## Introducción

Después de lo aprendido en la práctica anterior tenemos que ver dos nuevas funciones de envío y recepción de mensajes las cuales son "*MPI_Scatter*" y "*MPI_Reduce*", las cuales con la primera nos permite enviar una igual cantidad de datos para cada procesador, lo que difiere con el broadcast es que no se envía una copia sino los datos se dividen en partes iguales para cada uno; pasando a la recepción tenemos que el Reduce aparte de recolectar los datos de los procesadores a la raíz, permite hacer una operación con los datos que se reciben.

## Preguntas

- Realizar un programa que haga el producto punto de dos vectores utilizando las funciones descritas en el manual.
- Tomar las medidas de tiempo.
- Adjunta tu código y la impresión de pantalla de lo que ocurre.

## Respuestas

```
ramy@Ramy:~/Distribuidos/pr4$ mpicc pr4.c -o pr4
ramy@Ramy:~/Distribuidos/pr4$ mpirun -np 150 ./pr4
--------------------------------------------------------------------------
WARNING: Linux kernel CMA support was requested via the
btl_vader_single_copy_mechanism MCA variable, but CMA support is
not available due to restrictive ptrace settings.

The vader shared memory BTL will fall back on another single-copy
mechanism if one is available. This may result in lower performance.

  Local host: Ramy
--------------------------------------------------------------------------
El resultado es: 2988
El resultado del envío de mensajes es de: 0.000171[s]
El tiempo de la suma es de: 0.010548[s]
El tiempo de la multiplicación es de: 0.000002[s]
El tiempo entre la multiplicación y la suma es de: 0.010550[s]
El tiempo total fue de: 0.010721[s]
[Ramy:07944] 149 more processes have sent help message help-btl-vader.txt / cma-permission-denied
[Ramy:07944] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error messages
```

```
ramy@Ramy:~/Distribuidos/pr4$ mpirun -np 10 ./pr4
--------------------------------------------------------------------------
WARNING: Linux kernel CMA support was requested via the
btl_vader_single_copy_mechanism MCA variable, but CMA support is
not available due to restrictive ptrace settings.

The vader shared memory BTL will fall back on another single-copy
mechanism if one is available. This may result in lower performance.

  Local host: Ramy
--------------------------------------------------------------------------
El resultado es: 217
El resultado del envío de mensajes es de: 0.000190[s]
El tiempo de la suma es de: 0.000223[s]
El tiempo de la multiplicación es de: 0.000001[s]
El tiempo entre la multiplicación y la suma es de: 0.000225[s]
El tiempo total fue de: 0.000414[s]
[Ramy:08504] 9 more processes have sent help message help-btl-vader.txt / cma-permission-denied
[Ramy:08504] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error messages
```

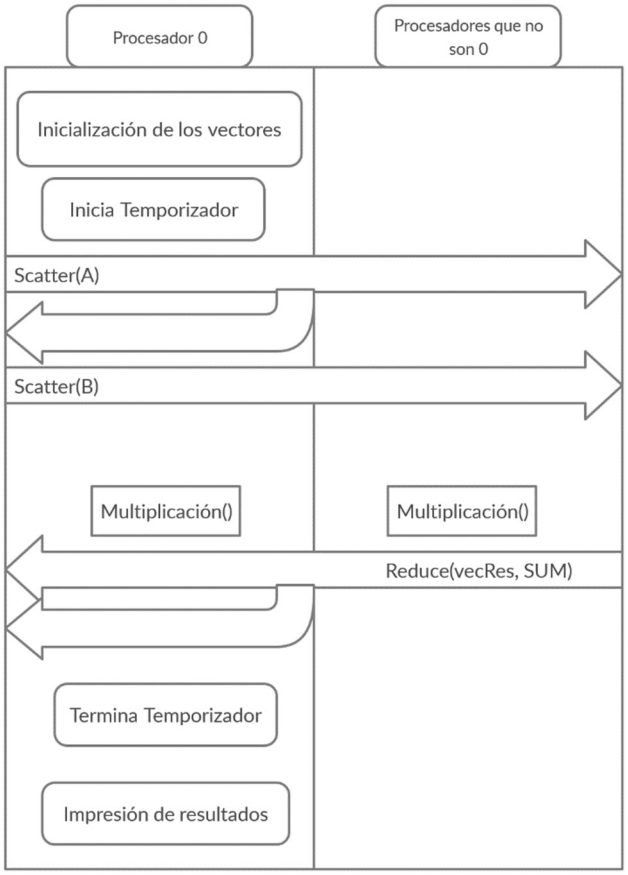
En este programa tenemos que se hace el producto punto de dos vectores de tamaño n, de donde n es el número de procesadores que nosotros utilicemos, en los ejemplos mostrados con las capturas tenemos que los vectores son de 150 y 10, en ellos podemos ver los diferentes tiempos que queremos

observar, como lo es el paso del mensaje a los procesadores es similar en ambos tamaños, mientras que en la recolección del mensaje junto a la suma de los productos incrementa conforme el tamaño de los vectores aumenta, mientras que el tiempo de la multiplicación que se realiza en cada procesador es constante.

```
ramy@Ramy:~/Distribuidos/pr4$ ./pp
Dame el tamaño de los vectores: 10000
El resultado es: 204150
El tiempo total fue de: 0.000064[s]
```

Con respecto a uno hecho de manera secuencial, podemos comparar los tiempos, del cual vemos que de manera secuencial se realiza de manera más rápida, pero esto se debe a los mensajes que se envían, eso es lo que ocasiona que utilizando mpi a una menor cantidad de procesadores el tiempo empleado en el envío y recepción es demasiado, el impacto real lo veríamos cuando se utilicen mayores procesadores. El código se anexa al final.

**Diagrama UML**




**Conclusión**

En conclusión tenemos que es más sencillo y facilita de mejor manera el envío de mensajes por medio de la función Scatter ya que nos permite dividir los datos a trabajar de manera automática, y junto a ello la función Reduce para este tipo de problemas que teníamos que resolver fue demasiado útil, ya que aparte de permitirnos recibir lo que envían los otros procesadores, esos valores nos los suma por lo que de cierta medida nos ahorra tiempo al intentarlo hacer por separado.