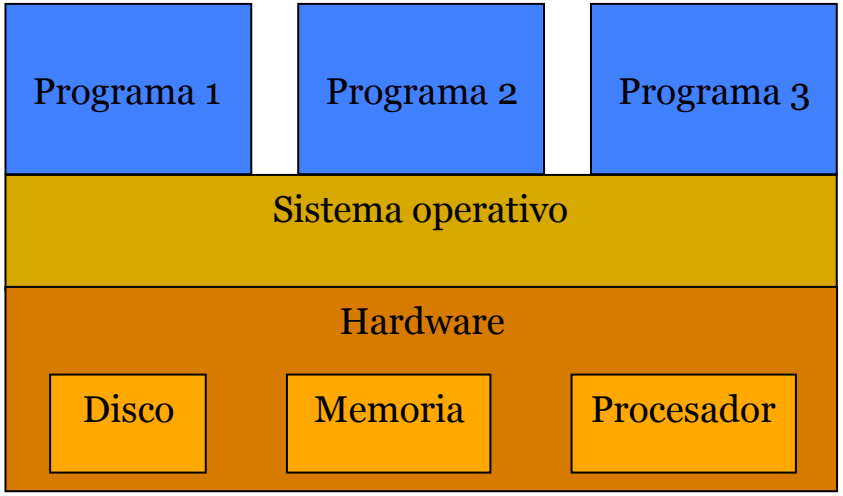
Infraestructura Computacional

Virtualización

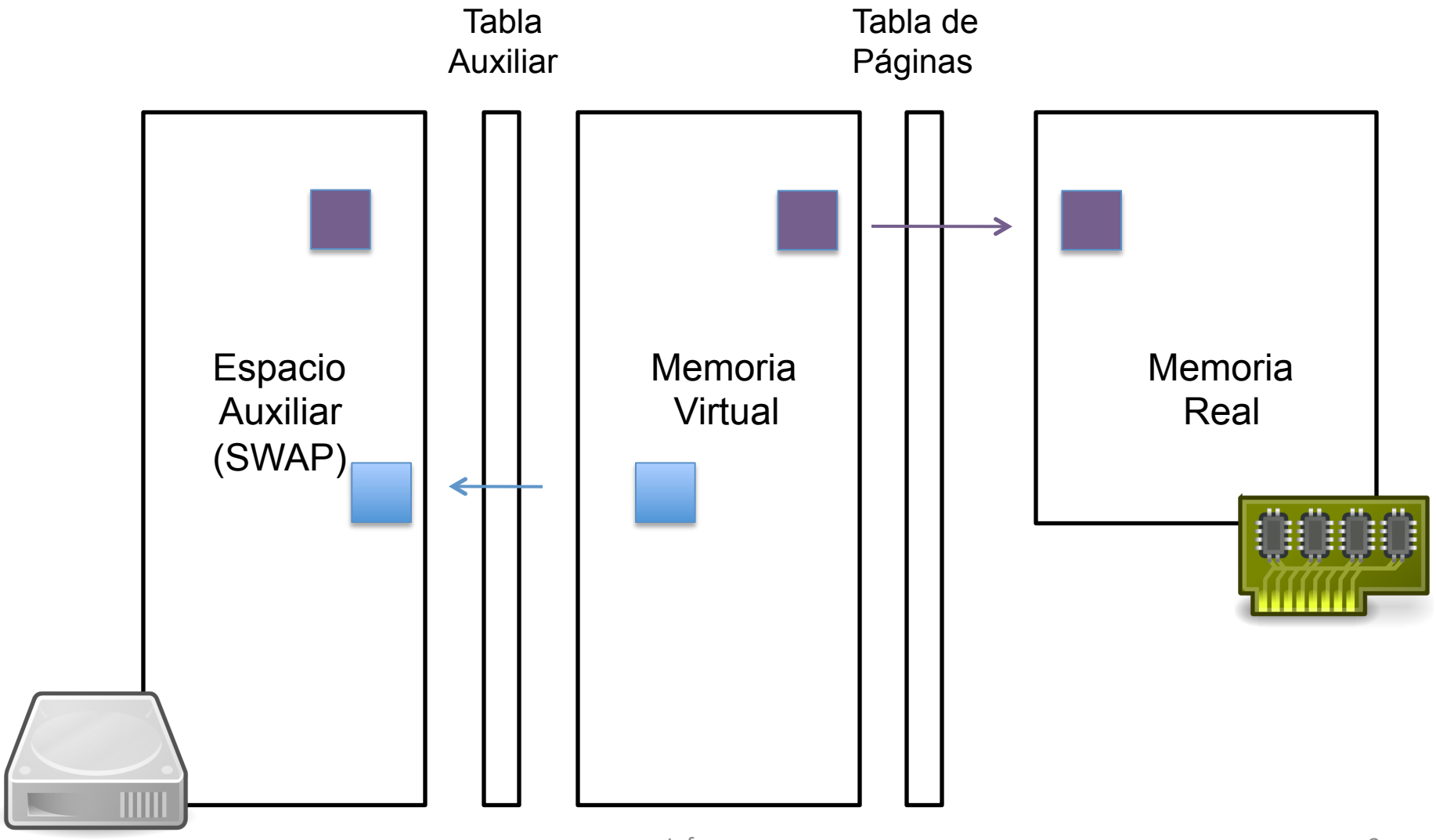
Rafael Gómez
Francisco Rueda
Sandra Rueda

Sistema Operativo Multiusuario

- Administración de los recursos de la máquina
  - Manejo de usuarios
  - Asignación de recursos
  - Virtualización de recursos
  - Contabilidad
  - Protección

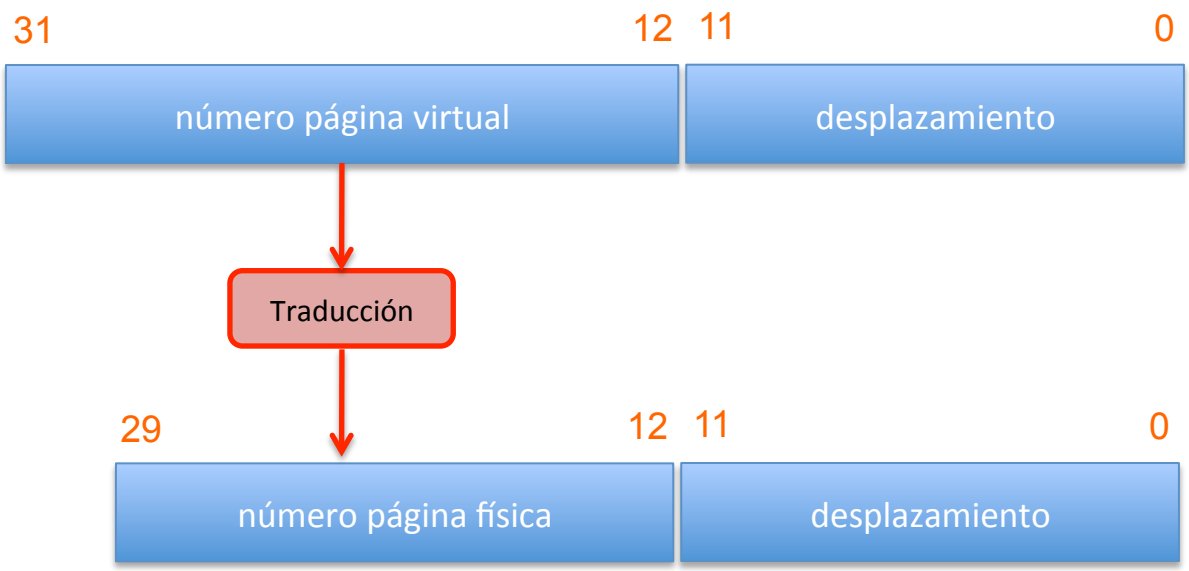


Memoria Virtual

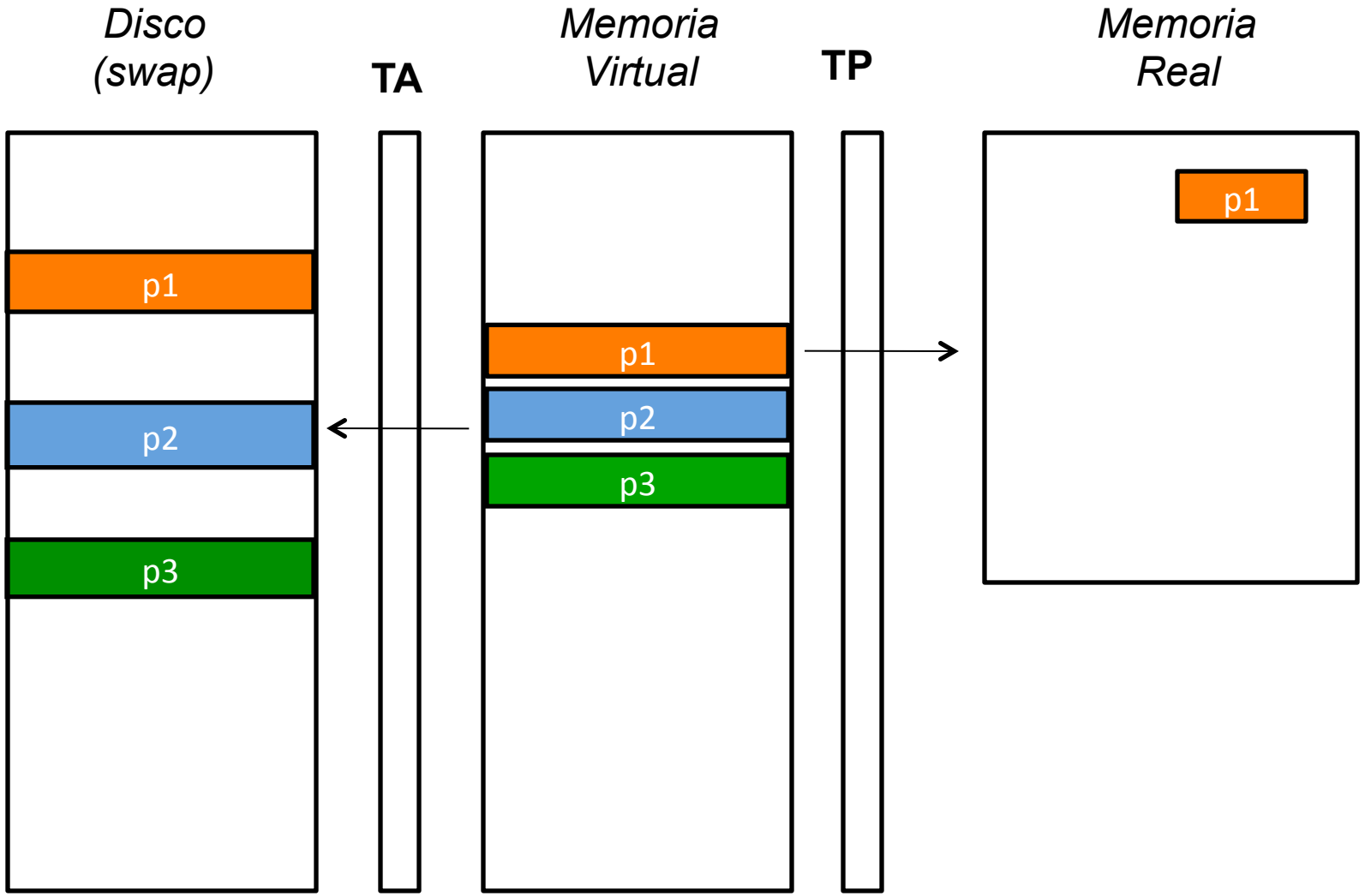


Traducción de Direcciones

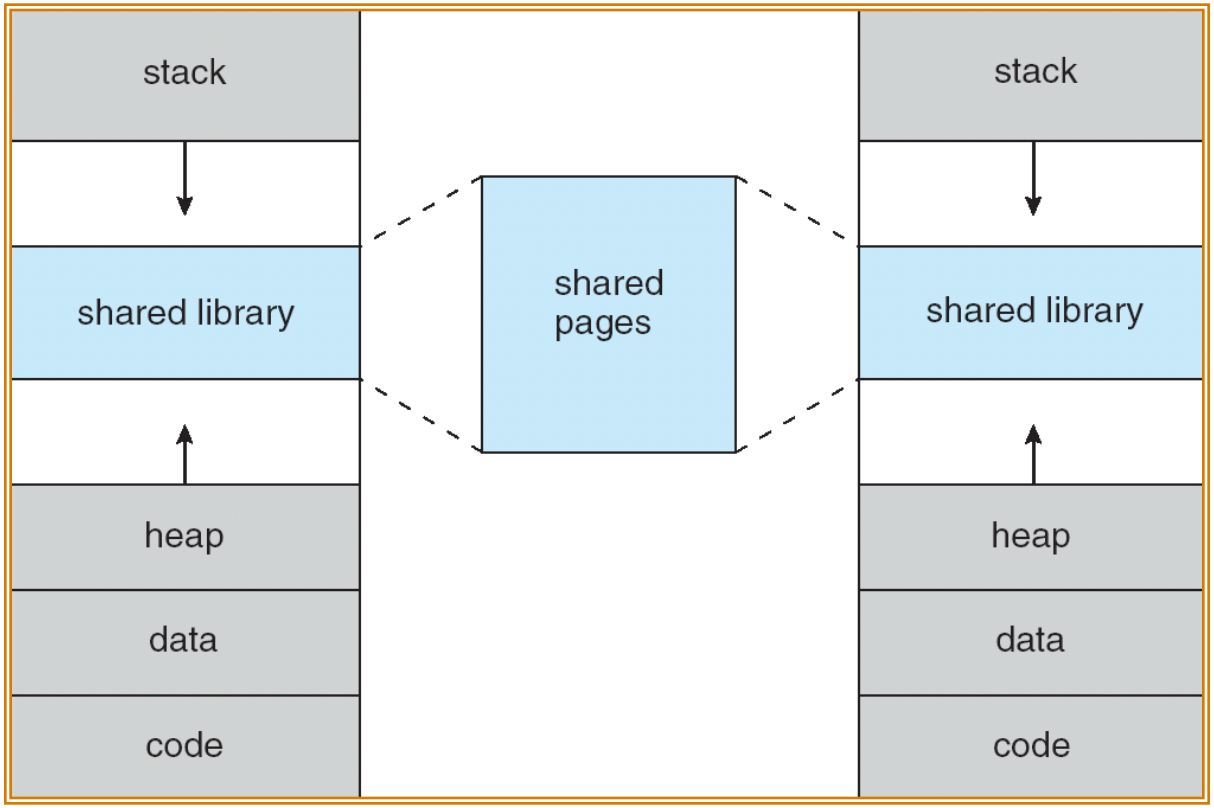
- Una dirección virtual tiene dos partes:



Carga Parcial de un Proceso



Librerías Compartidas



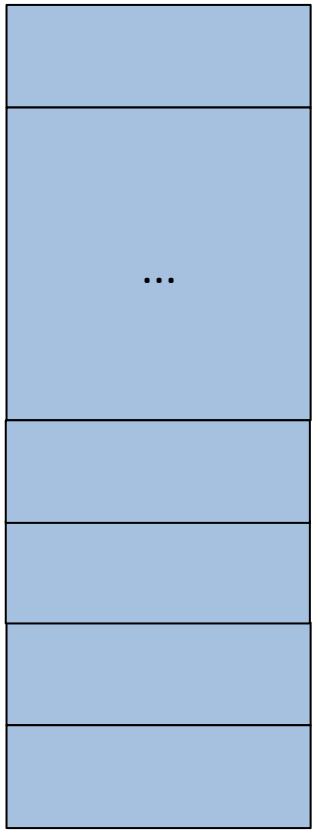
[Silberschatz, Galvin, Gagne]

Aspectos a Tener en Cuenta

- Protección
  - ¿Cómo evitar que un proceso tenga acceso a los recursos de otro?
  - ¿Cómo evitar que un proceso tenga acceso al espacio en memoria del sistema operacional?
  - ¿Cómo hacer para compartir librerías cuando sea necesario?

Protección

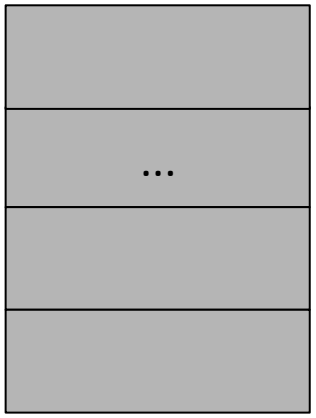
- Indicadores en la tabla de páginas



Memoria Virtual

15	X	
14		
13		
12		
11		
10		
9		
8		
7		
6		
5		
4		
3	1	E
2	X	
1	0	L
0	7	E

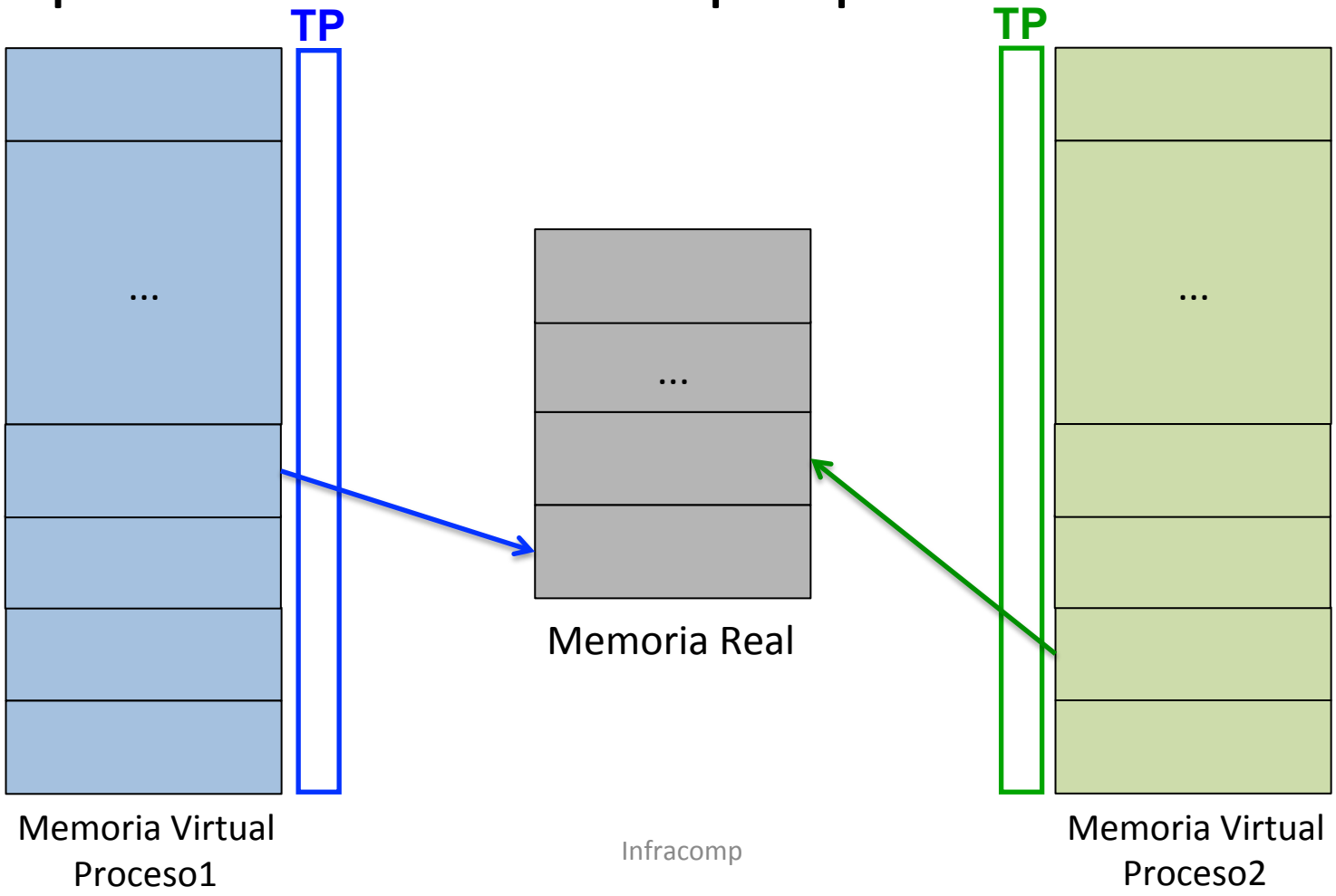
Escritura
Lectura



Memoria Real

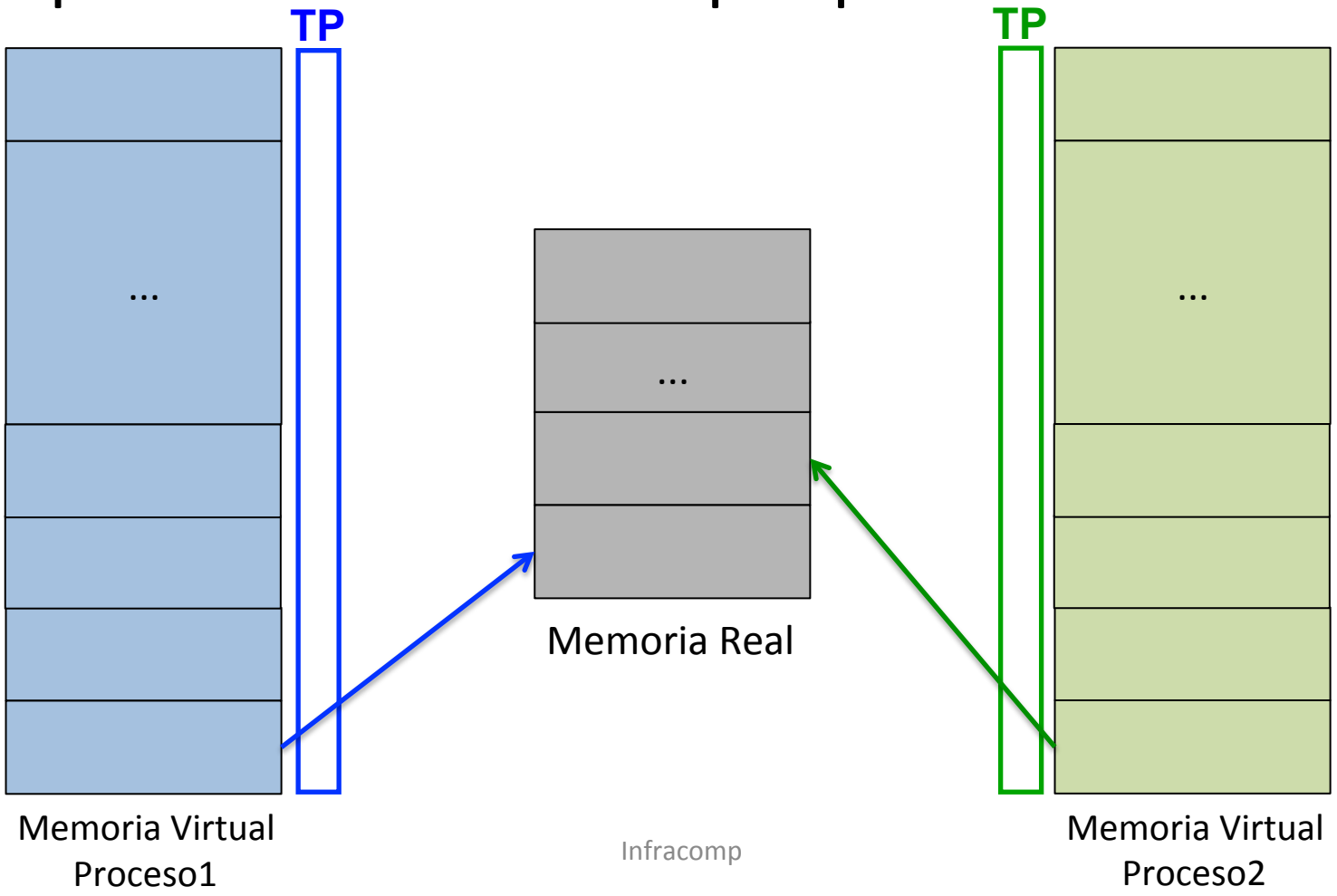
Protección

- Espacio de direcciones por proceso



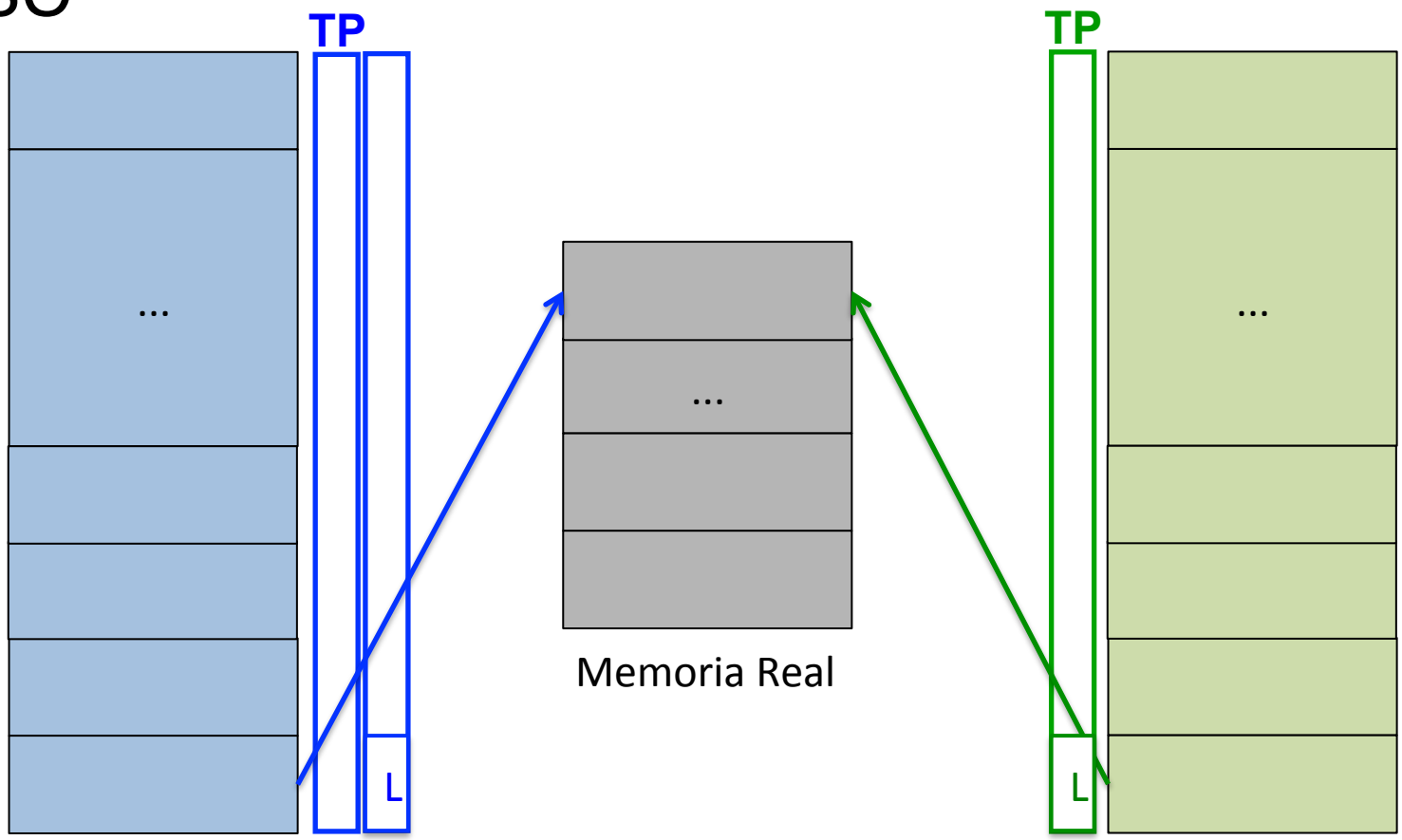
Protección

- Espacio de direcciones por proceso



Protección

- SO



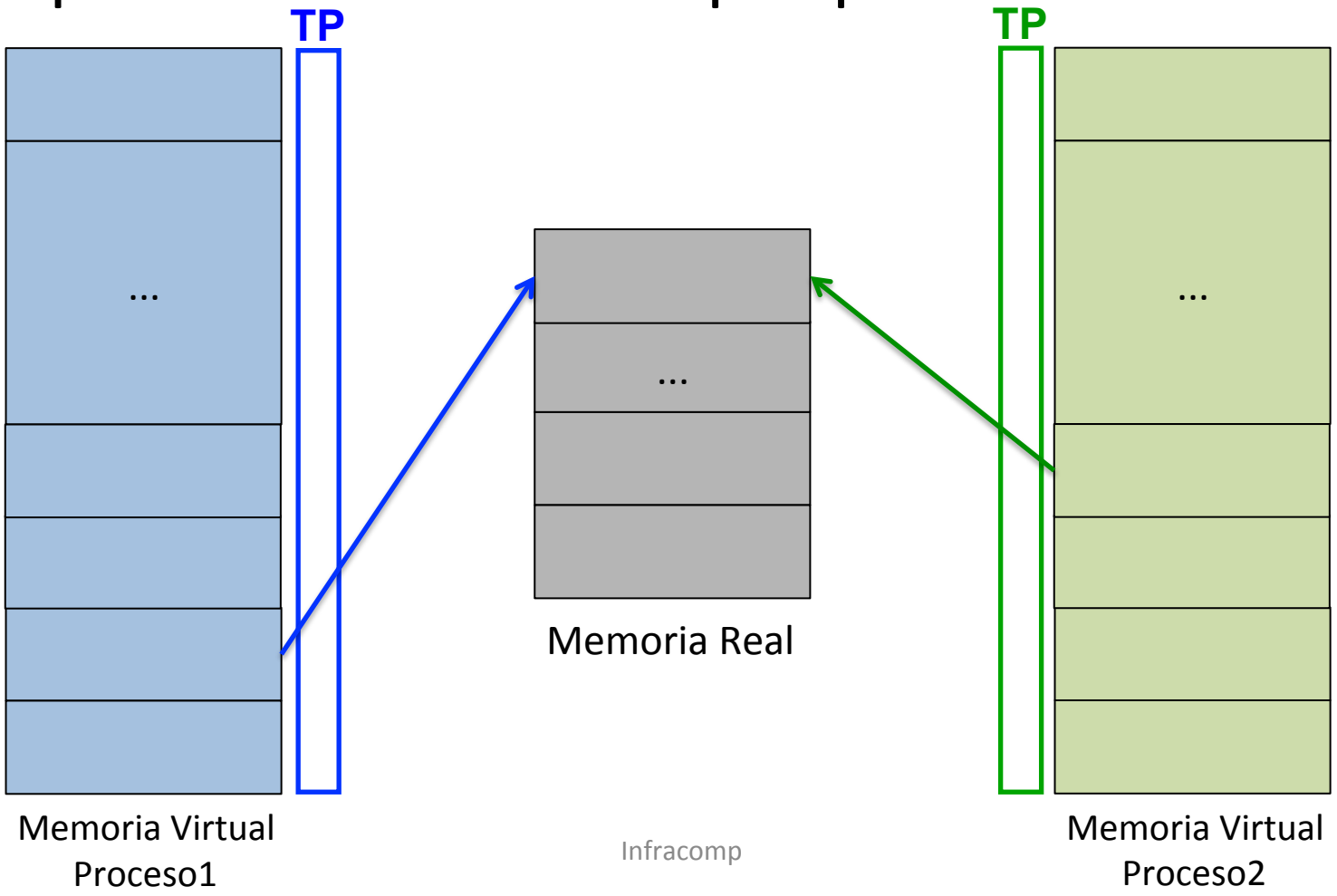
Pregunta para Discutir

¿Qué se puede hacer en un sistema de memoria virtual para que un programa no tenga acceso a una parte de la memoria que no le ha sido asignada?

- Los sistemas ofrecen mecanismos para compartir memoria haciendo que páginas de distintos espacios apunten a la misma página real

Compartir

- Espacio de direcciones por proceso



Compartir

- Lo anterior permite que el usuario pueda compartir memoria entre espacios virtuales (y por lo tanto entre procesos)

- Por ejemplo en Unix existen las primitivas shmget, shmat y shmdt para compartir información

Compartir

- La primitiva shmget permite definir una zona para compartir con otro programa
  - get shared memory segment identifier
- La primitiva shmat permite asociarla con una parte del espacio virtual
  - attach shared memory
- La primitiva shmdt permite desasociarla
  - deattach shared memory

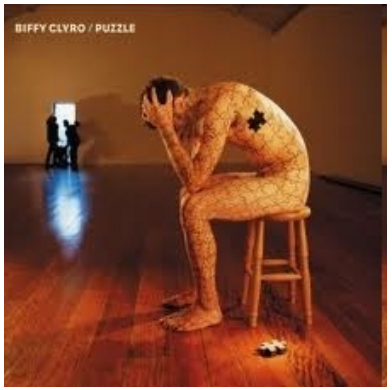
Compartir

```
Proceso 1

{
    id = shmget ( llave, tam, flags,..)
    dir1 = shmat ( id, NULL, flags,.... )
    pint = ( int * ) dir1
    /* pint apunta a la zona compartida
     * de memoria
     */
    ...
}
```

```
Proceso 2 ( id )
{
     dir2 = shmat ( id, NULL, flags,.... )
     pint = ( int * ) dir2
     /* pint apunta a la zona compartida
      * de memoria
      */
     ...
}
```

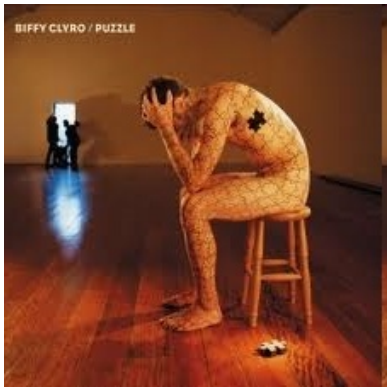
Construir un programa para hacer la copia de un archivo entre dos procesos, el primero lo lee y el segundo lo escribe, usando las primitivas para compartir memoria



Compartir

- Hay que tener en cuenta que al compartir memoria se generan problemas de exclusión mutua

¿ Qué problemas de exclusión mutua se generarían en el ejemplo anterior ?

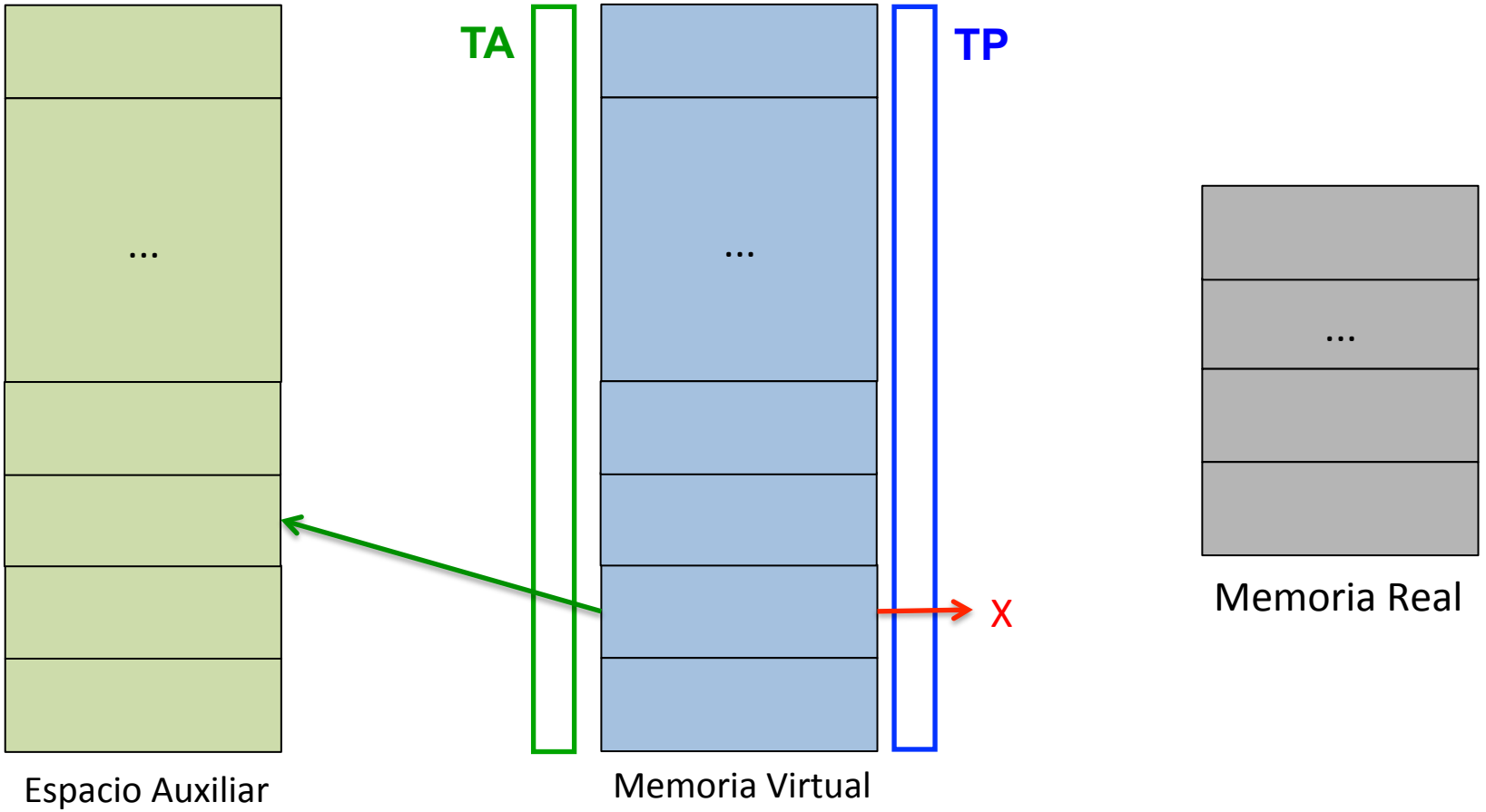


Archivos Proyectados en Memoria

- Otra posibilidad que existe es la de proyectar archivos en memoria

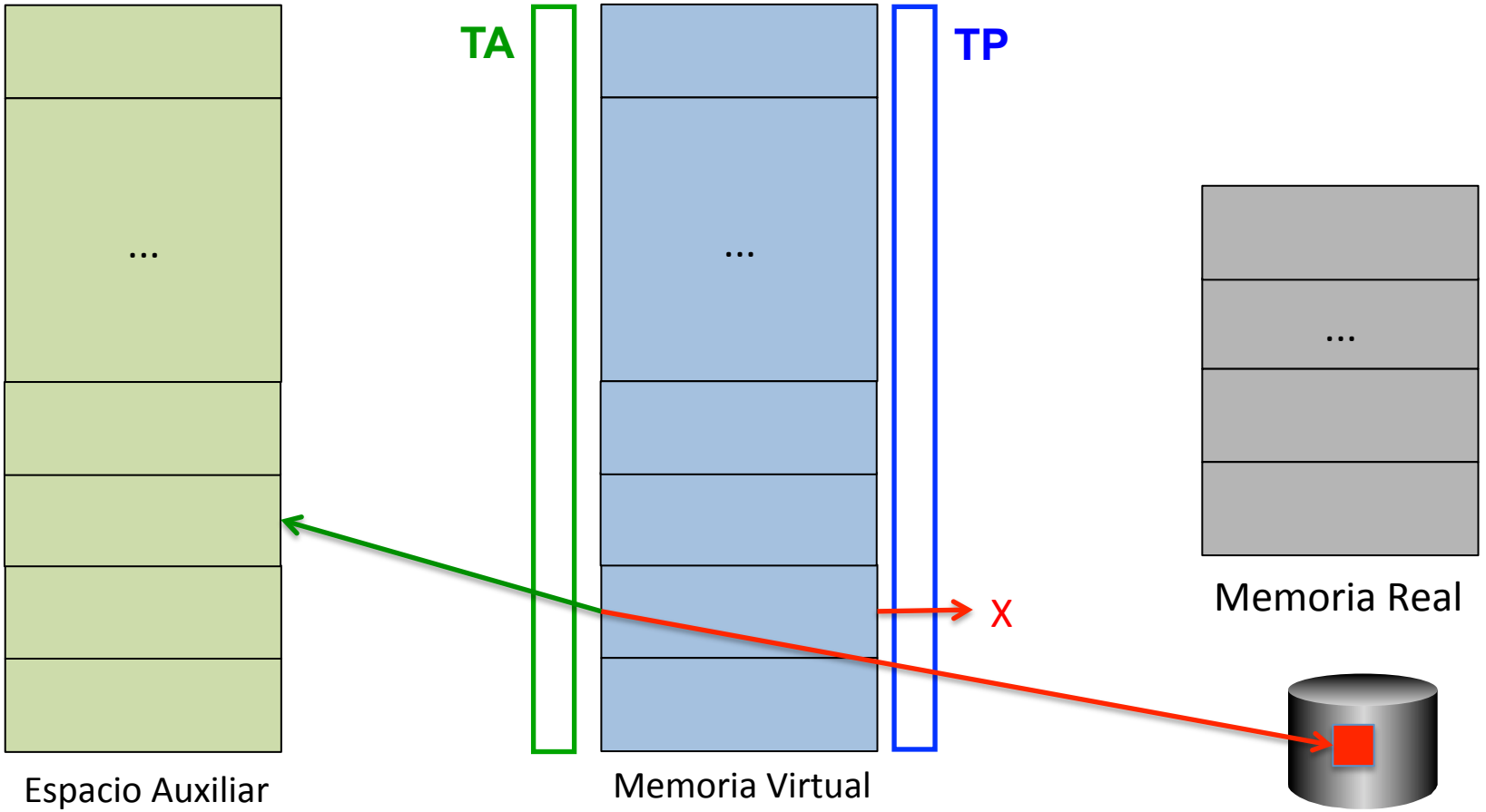
- Cuando hay un defecto de página el sistema busca la página en el disco, en el sitio que le indique la tabla auxiliar (TA)

Archivos Proyectados en Memoria



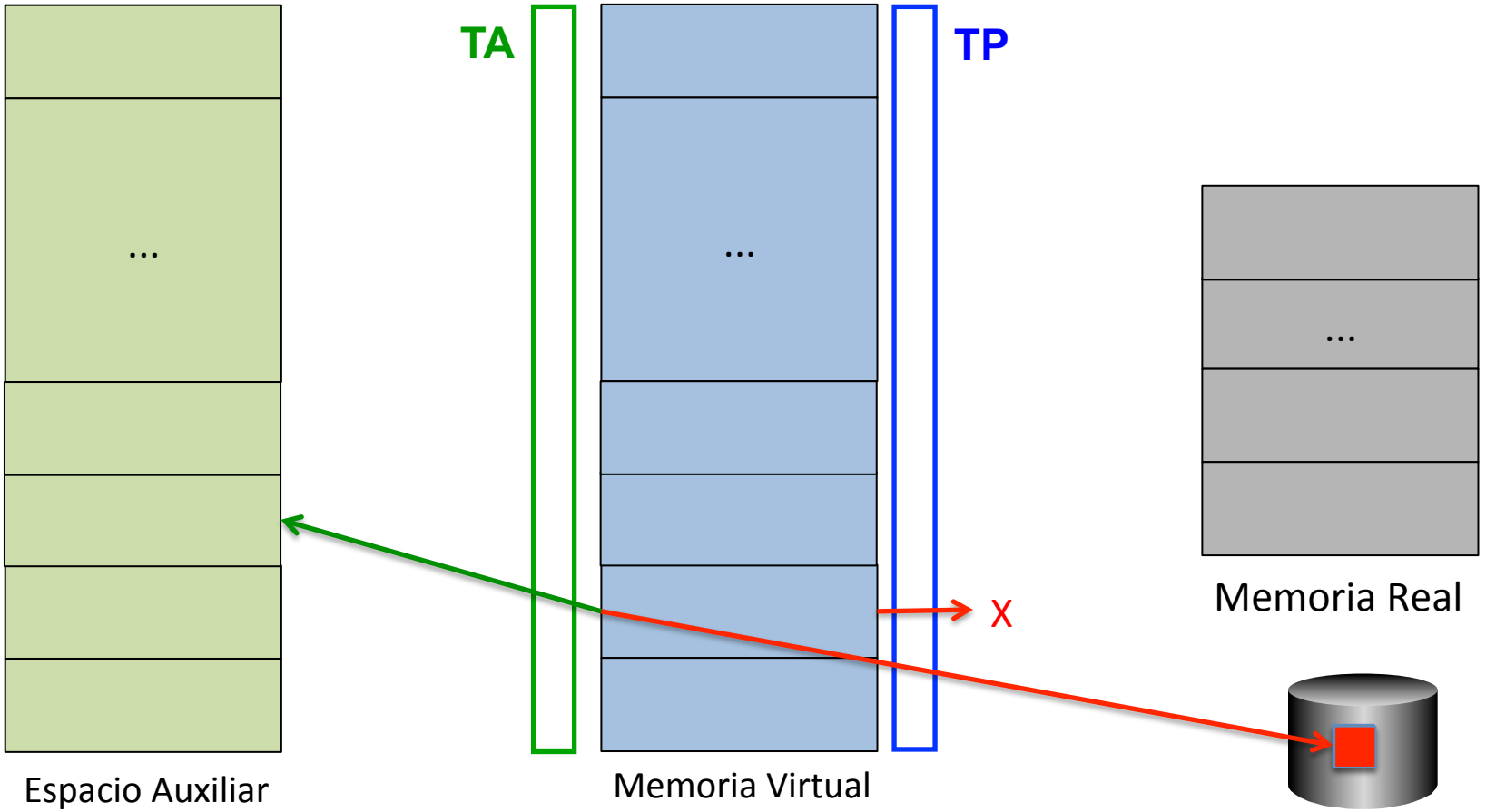
Archivos Proyectados en Memoria

- Buscar un archivo en memoria



Archivos Proyectados en Memoria

- Buscar un archivo en memoria

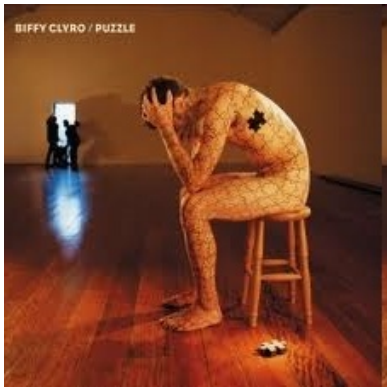


Archivos Proyectados en Memoria

- En el caso anterior el sistema buscaría la página en el archivo (y no en el espacio auxiliar), y correspondería a una operación de lectura

- La ventaja es que es un mecanismo de entrada salida muy eficiente

¿ Cómo se manejan las operaciones de escritura al archivo ?



Archivos Proyectados en Memoria

- Los sistemas tienen mecanismos para implementar lo anterior en las aplicaciones

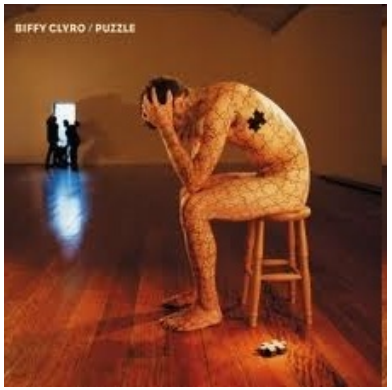
- Por ejemplo en Unix existe la primitiva mmap
  - Memory mapped

```
Proceso

{
    desc1 = open (archivo, lectura)
    dir = mmap ( 0, long, prot, flag, desc1,0)
    pint = ( int * ) dir
    for ( i=0; i < long; i++ ) {
        printf ( "%d", *pint ++)
    }
    close ( desc1)
}
```

prot = READ | WRITE

Construir un programa para copiar un archivo utilizando proyección de memoria en el archivo original y en la copia



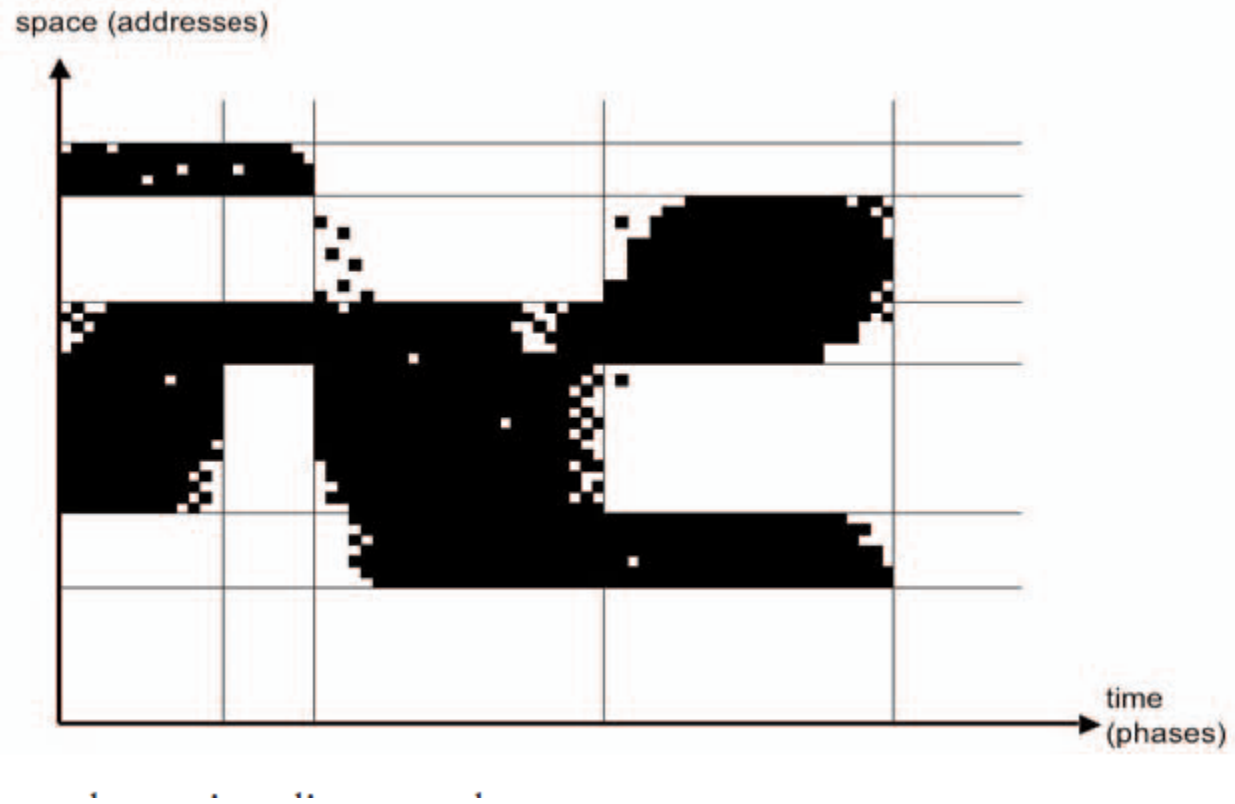
Aspectos a Tener en Cuenta

- Miraremos ahora algunos aspectos que hay que tener en cuenta en el manejo de la memoria real

Aspectos a Tener en Cuenta

- Las referencias que hacen los programas a las diferentes páginas de instrucciones y datos no son uniformes sino que tienden a concentrarse en unas ciertas páginas, es lo que se llama el principio de localidad

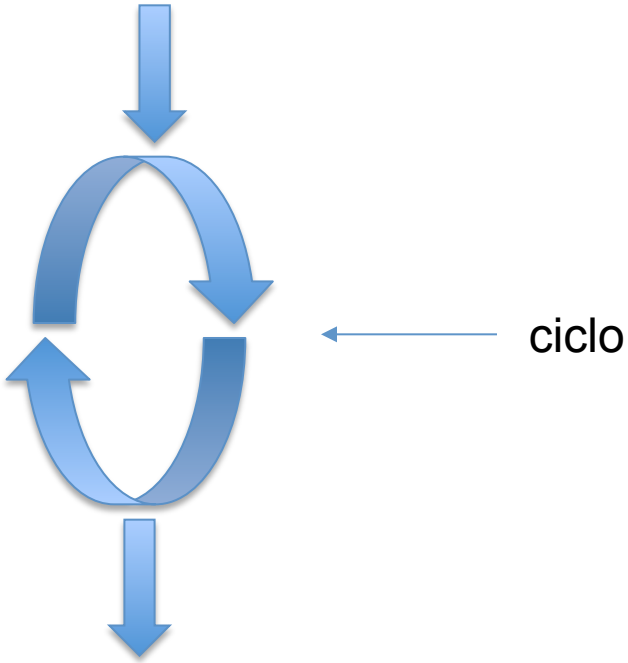
Principio de Localidad



Tomado de The Locality Principle, P. Denning, CACM, 2006

Principio de Localidad

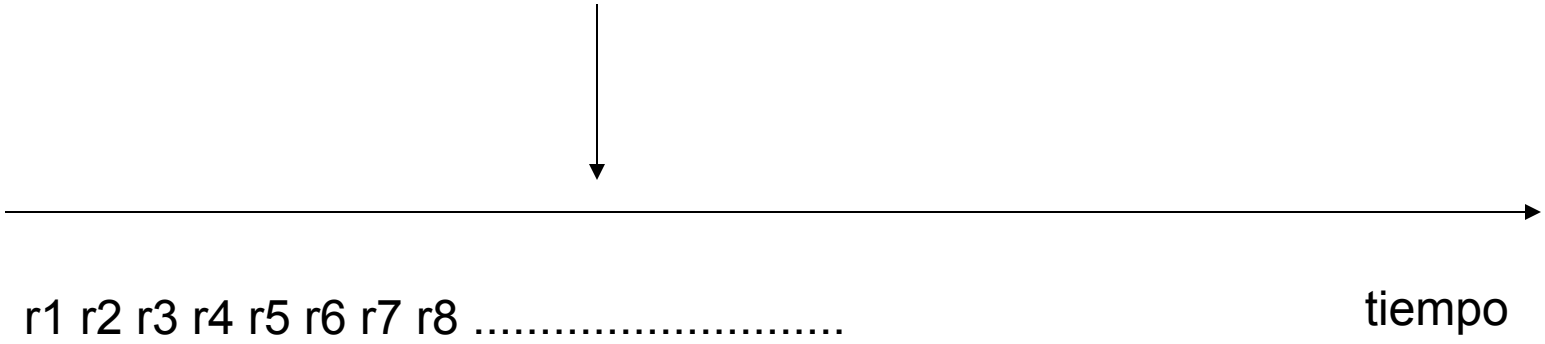
Instrucciones:



Datos?

Principio de Localidad

- Otra forma de ver el principio de localidad es decir que si una página fue referenciada recientemente, es muy probable que sea referenciada en el futuro próximo



Algoritmos de Remplazo

- Cuando se requiere traer una página del disco es necesario encontrarle un puesto en la memoria, para esto se usan los algoritmos de remplazo

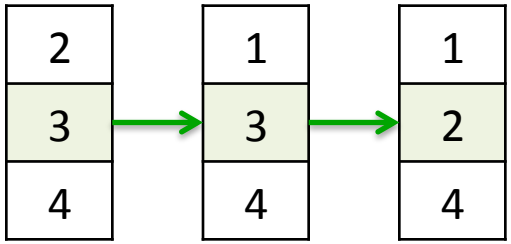
- El algoritmo debe decidir qué página reemplazar

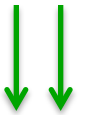
Algoritmos de Remplazo

- Existen varios algoritmos de reemplazo posibles:
  - FIFO
  - LRU
  - Bit de referencia y de cambio
- El algoritmo ideal?

Algoritmos de Remplazo

Suponga un sistema con 3 marcos de memoria disponibles:





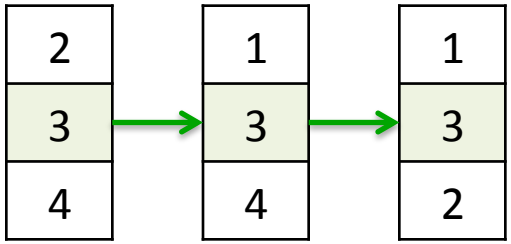
2 3 2 4 3 1 2 3 1 4 3 1 2 3 4 .....

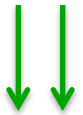
tiempo

FIFO

Algoritmos de Remplazo

Suponga un sistema con 3 marcos de memoria disponibles:





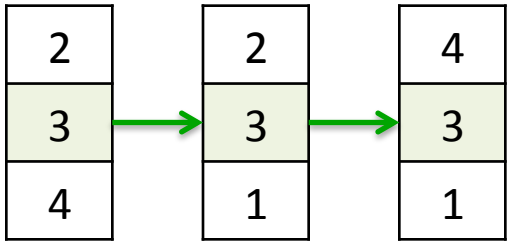
2 3 2 4 3 1 2 3 1 4 3 1 2 3 4 .....

tiempo

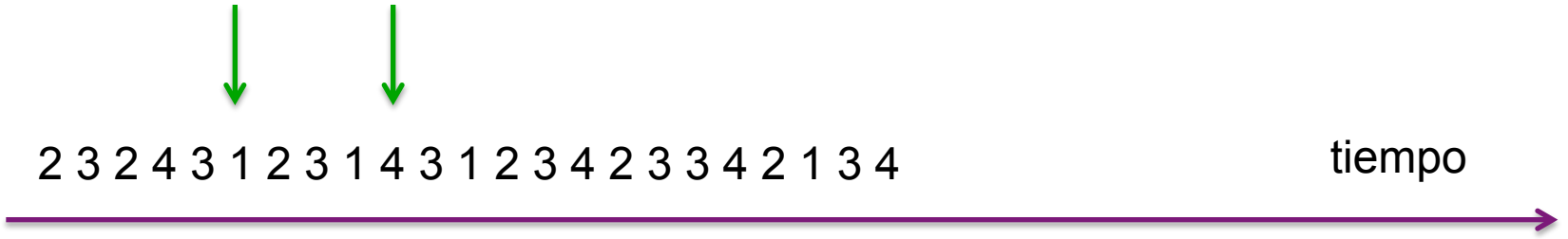
LRU

Algoritmos de Remplazo

Suponga un sistema con 3 marcos de memoria disponibles:



Ideal: que va a ser referenciada en mayor tiempo en el futuro.



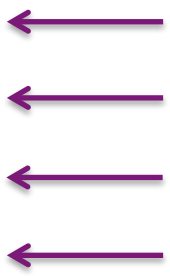
Ideal

Algoritmos de Remplazo

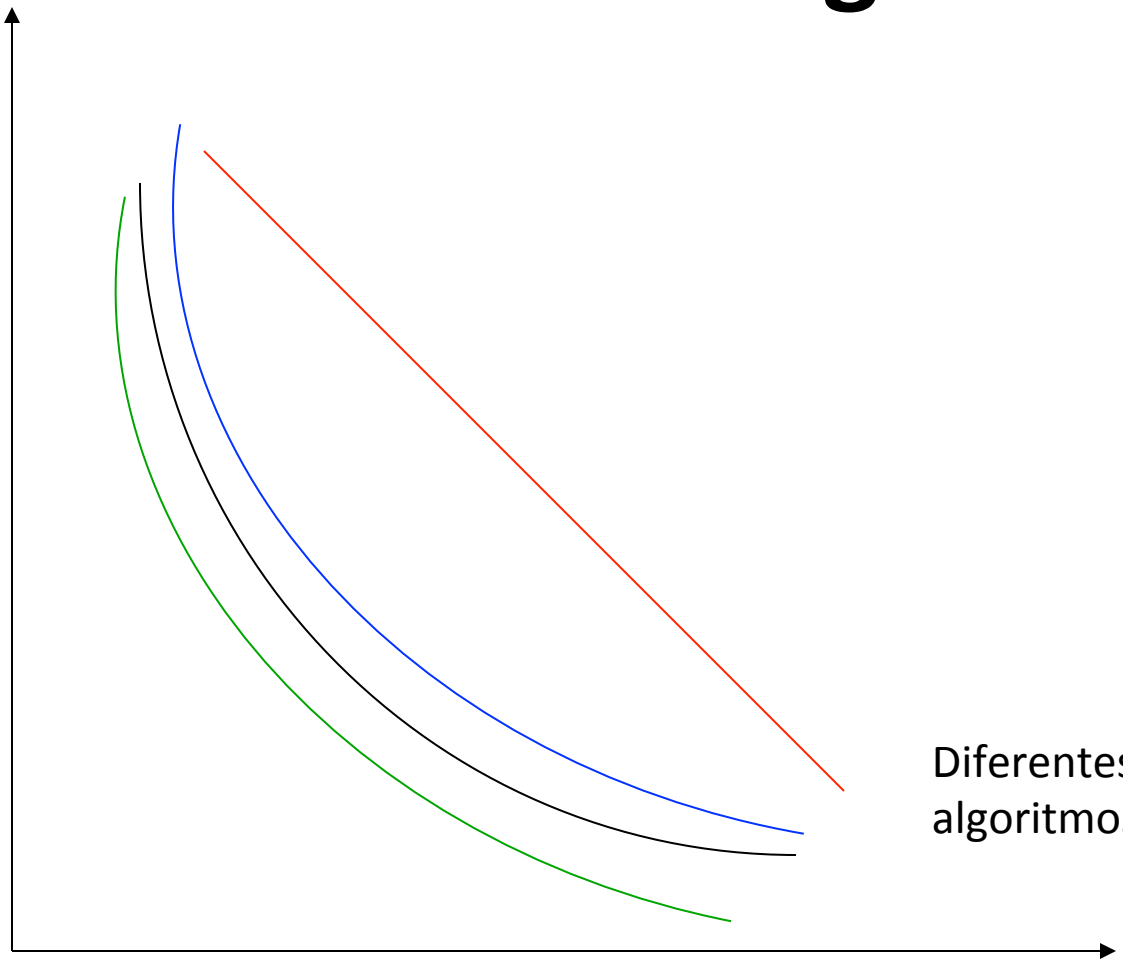
- El método del bit de referencia/cambio consiste en tener en cuenta si las páginas fueron referenciadas y/o cambiadas y registrarlo

- Habrá cuatro grupos

Referenciadas	Cambiadas
0	0
0	1
1	0
1	1



Número de Fallas de Página

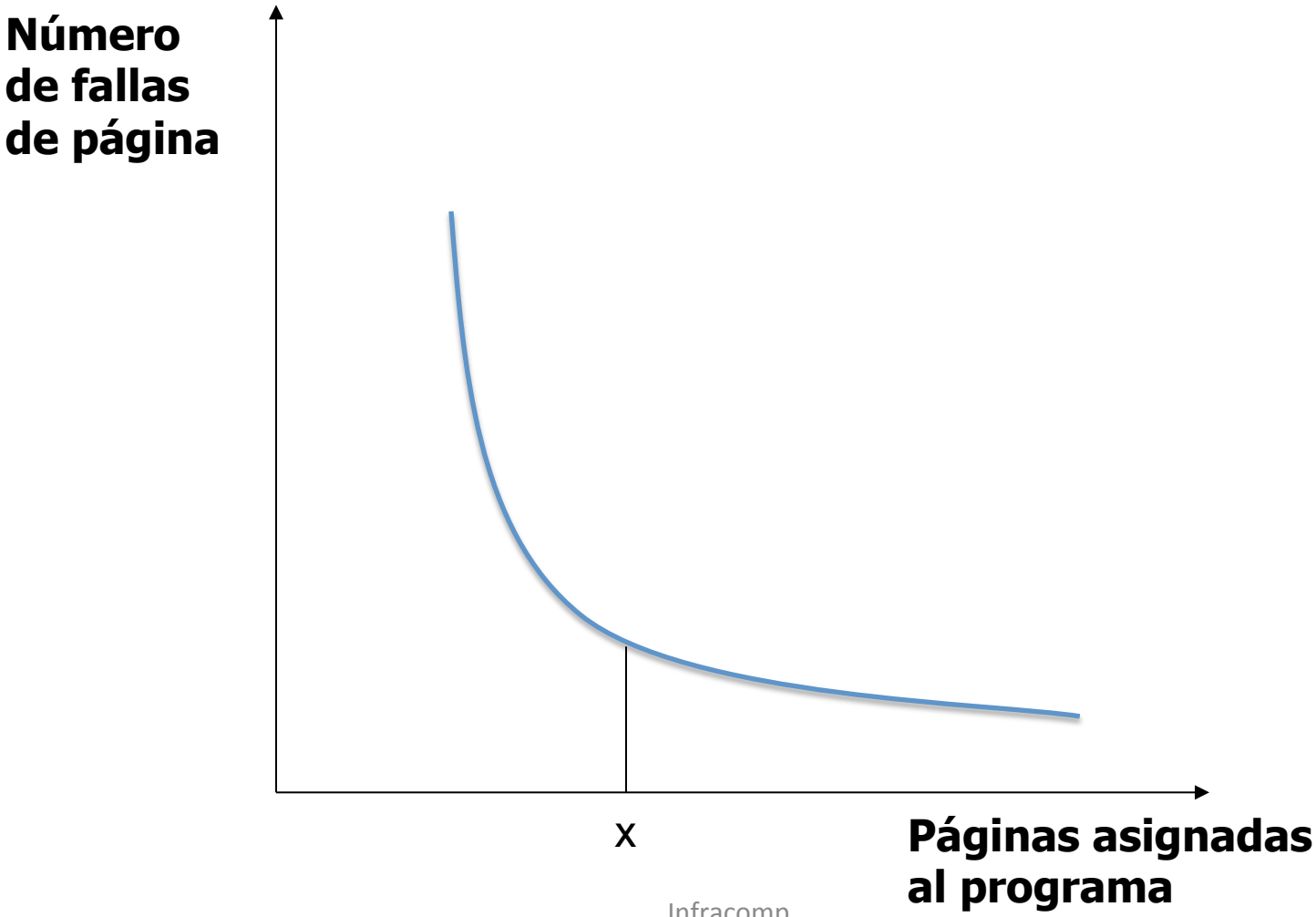


Manejo de Memoria

- El sistema debe controlar la cantidad de programas que carga y/o de páginas que le da a cada programa para que el sistema funcione adecuadamente

- En caso de que no se haga puede haber serios problemas de desempeño en el sistema (thrashing)

Manejo de Memoria



Manejo de Memoria

- Según la gráfica anterior, para todo programa existe un número *x* tal que si se le asignan menos páginas el número de fallas de página crece considerablemente

- Es lo que se denomina el espacio de trabajo

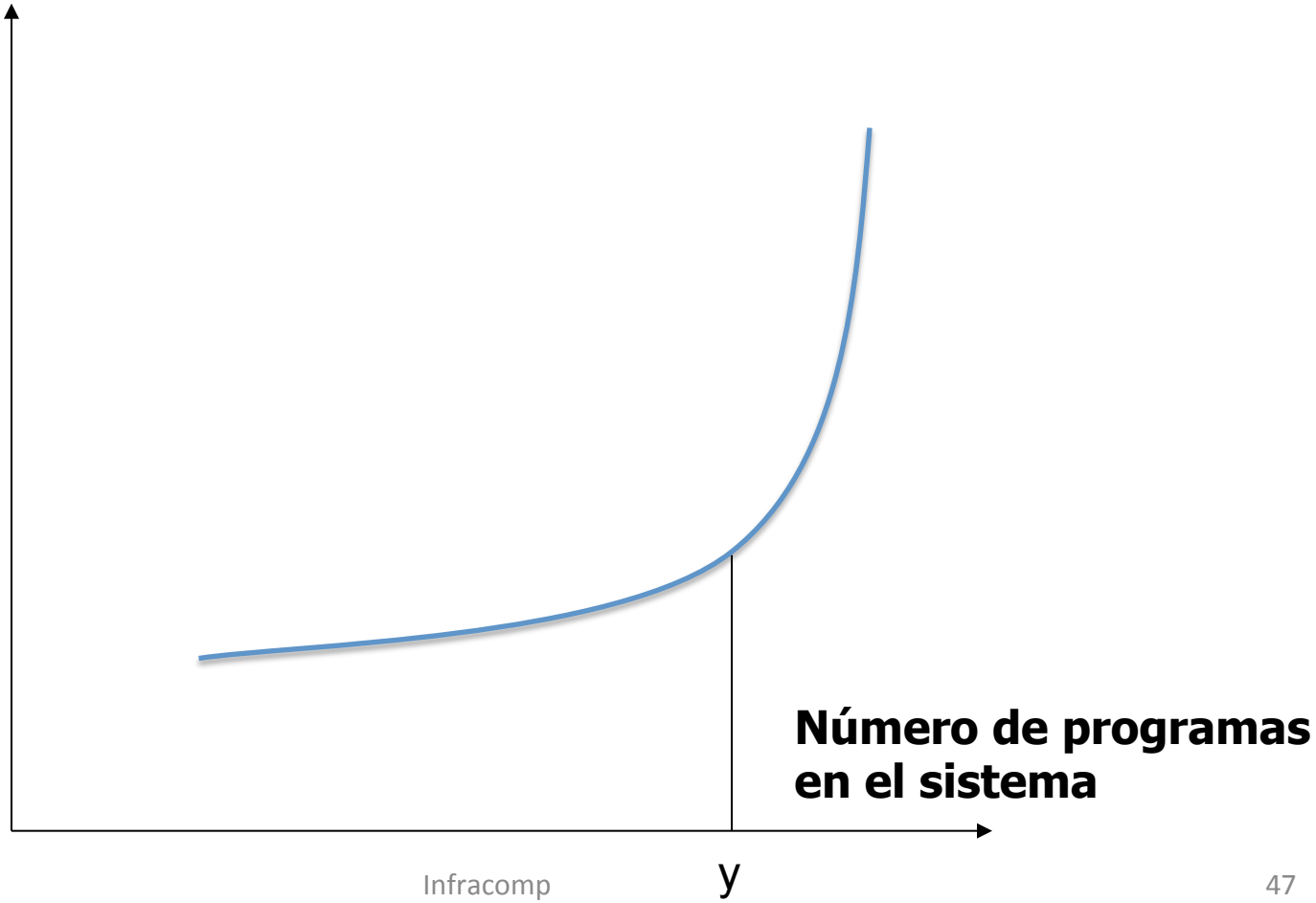
Manejo de Memoria

- La idea es entonces que una aplicación sólo se debería correr si hay suficiente espacio en memoria para contener su espacio de trabajo (de lo contrario la aplicación debe esperar)

- ¿Cómo calcular el tamaño del espacio de trabajo?

Manejo de Memoria - Thrashing

Número de fallas de páginas en el sistema

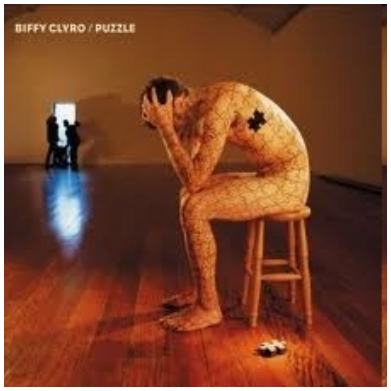


Manejo de Memoria

- En la gráfica anterior se puede ver que cuando el número de programas sobrepasa un cierto umbral *y* , el desempeño del sistema disminuye mucho

- La idea es entonces controlar que nunca se sobrepase ese umbral
  - si el número de fallas de página crece mucho se desactiva un programa

¿Cómo influye la memoria virtual en el desempeño de los programas?



Memoria Virtual

- Es un mecanismo que ofrece flexibilidad, eficiencia y seguridad
  - Pero tiene costos de desempeño
  - Manejado de forma adecuada los costos "desaparecen"

Referencias

- Fundamentos de Sistemas Operativos, Silberschatz, Galvin y Gagne,. Ed. McGrawHill, 2006. Capítulos Administración de la memoria y Memoria virtual.