Small Programs With Big Speedups

Stage avec Arthur Charguéraud

Joachim Bienvenüe

Inria

February 15th, 2024

Objectif

Trouver des petits programmes illustrants des speedups que l'on peut optenir avec des transformations de code

Swap

```
for(int i=0; i<n; i++){
  for(int j=0; j<n; j++){
    T[j][i]++;
  }
}
```

```
for(int j=0; j<n; j++){
  for(int i=0; i<n; i++){
    T[j][i]++;
  }
}
```

Speedup : 6x

Transposition With Tile 2d

```
for(int i=0; i<n; i++){
  for(int j=0; j<n; j++){
    ADDR(V, n, j, i) =
    ADDR(T, n, i, j);
  }
}
```

```
const int L3Space = 4194304;
int blockSize = L3Space/(n<<5);

//we want that blockSize divide n
while(n%blockSize)blockSize--;//blokSize divide n
for(int iBlock=0; iBlock<n; iBlock += blockSize){
  for(int jBlock=0; jBlock<n; jBlock += blockSize){
    for(int i=0; i<blockSize; i++){
      for(int j=0; j<blockSize; j++){
        ADDR(V, n, iBlock+i, jBlock+j) =
        ADDR(T, n, jBlock+j, iBlock+i);
      }
    }
  }
}
```

Speedup : 10.8x

Matrix Multiplication With Tile 2d

```
for(int i=0; i<n; i++){
  for(int j=0; j<n; j++){
    ADDR(V, n, i, j) = 0;
    for(int k=0; k<n; k++){
      ADDR(V, n, i, j) +=
        ADDR(T, n, i, k)*
        ADDR(Ut, n, j, k);
    }
  }
}
```

```
const int L3Space = 4194304;
//traverse the list block by block
int blockSize = L3Space/(n<<5);

//we want that blockSize divide n
while(n%blockSize)blockSize--;//blokSize divide n
for(int iBlock=0; iBlock<n; iBlock += blockSize){
  for(int jBlock=0; jBlock<n; jBlock += blockSize){
    for(int i=0; i<blockSize; i++){
      for(int j=0; j<blockSize; j++){
        ADDR(V, n, iBlock+i, iBlock+j) = 0;
        for(int k=0; k<n; k++){
          ADDR(V, n, iBlock+i, iBlock+j) +=
            ADDR(T, n, iBlock+i, k)*
            ADDR(Ut, n, iBlock+j, k);
        }
      }
    }
  }
}
```

Speedup : 2.1x

Aos2Soa

```
typedef struct{
  int data[32];
}point;

for(int i=0; i<n; i++){
  ps[i].data[0]++;
}
```

```
for(int i=0; i<n; i++){
  pss[0][i]++;
}
```

Speedup : 20.2x

Conditionnal Loop Spliting

```
for(int j=0; j<100; j++){
  for(int i=0; i<n; i++){
    if(sin(j)<=0)
      T[i] += 1;
    else
      U[i] += 2;
  }
}
```

```
for(int j=0; j<100; j++){
  if(sin(j) <= 0){
    for(int i=0; i<n; i++){
      T[i] += 1;
    }
  }else{
    for(int i=0; i<n; i++){
      U[i] += 2;
    }
  }
}
```

Speedup : 8.3x

Loop fusionning

```
int s=0;
for(int i=0; i<n; i++){
  T[i] = i;
}
for(int i=0; i<n; i++){
  s += T[i];
}
```

```
int s=0;
for(int i=0; i<n; i++){
  T[i] = i;
  s += T[i];
}
```

Speedup : 1.15x

Inline

```
void
__attribute__ ((noinline))
f(int i, int* x){
  (*x) += i;
}
```

```
int x=0;
for(int i=0; i<n; i++){
  f(i, &x);
}
```

```
int s=0;
for(int i=0; i<n; i++){
  T[i] = i;
  s += T[i];
}
```

Speedup : 13x