\ t\_1^S \\ t\_1^R \\ 1 \end{pmatrix} \geq 0$$

After ParameTrick

$$\text{Legality}(dep_{S\text{->}R}) \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} t\_i^S \\ t\_j^S \\ t\_k^S \\ t\_i^R \\ t\_j^R \\ t\_k^R \\ t\_N1^S \\ t\_N2^S \\ t\_N3^S \\ t\_N4^S \\ t\_N1^R \\ t\_N2^R \\ t\_N3^R \\ t\_N4^R \\ t\_1^S \\ t\_1^R \\ 1 \end{pmatrix} \geq 0$$

N1=4800

N2=4400

N3=3600

N4=3200

MINES PARIS | PSL HUAWEI

Results: PolyBench (using PolyTOPS scheduler)

We compared *Compilation time* between 2 variants of ParameTrick:

- **p-trick:** Using ParameTrick + Positivity Constraints ($N > 0$, $M > 0$)
- **p-trick-extra:** Using ParameTrick + Relation Constraints ($N > 0$, $M > 0$, $N > M$)

We also compared the *Execution time* of the final scheduling transformation:

- Speedup = 1 means that the same transformation was found

We used our scheduler PolyTOPS for the experiments, using isl-0.25 as ILP solver

HW Specification: Intel Xeon E5-2683 CPU(x86 64), 2 sockets, 16 cores per socket

G. Consolaro et al, PolyTOPS: Reconfigurable and Flexible Polyhedral Scheduler, CGO''24[Accepted]

MINES PARIS | PSL HUAWEI

Results: PolyBench (using PolyTOPS scheduler)

Compilation:

- Speedups (GMP? Fourier Motzkin?)
- Slowdowns (extra constraints and variables)

Execution:

- Identical transformations
- Speedups
- Slowdowns

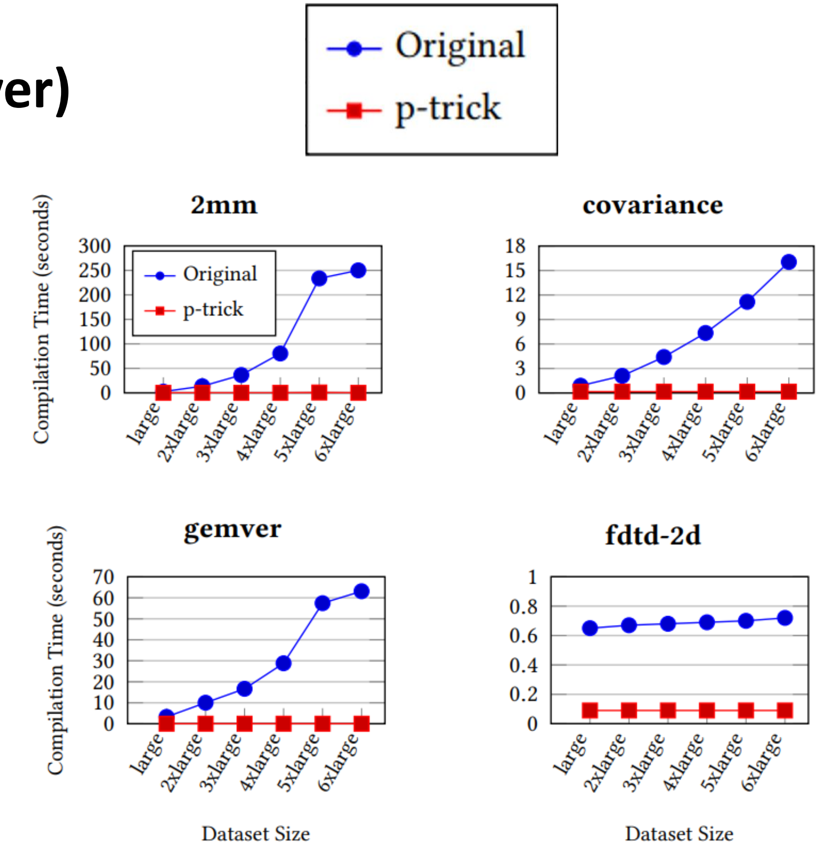
Case	Compilation			Execution	
	Original **Time** (ms)	**Speedup** (original / p-trick)	**Speedup** (original / p-trick-extra)	**Speedup** (original / p-trick)	**Speedup** (original / p-trick-extra)
3mm	>600000	**>3636.32**	>176.67	n.a.	n.a.
correlation	2201.4	**2.31**	0.09	**25.12**	1
cholesky	9032.07	**151.43**	147.72	2.05	**2.23**
lu	14774.03	263.04	**291.1**	1.61	1.61
floyd-warshall	11820.16	**391.19**	386.67	1.27	1.27
bicg	19.04	**0.91**	0.73	1	1
covariance	887.05	**5.63**	2.41	1	1
fdtd-2d	650.69	**6.86**	5.47	1	1
gemm	22.49	**1.01**	0.59	1	1
gemver	3282.38	**107.28**	106.12	1	1
gesummv	29.96	0.93	0.93	1	1
heat-3d	751.84	**4.41**	4.2	1	1
jacobi-1d	30.97	**1.23**	1.22	1	1
jacobi-2d	351.02	**5.33**	4.82	1	1
mvt	11.35	0.91	**0.94**	1	1
seidel-2d	51.89	**0.97**	0.9	1	1
syr2k	22.25	**0.99**	0.79	1	1
syrk	20.93	**0.92**	0.73	1	1
trisolv	23.15	**1.09**	1.08	1	1
trmm	31.14	**1.02**	0.61	1	1
2mm	2523.95	**35.93**	5.53	0.8	**1**
gramschmidt	274.84	**1.53**	0.32	0.79	0.79
symm	322.79	**4.55**	3.5	0.78	0.78
atax	59.94	**2.12**	1.28	0.76	**1**
doitgen	3984.24	**34.59**	7.6	0.08	0.08
durbin	195.04	**2.79**	2.67	0.00002	0.00002

Results: PolyBench

What happens if we increase the loop bounds? (isl-solver)

The compilation time is completely invariant to the loop bound values when usingParameTrick

Without ParameTrick, depending on the case, we can notice an important growth in terms of compilation time



MINES PARIS | PSL

HUAWEI

Results: MindSpore (using PolyTOPS and FPL solver)

MindSpore is an AI Framework that implements a polyhedral pipeline (AKG) using PolyTOPS scheduler

We applied ParameTrick to some AI operators from MindSpore

Case	Original Time (ms)	Time (ms) (p-trick)	Speedup (p-trick)
batch_norm	>600000	9764	**>61.45**
two2fractal_v1	96519	78	**1237.42**
two2fractal_v2	105	51	**2.05**
two2fractal_v3	344	28	**12.29**
maxpool_grad_v1	5583	2333	**2.39**
force_grad	381	180	**2.12**
max_pool_grad_v2	>600000	529	**>1134**
hpl_cholesky	9291	121	**76.78**
hpl_lu	29396	97	**303.05**

The scheduling transformation found is the same for these cases.
The execution time is identical.

MINES PARIS | PSL HUAWEI

Conclusion and Future Work

- We showed how ParameTrick can decrease tremendously the compilation time while losing only few optimization opportunities in practice
- This simple technique can be used to schedule cases that would be untreatable otherwise
- We indirectly showed that in some cases, Pluto cost function is definitely not enough. What is missing?
- Is there a way to understand if a kernel could benefit (or not) from ParameTrick?
- Further analysis about GMP impact would help explaining the compilation time reduction

Thank you.

See you at the Poster session :)

Bring digital to every person, home and organization for a fully connected, intelligent world.

Copyright©2024 Huawei Technologies Co., Ltd.
All Rights Reserved.

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.

MINES PARIS | PSL

HUAWEI