

## 循环优化系列第九讲

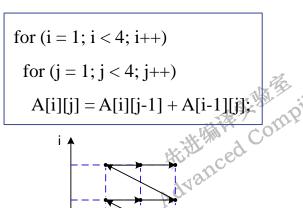
# 循环倾斜

嘉宾: 柴赟达



#### • 基础概念

循环倾斜是一种改变迭代空间形式的变换,用于挖掘循环中的并行潜能的优化方式,可以把存在的并行性用传统的并行循环的形式表示出来。

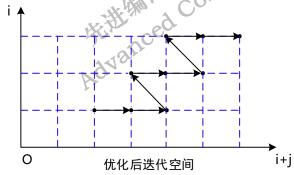


	•		A P	gya,	nced
0	原	始迭位	代空间	,	j

i=3,j=1 A31=A30+A21	i=3,j=2	i=3,j=3
7151-7150+7121	A32=A31+A22	A33=A32+A23
i=2,j=1	i=2,j=2	i=2,j=3
A21=A20+A11	A22=A21+A12	A23=A22+A13
i=1,j=1	i=1,j=2	i=1,j=3
A11=A10+A01	A12=A11+A02	A13=A12+A03

」 先进编译实验室 Advanced Compiler 循环倾斜

for $(i = 1; i < 4; i++)$
for $(j = i+1; j < i+4; j++)$
A[i][j-i] = A[i][j-i-1] + A[i-1][j-i];



		i=3,j=4	i=3,j=5	i=3,j=6
		A31=A30+A21	A32=A31+A22	A33=A32+A
	i=2,j=3	i=2,j=4	i=2.j=5	I=142
	A21=A20+A11	A22=A21+A12	A23=A22+A13	28) 118
i=1.j=2	i=1,j=3	i=1,j=4		
A11=A10+A01	A12=A11+A02	A13=A12+A03		625
•				

### 循环倾斜



A33=A32+A23

A32=A31+A22

A31=A30+A21

j=5, i=3

j=4, i=3

#### 基础概念

循环倾斜是一种改变迭代空间形式的变换,用于挖掘循环中的并行潜能的优化方式,可以把存在的并行性用传统的并行循环的形式表示出来。

for 
$$(i = 1; i < 4; i++)$$
  
for  $(j = 1; j < 4; j++)$   
 $A[i][j] = A[i][j-1] + A[i-1][j];$ 



for 
$$(i = 1; i < 4; i++)$$
  
for  $(j = i+1; j < i+4; j++)$   
 $A[i][j-i] = A[i][j-i-1] + A[i-1][j-i];$ 



for 
$$(j = 2; j < 8; j++)$$
for  $(i = max(1,j-4+1); i < min(j-1, 4); i++)$ 

$$A[i][j-i] = A[i][j-i-1] + A[i-1][j-i];$$

j=5, i=2

j=4, i=2

j=3, i=2

A23=A22+A13

A22=A21+A12

A21=A20+A11

			i=3,j=4	i=3,j=5	i=3,j=6
			A31=A30+A21	A32=A31+A22	A33=A32+A23
and and and		i=2,j=3	i=2,j=4	i=2.j=5	
	\	A21=A20+A11	A22=A21+A12	A23=A22+A13	
	i=1.j=2	i=1,j=3	i=1,j=4		
	/A11=A10+A01	A12=A11+A02	A13=A12+A03		

j=4, i=1

j=3, i=1

j=2, i=1

A13=A12+A03

A12 = A11 + A02

A11=A10+A01

先进编译实验室

**Advanced Compiler** 



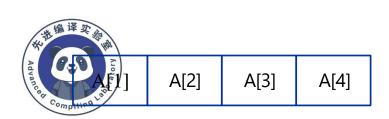
#### • 变换方式

for 
$$(i = 1; i < N; i++)$$
  
for  $(j = 1; j < N; j++)$   
 $A[i][j] = A[i-1][j] + A[i][j-1];$ 

循环倾斜

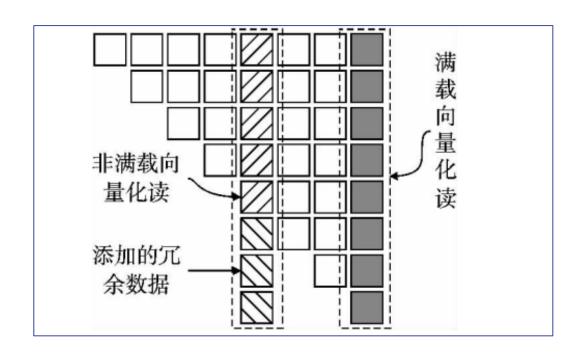
for 
$$(j = 2; j < 2 * N; j++)$$
  
for  $(i = max(1,j-N+1); i < min(N,j); i++)$   

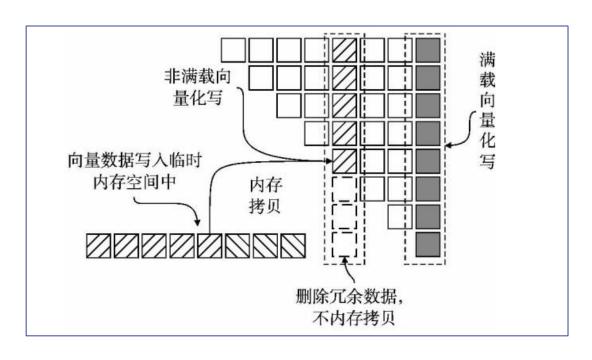
$$A[i][j - i] = A[i - 1][j - i] + A[i][j - i - 1];$$



0













#### 三层嵌套的循环倾斜

```
N=16,M=64,L=64;
for (i = 2; i < N + 1; i++)
    for (j = 2; j < M + 1; j++) {
       for (k = 1; k < L; k++)
         A[i][j][k] = A[i][j - 1][k] + A[i - 1][j][k]; 4 循环倾斜
         B[i][j][k+1] = B[i][j][k] + A[i][j][k];
```

```
for (i = 2; i < N + 1; i++)
 for (j = 2; j < M + 1; j++) {
  for (k = i + j + 1; k < i + j + L; k++)
      A[i][j][k-i-j] = A[i][j-1][k-i-j] + A[i-1][j][k-i-j];
      B[i][j][k - i - j + 1] = B[i][j][k - i - j] + A[i][j][k - i - j];
```

```
for (k = 2; k < M+L; k++)
    for (i = max(1, k - M - L - 1); i < min(N, k + L - 2); i++)
       for (j = max(1, k - i - L); j < min(M, k + i - 1); j++) {
         A[i][j][k-i-j] = A[i][j-1][k-i-j] + A[i-1][j][k-i-j];
          B[i][i][k - i - j] = B[i][i][k - i - j] + A[i][i][k - i - j];
```





## 分享完毕,感谢聆听!



#### 参考文献:

- [1] Optimizing Compilers for Modern Architectures: A Dependence-Based Approach [Book Review][J]. Computer,2002,35(4).
- [2] 多面体编译理论与深度学习实践.赵捷,李宝亮.清华大学出版社.2022.
- [3] 陈华军,王琦,洪超等.基于全局数据重组的循环倾斜优化[J].计算机与现代化,2017,No.262(06):45-49.



