# Ejercicio 5 de teoría: transformación vectorial iterativa:


# UNIVERSIDAD DE GRANADA

Luis González Romero, XXXXXXXXX, luisgonromero@correo.ugr.es

Escuela Técnica Superior de Ingeniería informática y Telecomunicaciones

29 de junio de 2021

# 1. Transformación vectorial iterativa

```
// calculo mpi
MPI_Status status;
int tam_bloq = (n+2)/size;
float * Bloque = new float[tam_bloq];
MPI_Scatter(V, tam_bloq, MPI_FLOAT, Bloque, tam_bloq, MPI_FLOAT, 0, MPI_COMM_WORLD);

float izq, der, tmp;
int dest_im1 = mod(rank-1, size);
int dest_ip1 = mod(rank+1, size);
for(int k=0;k<100;k++)
{
  //cout << "Envio " << Bloque[0] << " a " << dest_im1 << " desde " << rank << endl;
  MPI_Send(&Bloque[0], 1, MPI_FLOAT, dest_im1, 0, MPI_COMM_WORLD);
  //cout << "Recibiendo de " << dest_ip1 << " desde " << rank << endl;
  MPI_Recv(&der, 1, MPI_FLOAT, dest_ip1, 0, MPI_COMM_WORLD, &status);
  //cout << "Recibido de " << dest_ip1 << " desde " << rank << endl;

  //cout << "Envio " << Bloque[tam_bloq-1] << " a " << dest_ip1 << " desde " << rank << endl;
  MPI_Send(&Bloque[tam_bloq-1], 1, MPI_FLOAT, dest_ip1, 1, MPI_COMM_WORLD);
  //cout << "Recibiendo de " << dest_im1 << " desde " << rank << endl;
  MPI_Recv(&izq, 1, MPI_FLOAT, dest_im1, 1, MPI_COMM_WORLD, &status);
  //cout << "Recibido de " << dest_im1 << " desde " << rank << endl;

  int j = ( rank == 0 ) ? 1 : 0;
  for(j; j <= tam_bloq-2; ++j)
  {
    tmp = Bloque[j];
    Bloque[j] = (izq - Bloque[j] + Bloque[j+1])/2;
    izq = tmp;
  }
  if (rank!=size-1)
    Bloque[tam_bloq-1] = (izq - Bloque[tam_bloq-1] + der)/2;
}// fin calculo mpi

//for(int i=0; i<tam_bloq;++i)
  //cout << "desde " << rank << " con i=" << i << " valor: Bloque=" << Bloque[i] << endl;

MPI_Gather(Bloque, tam_bloq, MPI_FLOAT, V_final, tam_bloq, MPI_FLOAT, 0, MPI_COMM_WORLD);
```

Cada proceso tiene un tamaño de bloque igual a una parte igual de $n$ más los dos extremos. Se envía casi de forma idéntica que en el pseudocódigo mostrado en las transparencias y se calcula igual excepto para los extremos: rank(0) no ejecuta la iteración 0 y rank(size-1) no ejecuta la ultima.

Finalmente se hace el gather de todos los bloques. A continuación se muestra una ejecución para n=12(10+2), en la que se pasa el mismo test que en las prácticas y se muestran todos los elementos de los vectores.

```
roronoasins in ~/Documents/ppr/practicas/teoria5 λ mpirun -np 4 p1 10
Invalid MIT-MAGIC-COOKIE-1 keyPASSED TEST !!!
i: 0 V-secuencial: 0
i: 1 V-secuencial: -9.812e+14
i: 2 V-secuencial: 1.88307e+15
i: 3 V-secuencial: -2.63267e+15
i: 4 V-secuencial: 3.16928e+15
i: 5 V-secuencial: -3.44937e+15
i: 6 V-secuencial: 3.4501e+15
i: 7 V-secuencial: -3.17123e+15
i: 8 V-secuencial: 2.63522e+15
i: 9 V-secuencial: -1.88542e+15
i: 10 V-secuencial: 9.82595e+14
i: 11 V-secuencial: 0
i: 0 V-final: 0
i: 1 V-final: -9.812e+14
i: 2 V-final: 1.88307e+15
i: 3 V-final: -2.63267e+15
i: 4 V-final: 3.16928e+15
i: 5 V-final: -3.44937e+15
i: 6 V-final: 3.4501e+15
i: 7 V-final: -3.17123e+15
i: 8 V-final: 2.63522e+15
i: 9 V-final: -1.88542e+15
i: 10 V-final: 9.82595e+14
i: 11 V-final: 0
```