Qüestió 1 (1.1 punts)
Donada la següent funció:

```
void f(double A[N][N], double B[N][N], double v[N]){
  double s;
  int i, j, k;
  for(k=0; k<N; k++){
    s=0.0;
    for(i=0; i<N; i++)
      for(j=0; j<N; j++)
        s+=A[i][j]*B[i][j];
    v[k]=v[k]/s;
    for(i=0; i<N; i++)
      for(j=0; j<i; j++)
        A[i][j]+=B[i][j]/s;
  }
}
```

0.2 p. (a) Paral · lelitza, si és possible, el bucle més extern del codi anterior, justificant la resposta.

Solució: No és possible paral · lelitzar el bucle exterior, perquè hi ha dependència entre les iteracions del bucle k, donat que cada iteració del bucle modifica la matriu A.

0.6 p. (b) Paral · lelitza de la forma més eficient possible els dos bucle interns (bucles amb la variable `i`), usant una sola regió paral · lela.

Solució:

```
void f(double A[N][N], double B[N][N], double v[N]){
  double s;
  int i, j, k;
  for(k=0; k<N; k++){
    s=0.0;
    #pragma omp parallel
    {
      #pragma omp for private(j) reduction(+:s)
      for(i=0; i<N; i++)
       for(j=0; j<N; j++)
          s+=A[i][j]*B[i][j];
      #pragma omp single nowait
      v[k]=v[k]/s;
      #pragma omp for private(j)
      for(i=0; i<N; i++)
        for(j=0; j<i; j++)
          A[i][j]+=B[i][j]/s;
    }
```

```
    }
  }
```

0.3 p. (c) Calcula el temps seqüencial, el temps paral · lel, el speedup i l'eficiència corresponent a una sola iteració del bucle k, suposant que únicament s'ha paral · lelitzat el primer bucle intern (primer bucle `i`).

Solució:

$$t(N) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} 2 + 1 + \sum_{i=0}^{N-1}\sum_{j=0}^{i-1} 2 \approx \sum_{i=0}^{N-1} 2N + 2\sum_{i=0}^{N-1} i \approx 2N^2 + 2\frac{N^2}{2} = 3N^2 \text{flops.}$$

$$t(N,p) = \sum_{i=0}^{\frac{N}{p}-1}\sum_{j=0}^{N-1} 2 + \sum_{i=0}^{N-1}\sum_{j=0}^{i-1} 2 \approx \sum_{i=0}^{\frac{N}{p}-1} 2N + N^2 = \frac{2N^2}{p} + N^2 = \left(\frac{2}{p}+1\right)N^2 \text{flops.}$$

$$S(N,p) = \frac{t(N)}{t(N,p)} = \frac{3N^2}{\left(\frac{2}{p}+1\right)N^2} = \frac{3}{\frac{2}{p}+1}.$$

$$E(N,p) = \frac{S(N,p)}{p} = \frac{3}{2+p}.$$

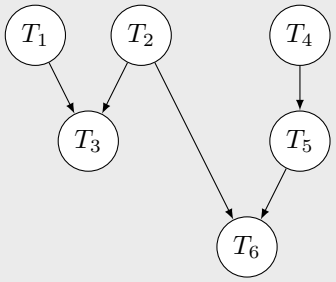
Qüestió 2 (1.2 punts)

Donada la següent funció, on cadascuna de les funcions invocades modifica només el vector que rep com a primer argument:

```
void func(double a[], double b[], double c[], double d[], int n) {
  double x;
  tasca1(b,a,n);      /* Cost: 4n flops */
  x=tasca2(c,a,n);    /* Cost: 3n flops */
  tasca3(b,c,n);      /* Cost: 2n flops */
  tasca4(d,a,n);      /* Cost:  n flops */
  tasca5(d,a,n);      /* Cost:  n flops */
  tasca6(d,x,n);      /* Cost: 3n flops */
}
```

0.4 p. (a) Dibuixa el graf de dependències de dades entre les tasques. Assenyala el camí crític indicant la seua longitud i obtín el grau mitjà de concurrència.

Solució:



Camí crític: T1→T3 o bé T2→T6.

Longitud del camí crític: $L = 4n + 2n = 6n$ flops

Grau mitjà de concurrència:

$$M = \frac{4n + 3n + 2n + n + n + 3n}{6n} = \frac{14n}{6n} = 2.33$$

0.6 p. (b) Fes una versió paral · lela usant OpenMP. S'ha de minimitzar el temps d'execució.

Solució:

```
void func(double a[], double b[], double c[], double d[], int n) {
  double x;
  #pragma omp parallel
  {
    #pragma omp sections
    {
      #pragma omp section
      tasca1(b,a,n);
      #pragma omp section
      x=tasca2(c,a,n);
      #pragma omp section
      {
        tasca4(d,a,n);
        tasca5(d,a,n);
      }
    }
    #pragma omp sections
    {
      #pragma omp section
      tasca3(b,c,n);
      #pragma omp section
      tasca6(d,x,n);
    }
  }
}
```

0.2 p. (c) Obtín el speedup de la versió paral · lela de l'apartat anterior si s'executa amb 3 processadors.

Solució: Temps d'execució seqüencial:

$$t(n) = 4n + 3n + 2n + n + n + 3n = 14n \text{ flops}$$

El temps d'execució paral · lel seria $4n$ flops per al primer `sections`, més $3n$ per al segon `sections`:

$$t(n,p) = 4n + 3n = 7n \text{ flops}$$

Speedup:

$$S(n,p) = \frac{14n}{7n} = 2$$

Qüestió 3 (1.2 punts)

La funció `pluviometria` calcula diferents valors estadístics relacionats amb la precipitació d'agua ocorreguda durant un conjunto de ND dies en un territori per a un total de NM mesos. A la seua vegada, la funció `mes_dia` proporciona l'identificador del mes corresponent a un dia en concret. Paralel · litza la funció amb OpenMP emprant una única regió paral · lela. L'ordre dels dies en el vector `dies_risc` no és rellevant en la versió paral · lela.

```
void pluviometria(float precipitacio[ND], int litres_risc, float pluja_mitja_mes[NM]) {
   int i, mes, ndies_pluja=0, dia_max, ndies_pluja_mes[NM];
   int ndies_risc=0, dies_risc[ND];
```

```
   float suma_pluja=0, precipitacio_max=0, pluja_mitja;
   float suma_pluja_mes[NM];

   /* Inicialització dels elements dels vectors a 0 */
   ...

   for (i=0;i<ND;i++) {
     if (precipitacio[i]>0) {
        suma_pluja+=precipitacio[i];
        ndies_pluja++;
        if (precipitacio[i]>litres_risc) {
           dies_risc[ndies_risc]=i+1;
           ndies_risc++;
        }
        if (precipitacio[i]>precipitacio_max) {
           precipitacio_max=precipitacio[i];
           dia_max=i;
        }
        mes=mes_dia(i);
        ndies_pluja_mes[mes]++;
        suma_pluja_mes[mes]+=precipitacio[i];
     }
   }
   pluja_mitja=suma_pluja/ndies_pluja;
   for (i=0;i<NM;i++)
      pluja_mitja_mes[i]=suma_pluja_mes[i]/ndies_pluja_mes[i];


   /* Més codi... */
   ...
}
```

Solució:

```
void pluviometria(float precipitacio[ND], int litres_risc, float pluja_mitja_mes[NM]) {
   int i, mes, ndies_pluja=0, dia_max, ndies_pluja_mes[NM];
   int ndies_risc=0, dies_risc[ND];
   float suma_pluja=0, precipitacio_max=0, pluja_mitja;
   float suma_pluja_mes[NM] ;

   /* Inicialització dels elements dels vectors a 0 */
   ...

   #pragma omp parallel
   {
      #pragma omp for private(mes) reduction(+:suma_pluja,ndies_pluja)
      for (i=0;i<ND;i++) {
         if (precipitacio[i]>0) {
            suma_pluja+=precipitacio[i];
            ndies_pluja++;
            if (precipitacio[i]>litres_risc) {
```

```
                #pragma omp critical (risc)
                {
                   dies_risc[ndies_risc]=i+1;
                   ndies_risc++;
                }
             }
             if (precipitacio[i]>precipitacio_max) {
                #pragma omp critical (maxim)
                {
                   if (precipitacio[i]>precipitacio_max) {
                      precipitacio_max=precipitacio[i];
                      dia_max=i;
                   }
                }
             }
             mes=mes_dia(i);
             #pragma omp atomic
             ndies_pluja_mes[mes]++;
             #pragma omp atomic
             suma_pluja_mes[mes]+=precipitacio[i];
          }
       }
       pluja_mitja=suma_pluja/ndies_pluja;
       #pragma omp for
       for (i=0;i<NM;i++)
          pluja_mitja_mes[i]=suma_pluja_mes[i]/ndies_pluja_mes[i];


    /* Més codi... */
       ...
}
```