**ijk loop**

for (i=0; i<n; i++)

for (j=0; j<n; j++)

{

c[i] [j] =0;

for (k=0; k<n; k++)

c[i] [j] +=a[i] [k]\*b[k] [j];

}

**jik loop**

for (j=0; j<n; j++)

for (i=0; i<n; i++)

{

c[j][i] =0;

for (k=0; k<n; k++)

c[i] [j] +=a[i] [k]\*b[k] [j];

}

**kij loop**

for (k=0; k<n; k++)

for (i=0; i<n; i++)

{

r = a[i][k];

for (j=0; j<n; j++)

c[i][j] +=r\*b[k][j];

}

**ikj loop**

for (i=0; i<n; i++)

for (k=0; k<n; k++)

{

r=a[i][k];

for (j=0; j<n; j++)

c[i] [j] +=r\*b[k][j];

}

**jki loop**

for (j=0; j<n; j++)

for (k=0; k<n; k++)

{

r=b[k] [j];

for (i=0; i<n; i++)

c[i][j] +=a[i] [k]\*r;

}

**kji loop**

for (k=0; k<n; k++)

for (j=0; j<n; j++)

{

r=b[k] [j];

for (i=0; i<n; i++)

c[i] [j] +=a[i][k]\*r;

}

**Analysis of ijk form**

**Array A**: For a fixed i and j, as k is varied, row-i will be accessed, resulting in n/b cold misses (one miss for every b accesses). As j is varied, the same row will be repeatedly accessed in cache, resulting in hits. The same costs recur for each outer iteration i, as different rows of A are accessed. So the total number of misses is n\*n/b for both direct-mapped and fully associative caches.

**Array b**: For fixed i and j, as k is varied, elements in a column of b are accessed. When j is changed by one, the adjacent column of b is accessed, but will incur misses for a direct mapped cache (hits for a fully associative cache since only n/b lines would be used). However, no temporal reuse is possible for different iterations of the i-loop, even with a fully associative cache since there is insufficient capacity to hold all of b till the outer loop i changes. So misses for a direct-mapped cache will be n\*n\*n, and n\*(n/b)\*n for a fully associative cache.

**Array C**: It will have both temporal and spatial reuse; the only misses will be initial cold misses for both direct and associative caches. So total number of misses will be n\*n/b.

Each entry in the table represents a multiplier with respect to that loop index, on the number of cache misses. For the innermost loop, it is the total number of cache misses for a fixed value of the outer two loops. For the middle loop, it represents how many times the inner-loop miss count will get multiplied as we run through all iterations of the middle loop, for a fixed value of the outer loop.

**Analysis of ikj form**

**Array A:** It will have total temporal and spatial reuse; the only misses will be initial cold misses for both direct and associative caches. So total number of misses = N\*N/B.

**Array B:** It will have complete spatial reuse since it is accessed by row in the innermost loop. But no temporal reuse is possible since there is insufficient capacity to hold all of B till the outer loop i changes. So misses for both direct and associative cache will be N\*N\*N/B.

**Array C:** For a fixed i and k, as j is varied, row-i will be accessed, occupying N/B adjacent blocks in the cache. As k is varied, the same row will be repeatedly accessed in cache. So only the initial compulsory misses will occur, with either direct-mapped or fully associative cache, i.e. total number of misses = N\*N/B.