Algoritmos Paralelos

(clase 22.09.15)

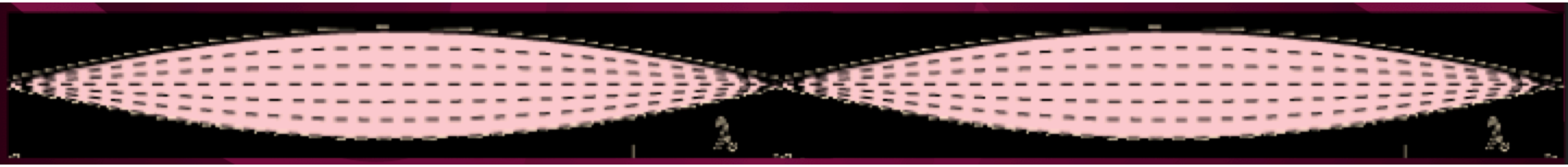
Prof. J.Fiestas

Ecuacion hiperbólica de onda:

La ecuación de onda es una ecuación diferencial de segundo orden, en una dimensión, que describe el movimiento de vibración de una cadena de tipo transversal o longitudinal.

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2}$$

u(x,t) representa la desviación al tiempo t de un punto de la cadena en la posición x en reposo

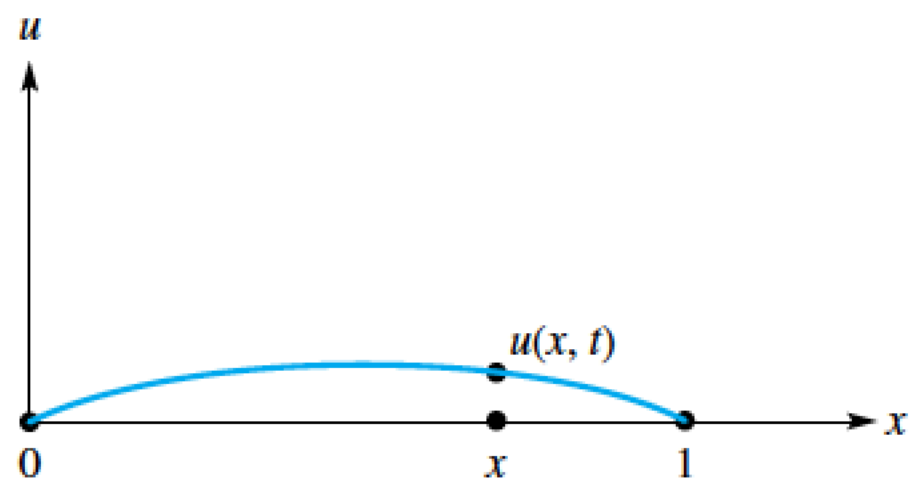


Ecuacion hiperbólica de onda:

Supongamos los puntos en la cadena tienen coordenadas en x entre 0<=x<=1, y que al tiempo t=0, la desviación satisface la ecuación u(x,0)=f(x) y u(x,0)=0
Asimismo, los extremos de la cadena son fijos.

Es decir:

$$\begin{cases} u_{tt} - u_{xx} = 0 \\ u(x,0) = f(x) \\ u_t(x,0) = 0 \\ u(0,t) = u(1,t) = 0 \end{cases}$$



Ecuacion hiperbólica de onda:

Para encontrar una **solución analítica**, podemos postular la solución

$$u(x, t) = \frac{1}{2}[f(x+t) + f(x-t)]$$

siempre que f tenga dos derivadas, y se cumpla

$$f(-x) = -f(x) \qquad f(x+2) = f(x)$$

$$u(x, 0) = f(x) \qquad u_t(x, 0) = \frac{1}{2}[f'(x) - f'(x)] = 0$$

$$u(0, t) = \frac{1}{2}[f(t) + f(-t)] = 0$$

$$u(1, t) = \frac{1}{2}[f(1+t) - f(t-1+2)] = 0$$

Ecuacion hiperbólica de onda:

Para encontrar una **solución numérica**, con intervalos h para x e intervalos k para t

$$\frac{1}{h^2}[u(x+h,t) - 2u(x,t) + u(x-h,t)]$$
$$= \frac{1}{k^2}[u(x,t+k) - 2u(x,t) + u(x,t-k)]$$

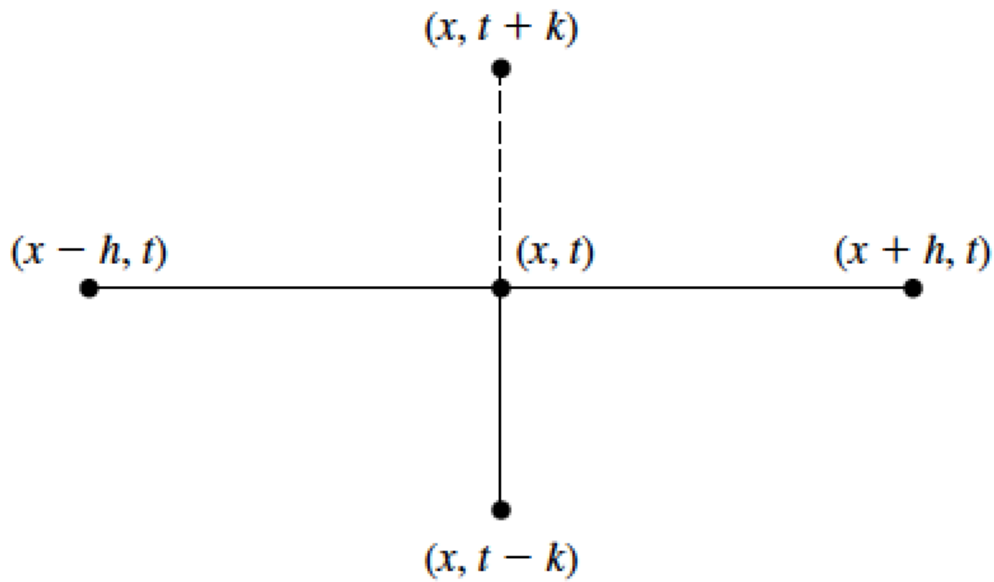
o simplificando:

$$\boxed{u(x,t+k) = \rho u(x+h,t) + 2(1-\rho)u(x,t) + \rho u(x-h,t) - u(x,t-k)}$$

con $\rho = \frac{k^2}{h^2}$

Ecuacion hiperbólica de onda:

Para encontrar una solución numérica, con intervalos h para x e intervalos k para t



Con condiciones de frontera

$$\begin{cases} u(x,0) = f(x) \\ \frac{1}{k}[u(x,k) - u(x,0)] = 0 \\ u(0,t) = u(1,t) = 0 \end{cases}$$

Ecuacion hiperbólica de onda:

El problema será resuelto inicialmente en t=0, donde u(x,0)=f(x), y consecutivamente en t=k, t=2k, t=3k, ...

Note que $u(x,k) = u(x,0) = f(x)$

Para t=0, será

$$u(x,k) = \rho u(x+h,0) + 2(1-\rho)u(x,0) + \rho u(x-h,0) - u(x,-k)$$

Y finalmente: $u(x,k) = \frac{1}{2}\rho[f(x+h) + f(x-h)] + (1-\rho)f(x)$

Lo que permite calcular $u(x,nk), n \geqq 2$ usando:

$$\boxed{u(x,t+k) = \rho u(x+h,t) + 2(1-\rho)u(x,t) + \rho u(x-h,t) - u(x,t-k)}$$

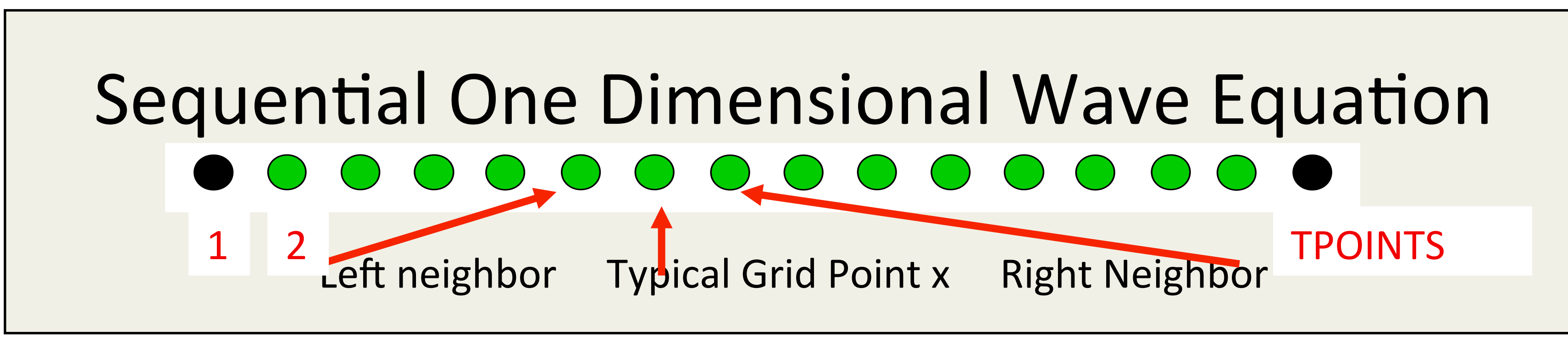
Ecuacion hiperbólica de onda:

Discretizando:

$$(u(x,t+1)-2u(x,t)+u(x,t-1))/\delta t^2 = -c^2\,(u(x+1,t)-2u(x,t)+u(x-1,t))/\delta x^2$$

O tambien, para calcular u en un paso en el futuro, basado en x actual y un paso en el pasado

$$u(x,t+1) = 2u(x,t) - u(x,t-1) - (u(x+1,t)-2u(x,t)+u(x-1,t))\,(c^2\,\delta t^2/\delta x^2)$$



Ecuacion hiperbólica de onda:

Código en serie: cálculo de tiempo t=0

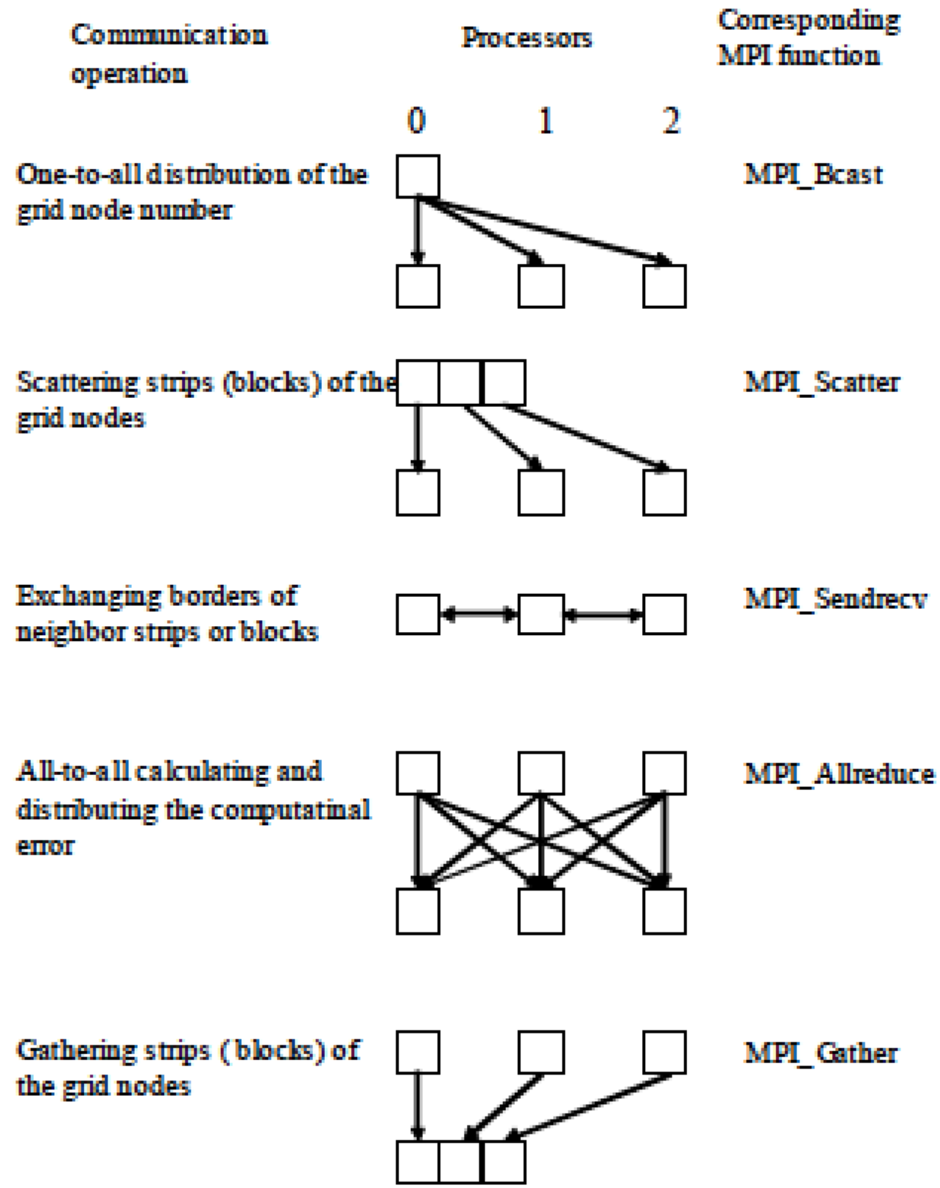
```
for (i = 1; i <= n - 1; i++)
{
   x = i * h;
   W[i] = f(x);
   V[i] = 0.5 * (rho * ( f(x-h) + f(x+h) ) + 2 * (1 - rho) * f(x) );

    fileout<<x<<" "<<V[i]<<endl;
}
```

Ecuacion hiperbólica de onda:

Código en serie: cálculo de tiempo t>0

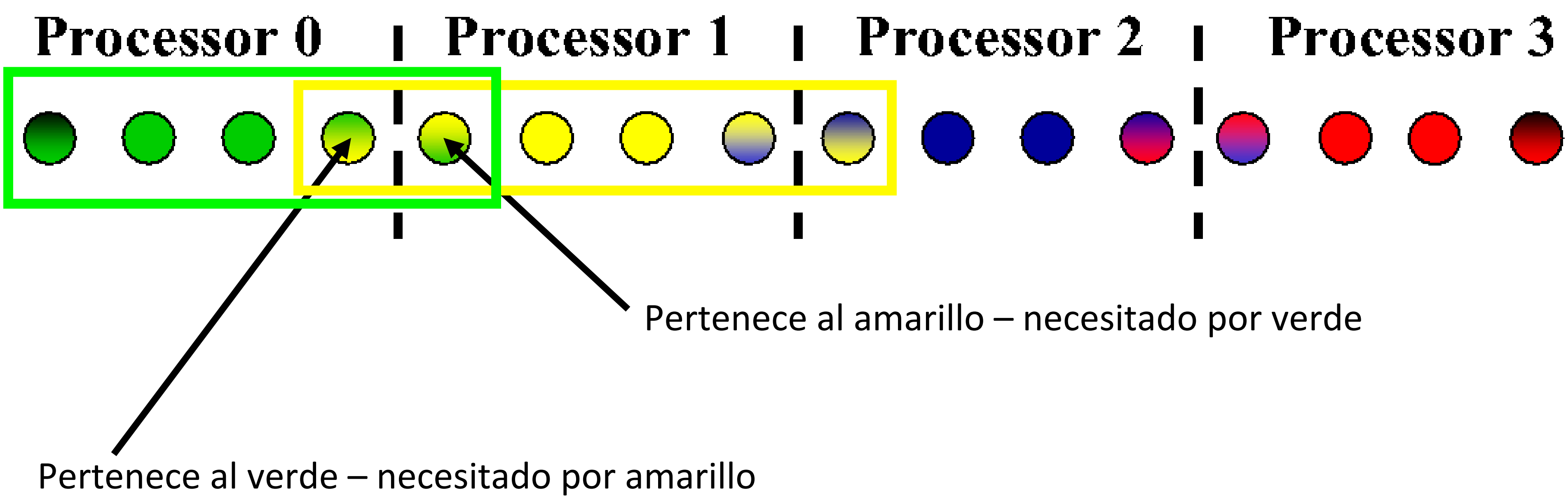
```
for (j = 2; j <= m; j++)
{
  sprintf(outname,"%04d.dat",j);
  fileout.open(outname);
  for (i = 1; i <= n - 1; i++)
  {
    x = i * h;
    U[i] = rho * (V[i+1] + V[i-1]) + 2 * (1 - rho) * V[i] - W[i];
    fileout<<x<<" "<<U[i]<<endl;
  }
  fileout.close();

  for (i = 1; i <= n - 1; i++)
  {
    W[i] = V[i];
    V[i] = U[i];
  }

}
```

Comunicación de data para la solución de ecuaciones diferenciales



Ecuacion hiperbólica de onda:

Esquema de algoritmo en paralelo



Necesita comunicación P2P para intercambiar puntos de frontera con vecinos

```
..........
for (j = 2; i <= m; j++) {
    /* Intercambio de data con vecino de la izquierda*/
    if (first != 1) {
      MPI_Send(&V[1], 1, MPI_DOUBLE, left, RtoL, MPI_COMM_WORLD);
      MPI_Recv(&V[0], 1, MPI_DOUBLE, left, LtoR, MPI_COMM_WORLD,
            &status);
      }
    /* Intercambio de data con vecino de la derecha*/
    if (first + n -1 != TPOINTS) {
      MPI_Send(&V[n], 1, MPI_DOUBLE, right, LtoR, MPI_COMM_WORLD);
      MPI_Recv(&V[n+1], 1, MPI_DOUBLE, right, RtoL,
            MPI_COMM_WORLD, &status);
      }
............
    }
```

```
npts = n/numtasks;
for (i = 0, k = 0; i < numtasks; i+
+) {
if (taskid == i) {
        first = k + 1;
    for (j = 1; j <= npts; j++, k++) {
    x=..
    W[j]=...
    V[j] = ...
         }
       }
     else k += npts;
     }
}
```

```
if (taskid == numtasks-1)
   right = 0;
else
   right = taskid + 1;

if (taskid == 0)
   left = numtasks - 1;
else
   left = taskid - 1;
```

Ejercicio 12:

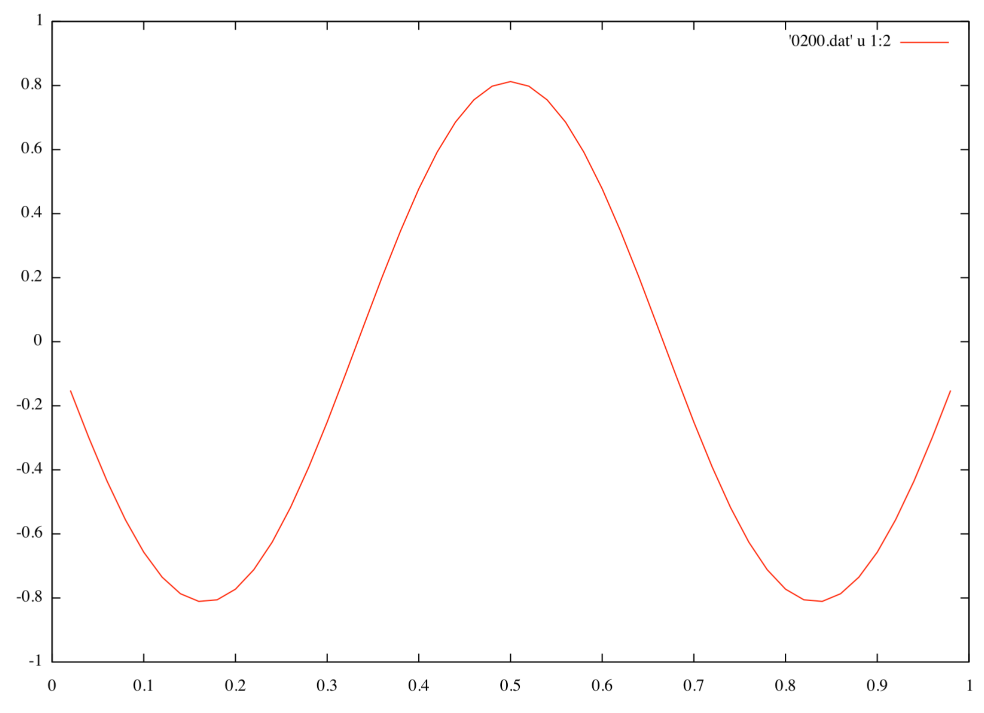
1. Compilar y ejecutar el código en serie de vibración de onda en una dimensión.

Utilizar n=50 puntos, h=1/n, k=0.002, m=500.

La función inicial es

$f(x)=\sin(3\pi x)$.

Condiciones de frontera son u(0)= u(n)=0



Ejercicio 12:

2. Paralelizar el código utilizando el algoritmo discutido en clase

Para ello:

- Repartir el dominio en partes iguales entre los procesos, i.e. n/numtasks
- Utlizar comunicación P2P (MPI_Send, MPI_Recv) para intercambiar valores de frontera con vecinos, y actualizar los puntos de cada proceso
- Enviar valores de procesos al nodo maestro, guardarlos en un vector, imprimirlos y graficar la onda de una dimension