

Listas de vecinos

Dividimos el recinto de simulación en celdas más chicas, de lado mayor a $\sigma + \text{skin}$.

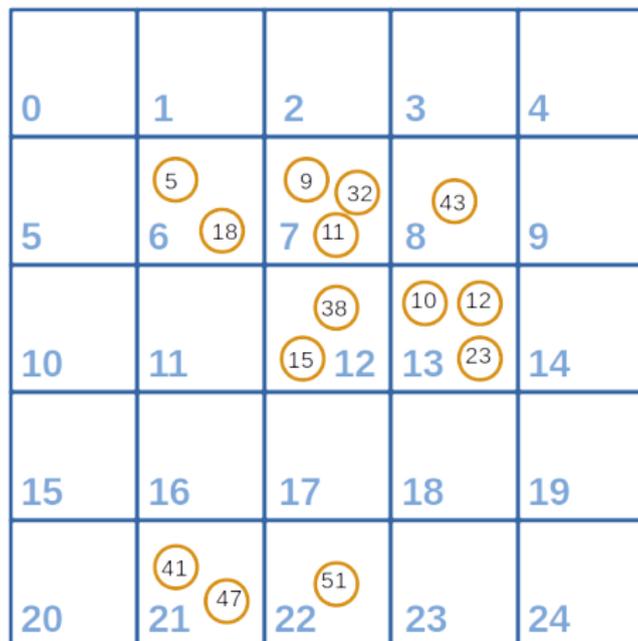


Figure: Celdas numeradas con partículas.

Organización de los datos

Las partículas se pueden ordenar como una “lista encadenda”.

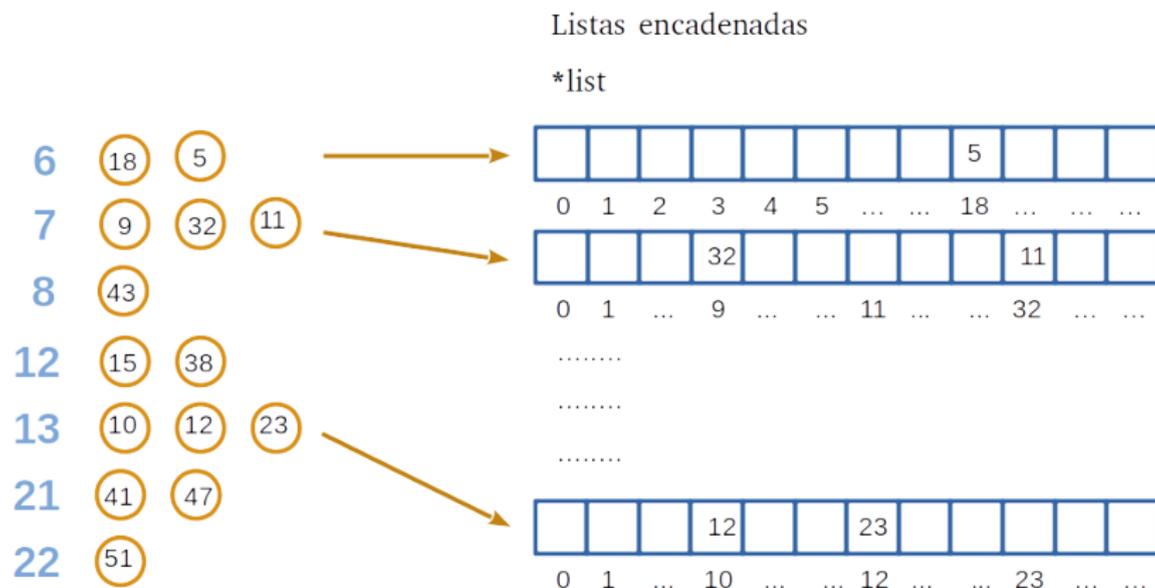


Figure: Ordenamiento de los datos.

La lista de vecinos

- (a) Podemos armar una lista única porque no hay repetición de índices ni datos entre los vectores. Llamamos al puntero que almacena esto `*list`.

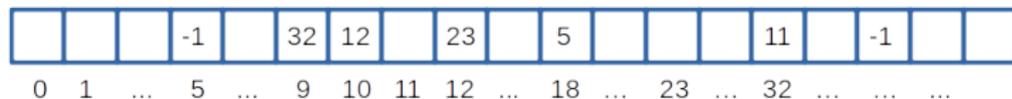
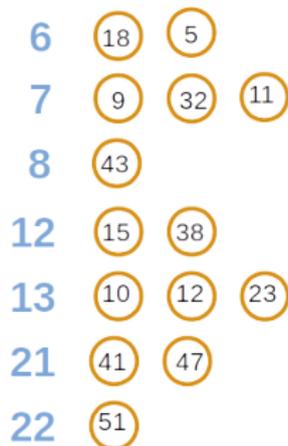


Figure: Puntero `*list`.

- (b) Para recorrer la `list` sólo hace falta una instrucción:
`j = *(list+j);`
- (c) Necesitamos un mecanismo para “terminar el recorrido” en la lista encadenada. Para “terminarlo” ponemos un valor `-1`.
`while (j != -1) j = *(list+j);`

Cabeza de la lista

(d) Necesitamos otro puntero para “iniciar el recorrido” en cada celda.



*head



Figure: Encabezamiento de la lista.

```
j = *(head+cell);  
while (j != -1) j = *(list+j);
```

Construcción de *head y *list

Pasos para construir ambos punteros (antes de comenzar la simulación!)

- (1) Inicializo *head con "-1".
- (2) Divido el recinto de simulación en celdas pequeñas. Genero la numeración de las celdas para cada partícula. Asigno *list y *head.

```
for(h=0;h<N;h++) {  
    i = (int)((*(x+3*h+0))/C);  
    j = (int)((*(x+3*h+1))/C);  
    k = (int)((*(x+3*h+2))/C);  
  
    cell = i*M*M+j*M+k;  
  
    *(list+h) = *(head+cell);  
    *(head+cell) = h;  
}
```

Ejemplo

Consideramos la celda 6:

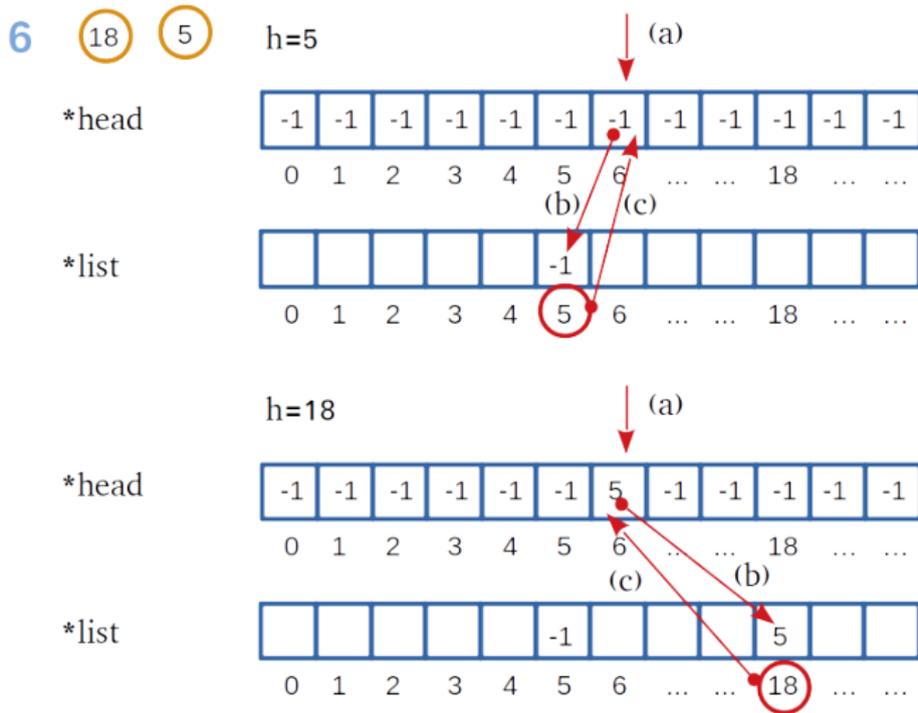


Figure: Encabezamiento de la lista.

Lectura de la lista

```
for(i=0;i<M;i++) {  
  for(j=0;j<M;j++) {  
    for(k=0;k<M;k++) {  
      c = i*M*M+j*M+k;  
      for(ii=i-1;ii<=i+1;ii++) {  
        for(jj=j-1;jj<=j+1;jj++) {  
          for(kk=k-1;kk<=k+1;kk++) {  
            cc = ((ii+M)%M)*M*M+jj*M+kk;  
            h = *(head+c);  
            while (h != -1) {  
              hh = *(head+cc);  
              while (hh != -1) {  
                [ if (i<j) calcular fuerzas entre i y j ]  
              }  
            }  
          }  
        }  
      }  
    }  
  }  
}
```