**به نام خدا**

تمرین هفتم درس سیستم‌های نهفته بی‌درنگ

****

نیمسال دوم 1401-1402

سوال1-

الف)

Loop Fusion:

در این روش به اینصورت است که اگر دو حلقه بصورت تکراری باشند باهم ترکیب میشوند.

میتوانیم حلقه بالایی را با حلقه پایین ترکیب کنیم:

for (i = 0; i < N; i++) {

D[i] = A[i+1] \* 2;

for (j = 0; j < N; j++) {

if (i>0){

A[j] = B[i] + C[i-1];

}else {

A[j] = B[i];

}

}

}

ترکیب به این صورت است که D[i] = A[i+1] \* 2; میتواند وارد حلقه i شود و حلقه بالایی حذف گردد همچنین گاهی اوقات حلقه j اجرا بشود که این حلقه وارد else-if ها شده است.

ب)Loop unrolling:

با استفاده از الگوریتم loop unrolling، حلقه تکراری داخلی کد با افزایش سرعت اجرای برنامه با تکرار دستورات حلقه به صورت زیر بهینه سازی می شود:

for (i = 0; i < N; i++){

D[i] = A[i+1] \* 2;

for (j = 0; j < N; j+=2) {

if (i>0) {

A[j] = B[i] + C[i-1];

A[j+1] = B[i] + C[i-1];

} else {

A[j] = B[i];

A[j+1] = B[i];

}

}

}

تعداد دستورات داخل حلقه را 2تا کردیم و بجای ان حلقه n/2 بار طی می شود.

ج)Loop splitting/nesting:

if (N > 0) {

      for (j = 0; j < N; j++) {

            A[j] = B[0];

A[j+1] = B[0];

      }

}

for (i = 1; i < N; i++) {

D[i] = A[i+1] \* 2;

      for (j = 0; j < N; j++) {

            A[j] = B[i] + C[i-1];

A[j+1] = B[i] + C[i-1];

      }

}

حلقه اول بدون تغییر باقی می ماند. و حلقه دوم به دو قسمت تقسیم می شود: یکی برای مواردی که N بزرگتر از صفر است و دیگری برای بقیه موارد. دلیلش این است که شرط if در داخل حلقه فقط به i بستگی دارد که در طول حلقه تغییر نمی کند. با تقسیم حلقه، از بررسی شرایط یکسان N بار جلوگیری می کنیم. - در قسمت اول حلقه تقسیم، تمام عناصر A را روی B[0] قرار می دهیم که معادل مورد else حلقه اصلی است. فقط باید یک بار به جای N بار این کار را انجام دهیم. - در قسمت دوم حلقه تقسیم، عناصر A را مجموع B[i] و C[i-1] قرار می دهیم که معادل if حلقه اصلی است. از i=1 شروع می کنیم زیرا قبلاً در حلقه قبلی A را روی B[0] قرار داده ایم.

د)Loop tiling/blocking:

int block\_size = 16;

for (int ii = 0; ii < N; ii += block\_size) {

for (int jj = 0; jj < N; jj += block\_size) {

for (int i = ii; i < min(ii + block\_size, N); i++) {

D[i] = A[i+1] \* 2;

}

for (int i = ii; i < min(ii + block\_size, N); i++) {

for (int j = jj; j < min(jj + block\_size, N); j++) {

if (i > 0) {

A[j] = B[i] + C[i-1];

} else {

A[j] = B[i];

}

}

}

}

}

در اینجا حلقه ‌های بیرونی روی بلوک‌ ها تکرار می‌شوند، در حالی که حلقه‌های داخلی روی عناصر درون هر بلوک تکرار می‌شوند. ما مقادیر "D" را در اولین حلقه داخلی و مقادیر "A" را در حلقه داخلی دوم محاسبه می کنیم. ما از «min» استفاده می‌کنیم تا اطمینان حاصل کنیم که هنگام تکرار روی بلوک‌ها از محدوده خارج نمی‌شویم.