OBHPC TD2
Performance Report
November 2022

R. THIENARD

Introduction & Objectives

The objective of this TP is to discover what performance measurement is. In order to do this, we must use 3 basic version functions and use their implementations. From the most basic to optimized versions, we have to implement and to measure dgemm, reduc and dotprod. Each functions will be running on 3 differents compilers, Gcc (gnu compiler), Clang and Icx (Intel compiler), to measure the impact of the compiler on the performances. I will use only 2 flags to show the differences between compilers and in the annex : Flag Comparaison, I show all my test with differents flag on gcc compiler. I used gcc to highlight the flags because ICX is available for Intel users and will be faster with the cblas implementation.

Unit test

During my implementation of these functions, I decided, to maximize the process, to use unit test. In fact, unit tests allowed me to check quickly and easily the operation of each fonction. Unit test verify for a small sample if the output corresponds to what I expect. I give small sample and I manually calculate the output so that there is no risk of error.

Performance capturing

To measure all functions, I have configured my CPU's to be as stable as possible. In order to do this, I fixed their frequency while making sure that the execution would be on the same core. I used cpupower and taskset. Cpupower allows user to set the cpu frequency if it's possible or just choose the gouvernor. The gouvernor has several possibility like "powersave", "performance" or "userspace". For some CPU, the "usrspace" is not reachable and only "powersave" and "performance" could be chosen. this is my case on my laptop, but on my server, I can fix manualy my frequency with the "userspace". You must disable the boost on your CPU.
Now, to run the program, I only have to fix it on 1 core of my cpu using taskset.

CPUs used

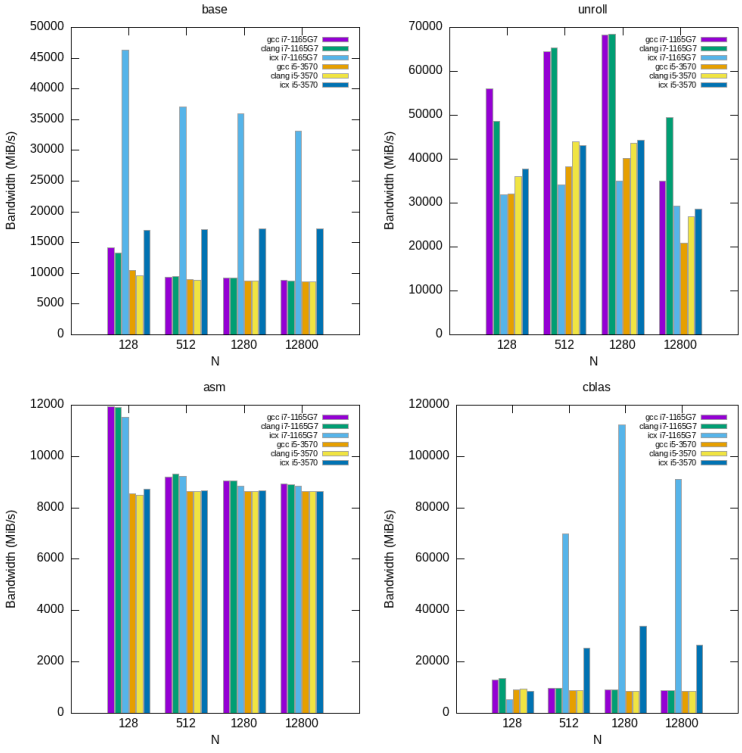
To realize this performance report, I used 2 processors. Both commercialized by Intel. The first one is a I7-1165G7 and the second a I5-3570. It's a Tigerlake's processor released in 2020. It is on my laptop and has 4 cores with 8 threads. It has a maximum frequency of 4.7GHz. It has also 12MiB of L3 cache and has AVX512. The cons of this processor is that I can not be configured manualy its frequency because of P-STATE.
The second cpu is a I5-3570. It is an older version belonging to the Ivy Brige generation. Released in 2012, this CPU has 4 cores and 8 threads too. Its maximum frequency is 3.8GHz and it has 6MiB of L3 cache. It hasn't AVX512 but on this CPU I can manage easily the frequency to generate more stable measure.
All the information concerning the characteristics are available in the files I7-1165G7.txt and I5-3570.txt.

Dotproduct

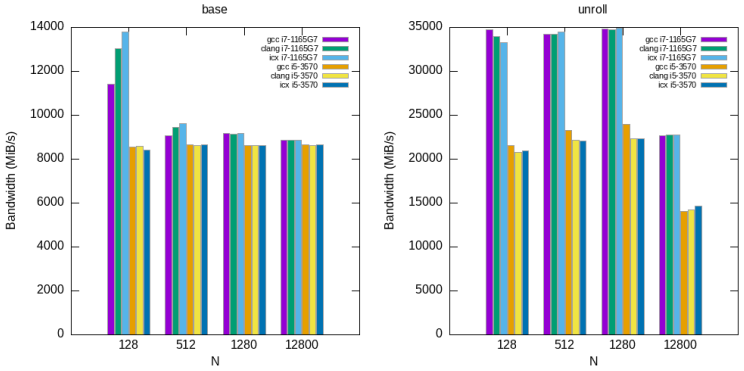
The Dotproduct is a function to sum the product of two vectors. I have set up 4 versions of the dotproduct. Each of the performance measures was done respecting the part Performance capturing. The measured versions are :

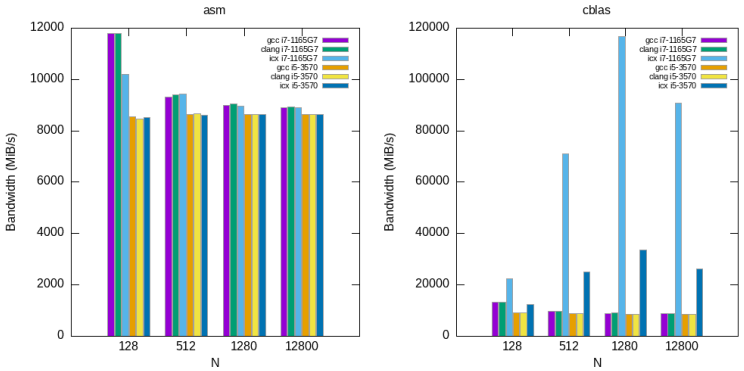
- The basic version written in C.
- The version with an unroll 8, also written in C.
- The x86 assembler version.
- And finally the one provided by CBLAS.

The compilation flags are "-O1" and "-O3".

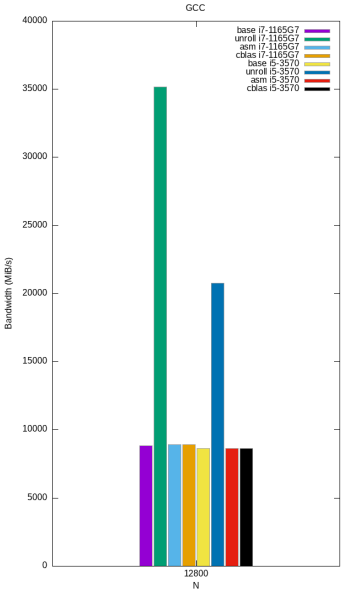


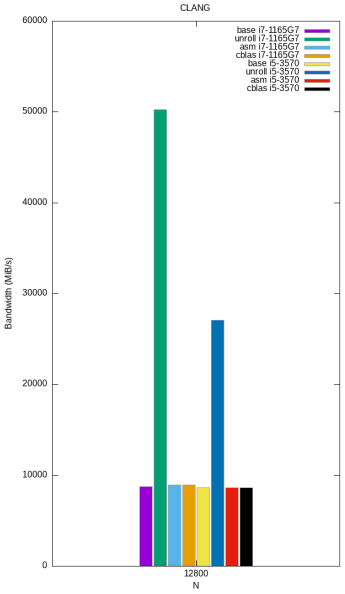
O3 flag

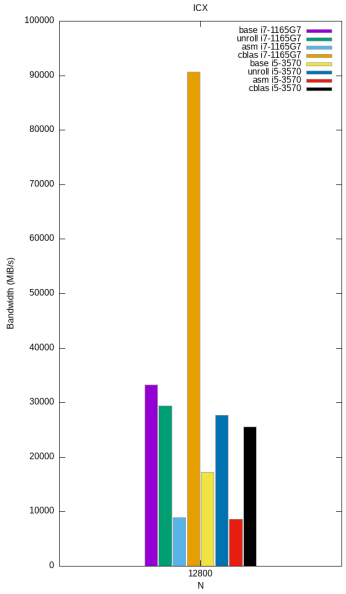




O1 flag







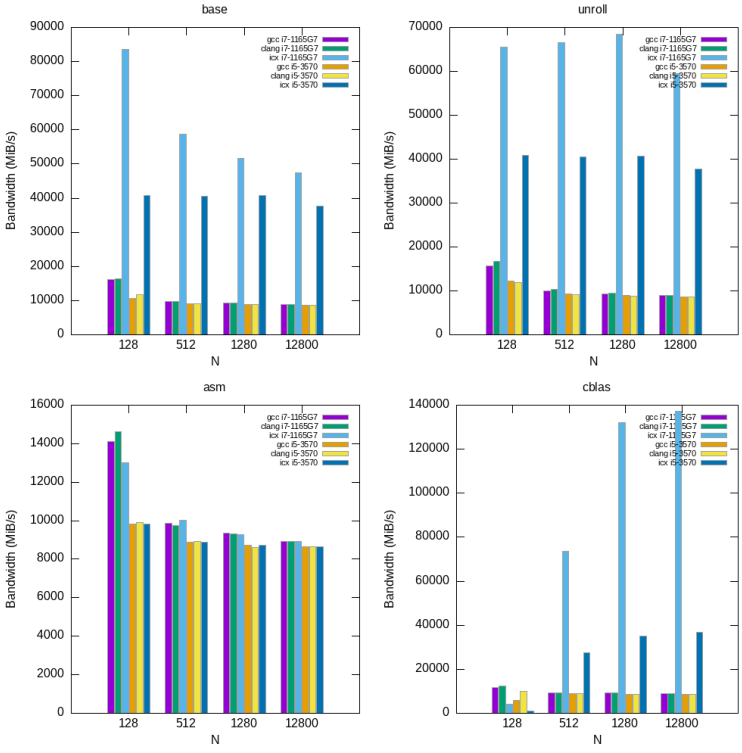
O3 flag

Reduction

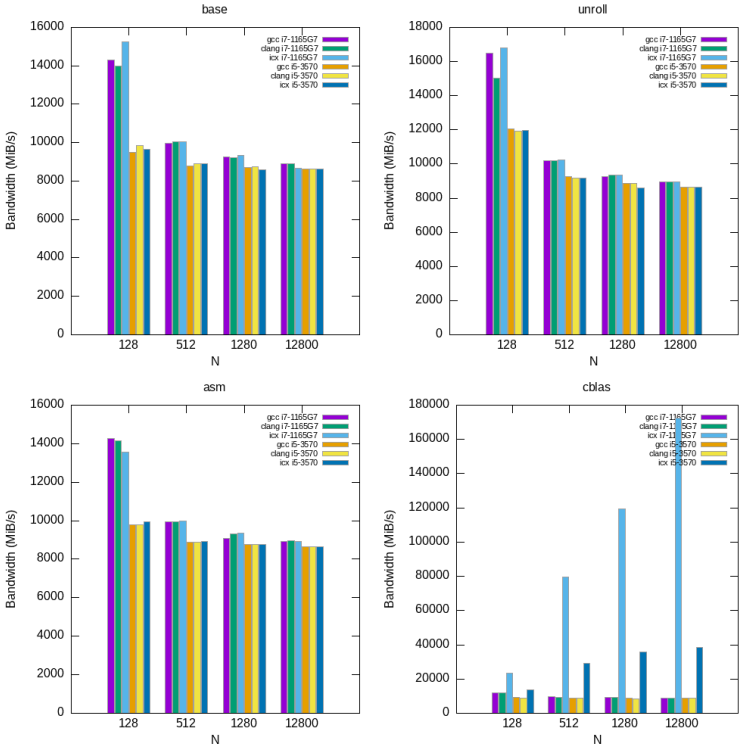
The Reduction is a function to sum one vector. I have set up 4 versions of the reduction. Each of the performance measures was done respecting the part Performance capturing. The measured versions are :

- The basic version written in C.
- The version with an unroll 8, also written in C.
- The x86 assembler version.
- And finally the one provided by CBLAS.

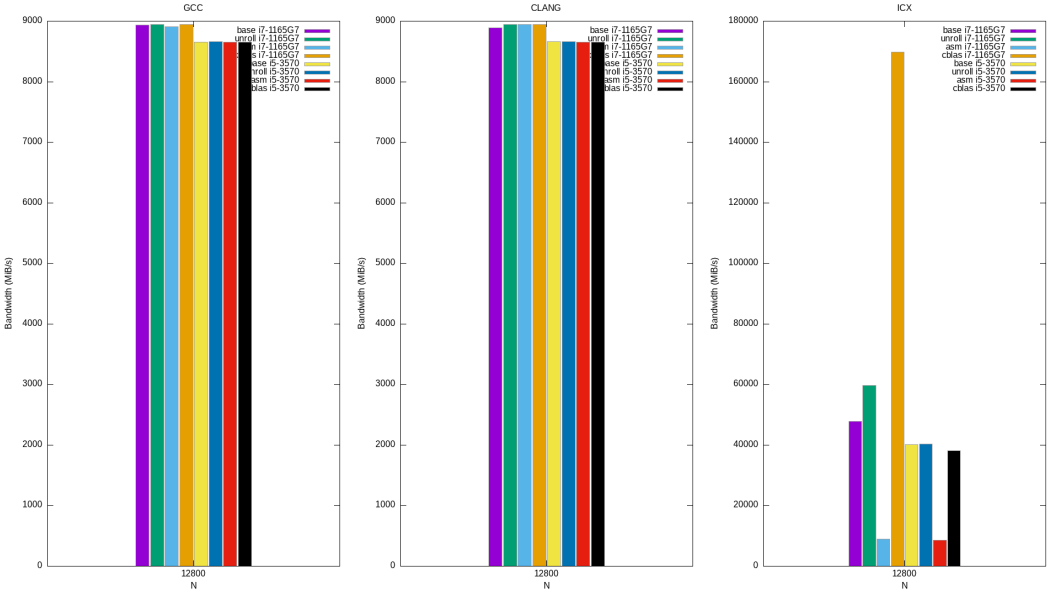
The compilation flags are "-O1" and "-O3".



O3 flag



O1 flag



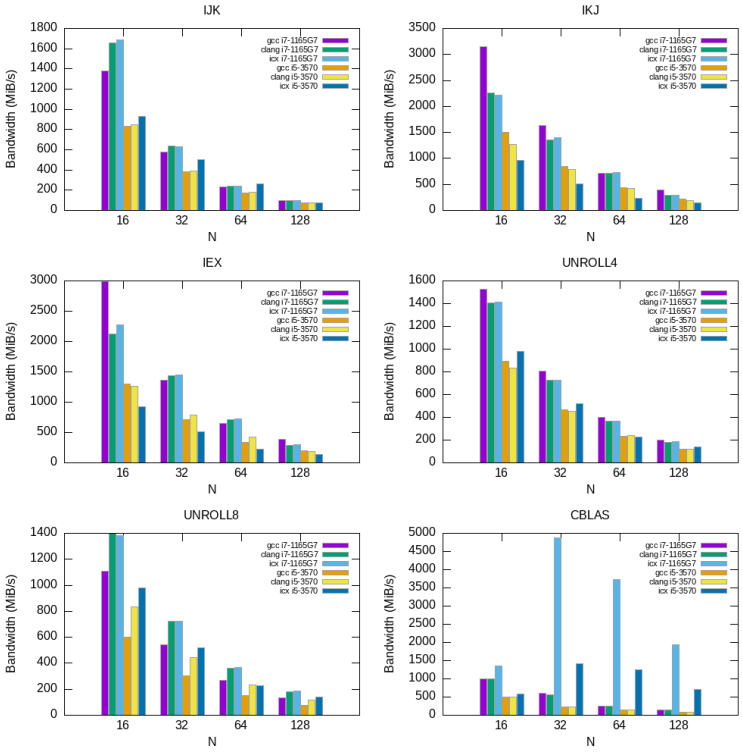
O3 flag

Dgemm

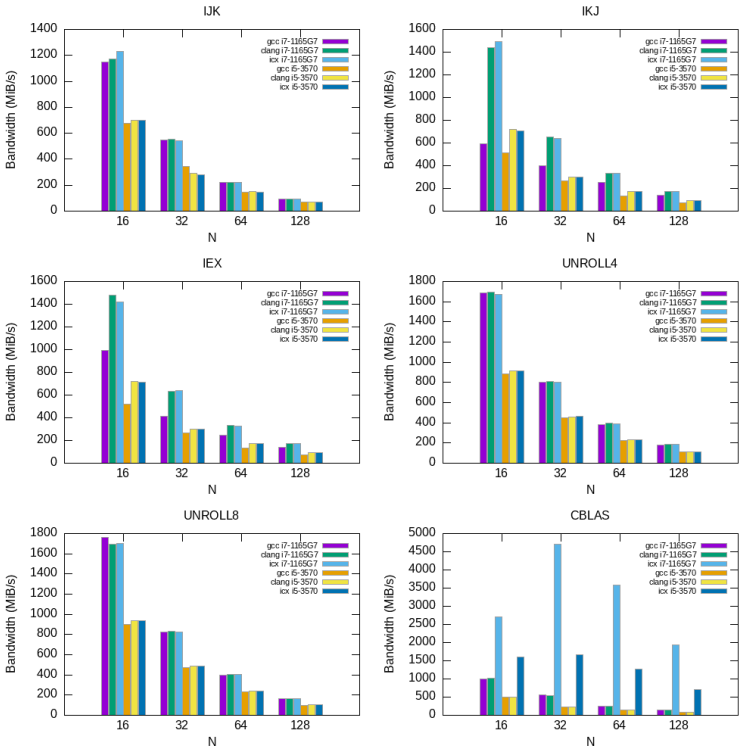
The Dgemm is a function for multiplying two matrix. I have set up 6 versions of the dgemm. Each of the performance measures was done respecting the part Performance capturing. The measured versions are :

- The basic version written in C which is also call IJK.
- The first optimized version written in C which is also call IKJ. This optimization requires only to move 2 loops of the basic version.
- The second optimized version written in C which is also call IEX.
- The version with an unroll 4, also written in C. Like the unroll 8 but smaller.
- The version with an unroll 8, also written in C.
- And finally the one provided by CBLAS.

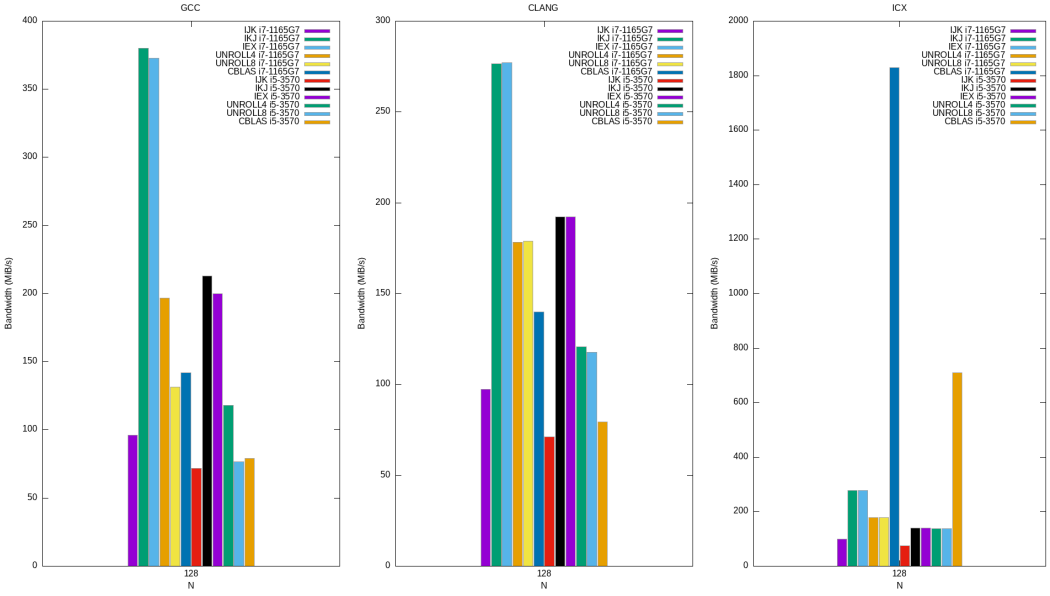
The compilation flags are "-O1" and "-O3".



O3 flag



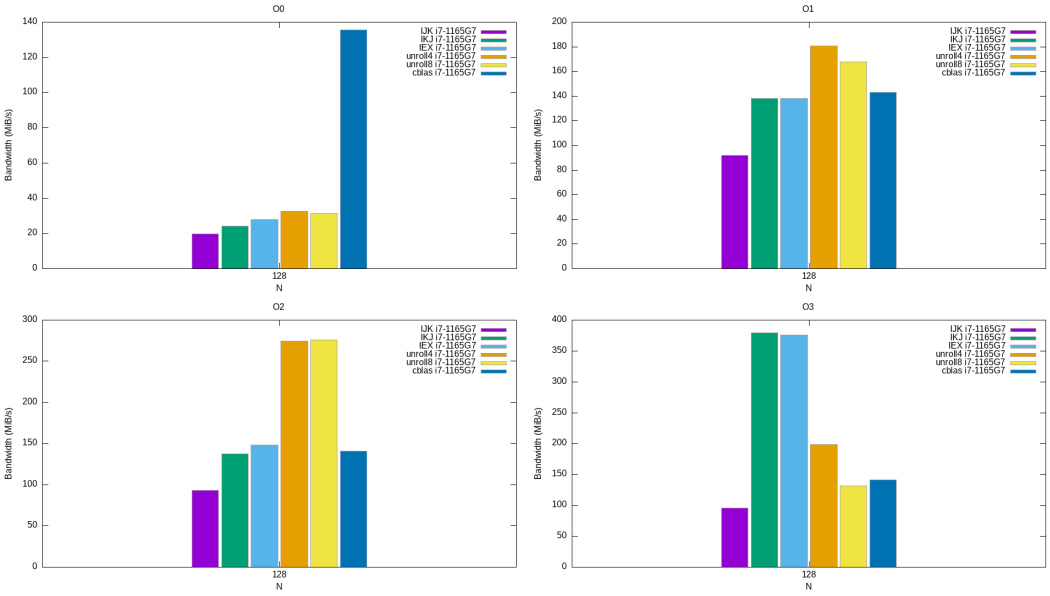
O1 flag



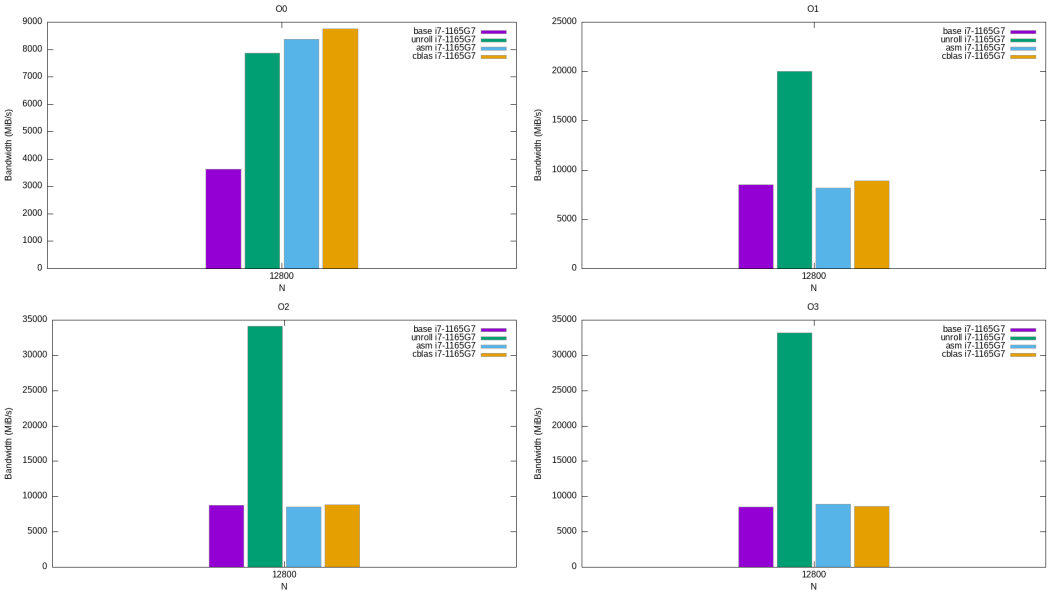
O3 flag

Flag Comparaison

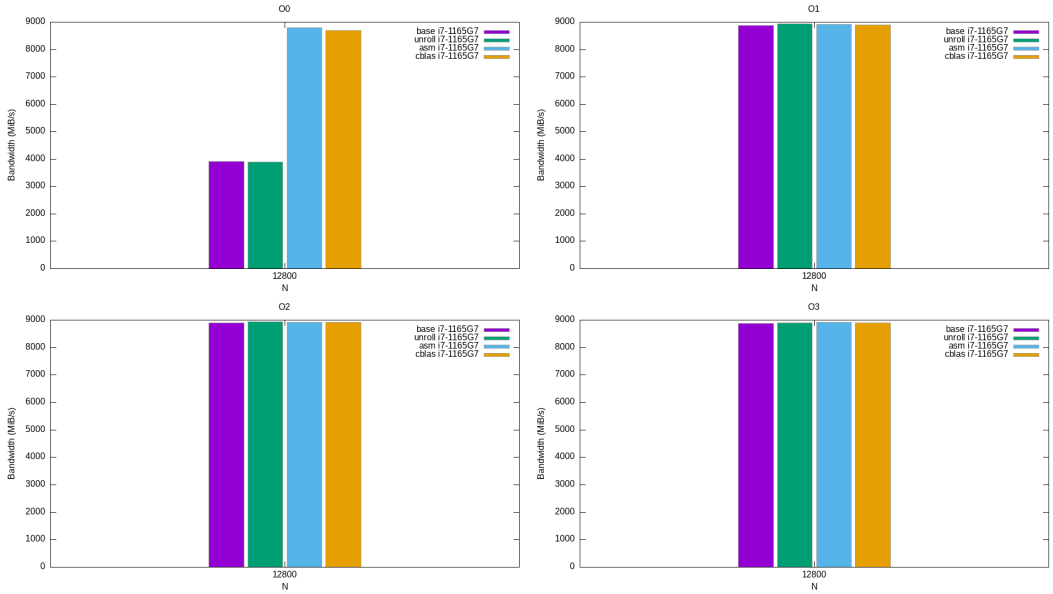
Dgemm



Dotprod



Reduc


O0
O1
O2
O3
base i7-1165G7
unroll i7-1165G7
asm i7-1165G7
cblas i7-1165G7
Bandwidth (MB/s)
N
12800
0
1000
2000
3000
4000
5000
6000
7000
8000
9000