先进编译实验室
Advanced Compiler

循环优化系列第六讲

循环分段

嘉宾：柴赟达

先进编译实验室
Advanced Compiler

循环分段

- **基础概念**

循环分段是将单层循环变换为两层嵌套循环，内层循环遍历的是迭代次数为strip的连续区域(或叫条带循环)，外层循环的步进单位为strip，这个strip就是分段因子，分段因子需要根据硬件情况选取。

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

循环分段

```
for(i=0; i<N; i+=k){    //并行执行
    for(j=i; j<=i+k-1; j++){
        A[j] = B[j] + C[j];    //向量执行
    }
}
```

假设有P个处理器用于执行循环，N次迭代，总共可分为N/P=K次迭代。

- **优点：**

① 充分利用多个处理器资源

② 将可用的[illegible]成更适合硬件的形式

先进编译实验室
Advanced Compiler

循环分段

- **优化效果**

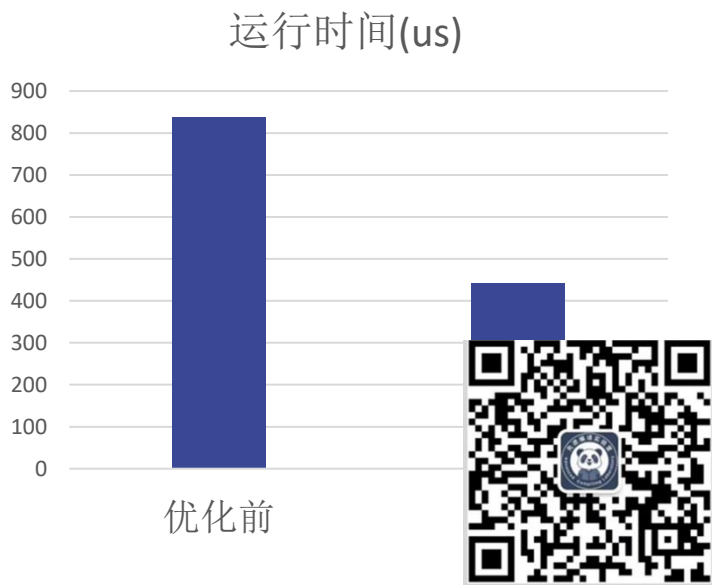
测试环境：Hygon C86 7185 32-core Processor；x86_64；

编译器版本：llvm-13

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#define N 204800
int main() {
  int i;
  float A[N], B[N], C[N];
  struct timeval  time_start, time_end;
  for (i = 0; i < N; i++) {
    A[i] = 1;
    B[i] = rand()%10;
    C[i] = rand()%10;
  }
   gettimeofday(&time_start, NULL);
  for (i = 0; i < N; i++)
    A[i] = B[i] + C[i];
  gettimeofday(&time_end, NULL);
   printf("unroll used time %ld us\n",
time_end.tv_usec - time_start.tv_usec);
  return A[7];
}
```

循环分段

```
#include <stdio.h>
#include <sys/time.h>
#include <x86intrin.h>
#define N 204800
int main() {
  __m128 ymm0, ymm1, ymm2, ymm3;
  float A[N], B[N], C[N];
  struct timeval  time_start, time_end;
  int i, j;
  for (i = 0; i < N; i++) {
    A[i] = 1;
    B[i] = rand()%10;
    C[i] = rand()%10;
  }
  int K = 32;
   gettimeofday(&time_start, NULL);
  for (i = 0; i < N; i += K) {
    for (j = i; j <= i + K - 1; j += 4) {
      ymm0 = _mm_load_ps(B + j);
      ymm1 = _mm_load_ps(C + j);
      ymm2 = _mm_add_ps(ymm0, ymm1);
      _mm_storeu_ps(A + j, ymm2);
    }
  }
  gettimeofday(&time_end, NULL);
   printf("unroll used time %ld us\n",
time_end.tv_usec - time_start.tv_usec);
  return A[7];
}
```



循环分段

- 不适用的情况

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

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]