Q1  
256x256

|  |  |
| --- | --- |
| matmul()  A screenshot of a computer  Description automatically generated | matmul ikj()  A screenshot of a computer  Description automatically generated |
| matmul AT()  A computer screen shot of a number  Description automatically generated | matmul BT() |

In 256 x 256, matmul AT() is the faster in this task

512x512

|  |  |
| --- | --- |
| matmul() | matmul ikj() |
| matmul AT() | matmul BT() |

matmul ikj() is the faster when it comes to 512x512

1024x1024

|  |  |
| --- | --- |
| matmul() | matmul ikj() |
| matmul AT() | matmul BT() |

Conclusion :matmul ikj() is the faster in average run time

Q2

Size 2048x2048

Tiling method:

A screen shot of a computer program

Description automatically generated

|  |  |
| --- | --- |
| matmul() | matmul\_tiling() |
| A screenshot of a computer  Description automatically generated |  |

Unrolling method:

A computer screen with text

Description automatically generated

|  |  |
| --- | --- |
| matmul() | matmul\_unrolling() |
| A screenshot of a computer  Description automatically generated |  |

unrolling method is the best in 2048x2048 cache optimization.

reordering and unrolling:

matmul\_ikj()

A screenshot of a computer

Description automatically generated

matmul\_ikj\_unrolling()

A screenshot of a computer

Description automatically generated

The combination of reordering and unrolling does not perform well in cache optimization. Combining reordering and unrolling increases the complexity of the loop, which might make it harder for the compiler to apply its own optimizations effectively.

Loop unrolling over loop reordering in 2048x2048 as Unrolling can increase instruction-level parallelism which leads to better cache optimization than just reordering ikj method.