Arquitectura de Computadoras para Ingeniería

(Cód. 7526)
1° Cuatrimestre 2016

Dra. Dana K. Urribarri
DCIC - UNS

Pipelining

Pipeline lineal

- Una secuencia de etapas que están linealmente conectadas para realizar una tarea fija sobre un flujo de datos que circulan de un extremo al otro.

- El pipeline tiene *k* etapas de procesamiento $S_1 \ldots S_k$
- Las entradas externas ingresan al pipeline por la etapa S_1
- Los resultados del procesamiento de la etapa S_i es la entrada de la etapa S_{i+1}.
- El resultado final es la salida de la etapa S_k.

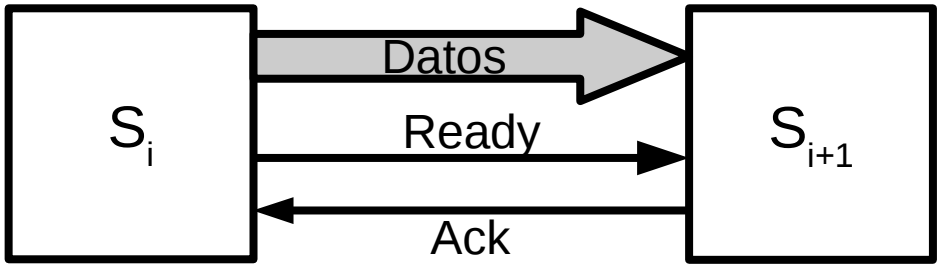
Pipeline lineal

- Una tarea se divide en *k* subtareas.
- La ejecución de las *k* subtareas equivalen a la ejecución de la tarea original.

- Cada subtarea se ejecuta con un hardware específico.
- Cada subtarea puede ejecutar concurrentemente con otras subtareas del pipeline.

Modelos de pipeline

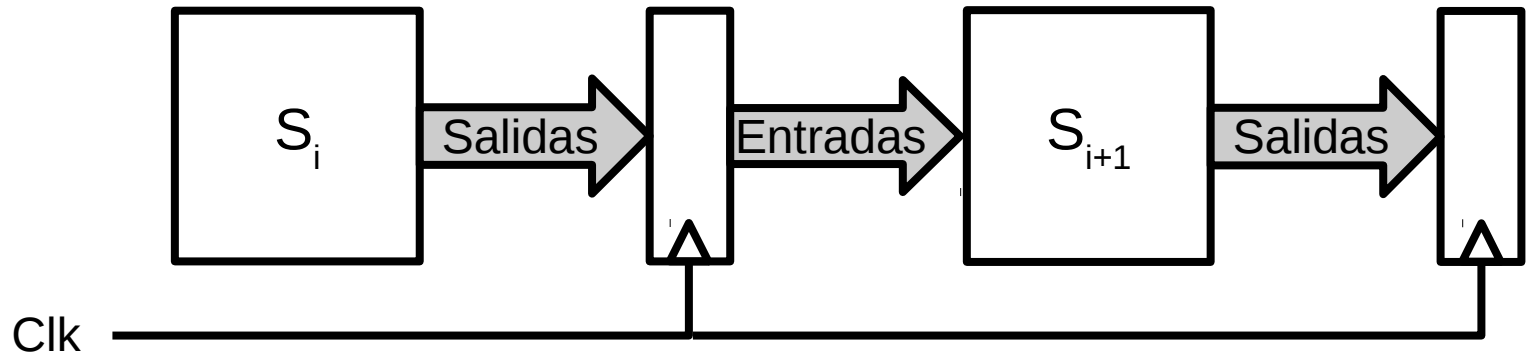
- Modelo Asincrónico
  - El flujo de datos entre etapas adyacentes se realiza a través de un protocolo de sincronización (handshaking).
  - Cuando la etapa S_i está lista para transmitir le envía una señal de *ready* a la etapa S_{i+1}. Cuando la etapa S_{i+1} recibe el dato le envía a S_i una señal de *acknowledge*.



Modelos de pipeline

Vamos a considerar sólo pipelines sincrónicos

- Modelo Sincrónico
  - Las etapas del pipeline están separadas por registros habilitados por pulsos de un reloj.
  - Aíslan las salidas de una etapa con las entradas de la siguiente.
  - Con la habilitación del pulso de reloj todos los registros transfieren los datos a la próxima etapa.



Pipeline lineal

- Cada etapa es combinacional.
- Es deseable que todas las etapas tengan aproximadamente el mismo retardo.
- El máximo retardo de las etapas determina el período del reloj y por lo tanto la velocidad del pipeline.

Reloj

- Si τ_i es el retardo de la etapa S_i y d es el retardo de los registros interetapas, entonces el reloj τ es

$$\tau = \max\{\tau_i\} + d$$

$$\max\{\tau_i\} >> d$$

- La frecuencia del pipeline se define como la inversa del reloj

$$f = 1/\tau$$

- El máximo *throughput* es f.

Tiempos de procesamiento

- Idealmente, un pipeline de *k* etapas procesa *n* tareas en *k* + *n*–1 ciclos de reloj.
  - k ciclos para completar la primera tarea
  - *n*–1 ciclos para completar las tareas restantes.

  El tiempo total requerido será

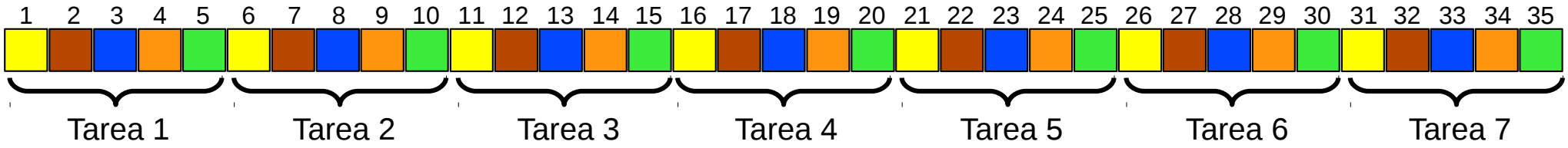
$$T_k = (k + n–1)\ \tau$$

- En un procesador no en pipeline cada tarea demandaría un tiempo de $k\tau$

  Las *n* tareas demandarían un tiempo total de $nk\tau$.

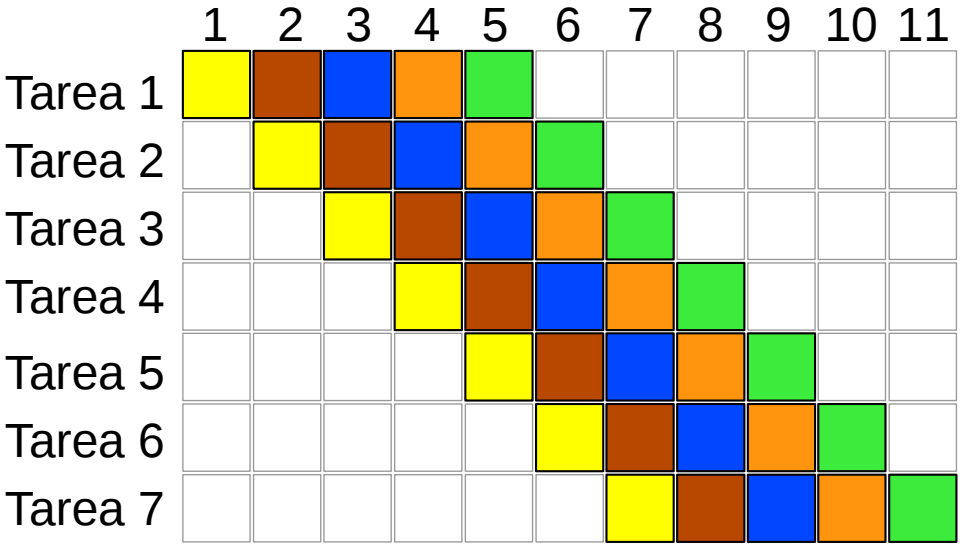
Tiempos de procesamiento

- 7 tareas – 5 etapas de retardo τ c/u
- De forma secuencial: 7 x 5τ = 35τ



- En pipeline:

$5\tau + (7-1)\tau = 11\tau$



Medidas de desempeño

Speedup

- El *speedup* de un pipeline de *k* etapas, en comparación a un procesamiento lineal es

$$\text{Speedup}_k = \frac{T_1}{T_k} = \frac{n k \tau}{k\tau + (n-1)\tau} = \frac{nk}{k + (n-1)}$$

$$\lim_{n \to \infty} \text{Speedup}_k = k$$

- El máximo *speedup* se logra con $n >> k$

Medidas de desempeño

Eficiencia

- Mide la utilización de los recursos (las *k* etapas)

$$\text{Eficiencia} = \frac{\text{Speedup}_k}{k} = \frac{nk}{k\left(k+\left(n-1\right)\right)}$$

$$= \frac{n}{k+\left(n-1\right)}$$

$$\lim_{n\to\infty} \text{Eficiencia} = 1$$

Medidas de desempeño

Throughput

- Es la cantidad de tareas que se completan por unidad de tiempo.

$$\text{Throughput} = \frac{n}{\tau\left(k+n-1\right)} = \frac{\text{Eficiencia}}{\tau}$$

- Cuando la cantidad de tareas tiende a infinito, la eficiencia es máxima y el *throughput* iguala la frecuencia del pipeline:

$$\lim_{n\to\infty} \text{Throughput} = \lim_{\text{Eficiencia}\to 1} \frac{\text{Eficiencia}}{\tau} = 1/\tau = f$$

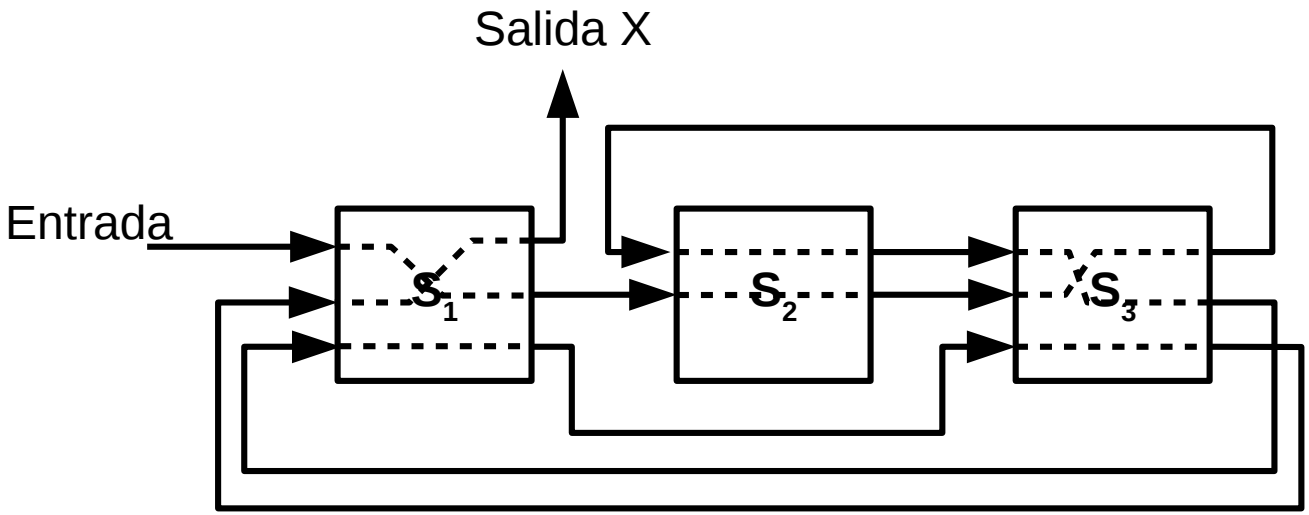
Pipeline no lineal

- En el pipeline lineal las conexiones son $S_i \rightarrow S_{i+1}$
- En un *pipeline no lineal* permite
  - Conexiones lineales $S_i \rightarrow S_{i+1}$
  - Conexiones hacia adelante $S_i \rightarrow S_j$ tal que $i + 1 < j$
  - Conexiones hacia atrás $S_i \rightarrow S_j$ tal que $j < i$
- Cualquier estado puede generar la salida.
- La planificación de los eventos sucesivos no es trivial.

Pipeline no lineal

- Tabla de reservación
- Representa el flujo de las tareas en el pipeline.
  - Las filas representan las etapas del pipeline.
  - Las columnas representan los ciclos de operación.
  - Se marca en qué ciclo la tarea se encuentra en cada etapa.
    - Más de una marca por fila implica que en más de un ciclo se usa la misma etapa.
    - Más de una marca por columna, implica que en un mismo ciclo se utilizan en paralelo más de una etapa.

Pipeline no lineal



	1	2	3	4	5	6	7	8
S_1	X					X		X
S_2		X		X				
S_3			X		X		X	

Colisiones

- Iniciación

  Corresponde al comienzo de una evaluación.

- Latencia:

  Cantidad de ciclos entre dos iniciaciones.

- Colisión:

  Hay un conflicto de recursos porque dos iniciaciones intentan usar la misma etapa.

- Latencias prohibidas:

  Latencias que generan colisión.

Colisiones

	1	2	3	4	5	6	7	8
S_1	X					X		X
S_2		X		X				
S_3			X		X		X	

Latencia 2

	1	2	3	4	5	6	7	8	9	10
S_1	A		B			A		AB		B
S_2		A		AB		B				
S_3			A		AB		AB		B	

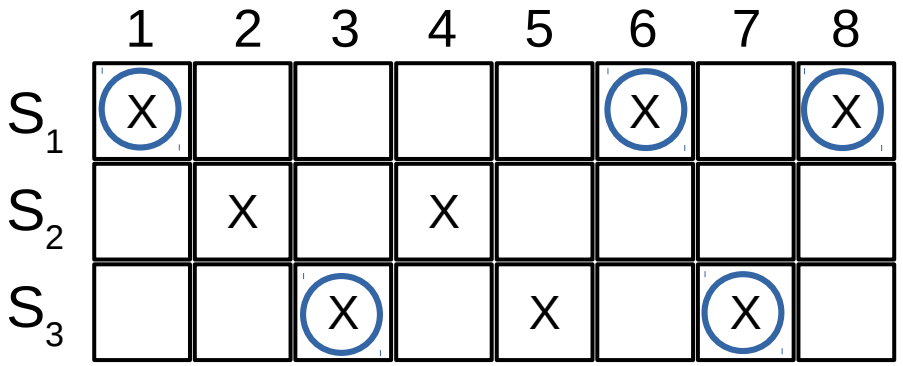
Latencia 3

	1	2	3	4	5	6	7	8	9	10	11
S_1	A			B		A		A	B		B
S_2		A		A	B		B				
S_3			A		A	B	A	B		B	

Latencias prohibidas

- Las latencias prohibidas se detectan controlando las marcas en una misma fila de la tabla de reservación.

- Las latencias prohibidas son:
  - 2
  - 4
  - 5
  - 7



Ciclo de latencias

- Secuencia de latencias (no prohibidas) que se repite indefinidamente.
  - Ciclo de latencias: {3}

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
S_1	A			B		A	C	A	B	D	B	C	E	C	D	F	D	E		E
S_2		A		A	B		B	C		C	D		D	E		E	F		F	
S_3			A		A	B	A	B	C	B	C	D	C	D	E	D	E	F	E	F

  - Ciclo de latencias: {1,8}

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
S_1	A	B				A	B	A	B	C	D				C	D	C	D	E	
S_2		A	B	A	B						C	D	C	D						E
S_3			A	B	A	B	A	B				C	D	C	D	C	D			

Ciclo de latencias

- La *latencia promedio* de un ciclo de latencias es la suma de las latencias del ciclo dividido la cantidad de latencias.

- Un *ciclo constante* es un ciclo de latencias con un solo valor.

Planificación sin colisiones

- Al planificar los eventos en un pipeline, el objetivo es obtener la menor latencia promedio entre iniciaciones sin ocasionar colisiones.

- Si la tabla de reservación tiene *n* columnas, la *máxima latencia prohibida* será $m \leq n - 1$

    - Vector de colisión
    - Diagrama de estado
    - Ciclos simples
    - Ciclo greedy

Vector de colisión

- Vector binario C = (C_m C_{m-1} … C_2 C_1) tal que:
  - C_i = 1 si *i* es una latencia prohibida
  - C_i = 0 si *i* es una latencia permitida

	1	2	3	4	5	6	7	8
S_1	X					X		X
S_2		X		X				
S_3			X		X		X	

Latencias prohibidas: 7,5,4,2

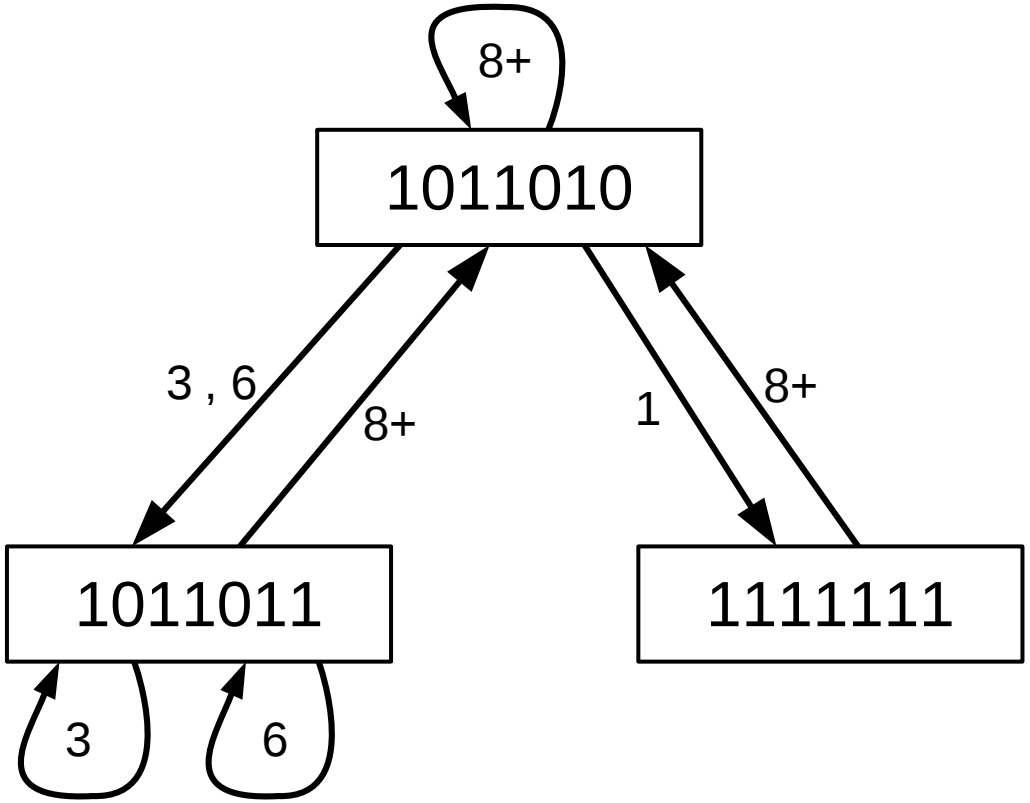
C = (1011010)

Diagrama de estados

- A partir del vector de colisión se construye un diagrama de estados que especifica las latencias permitidas luego de cada iniciación.
- El vector de colisión C representa el estado inicial del pipeline.

- Sea *p* una latencia permitida (*p*<*m*) en el tiempo *t* y C_P igual a C desplazado *p* lugares a derecha (ingresando 0 por la izquierda)
- El próximo estado del pipeline en el tiempo *t*+*p* será C or C_P

Diagrama de estados



Latencia 1:
1011010
or 0101101
1111111

Latencia 3:
1011010
or 0001011
1011011

Latencia 6:
1011010
or 0000001
1011011

Latencia 3:
1011010
or 0001011
1011011

Latencia 6:
1011010
or 0000001
1011011

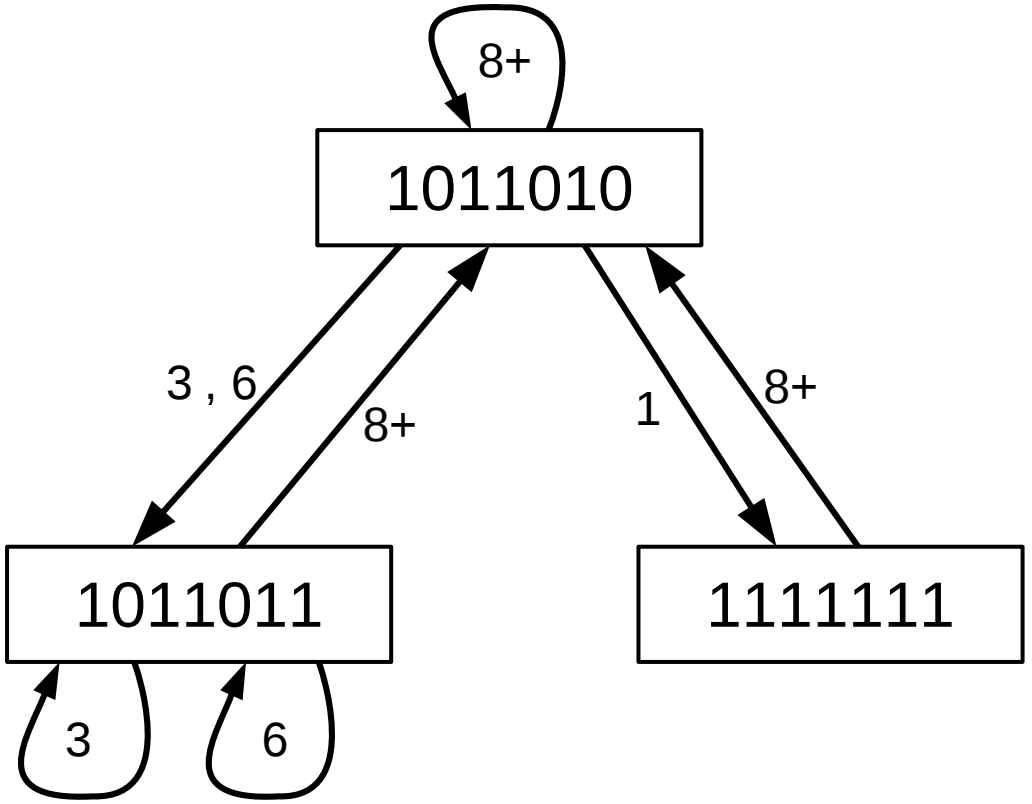
Ciclos

- A partir del diagrama se pueden detectar los ciclos de latencia.
- Hay infinitos ciclos de latencia.

- Ciclos simples:
  - Ciclos en los que no se repiten estados.
- Ciclos *greedy*:
  - Ciclos simples en los cuales la transición entre estados es a través de la mínima latencia.

Ciclos

- Posibles ciclos
  - {1,8}, {3}, , {6}, {3,8}
  - {3,6,3}, {1,8,6,8}
- Ciclos simples
  - {3}, {6}, {8}
  - {1,8}, {3,8}, {6,8}
- Ciclos *greedy*
  - {1,8} y {3}



Planificación sin colisiones

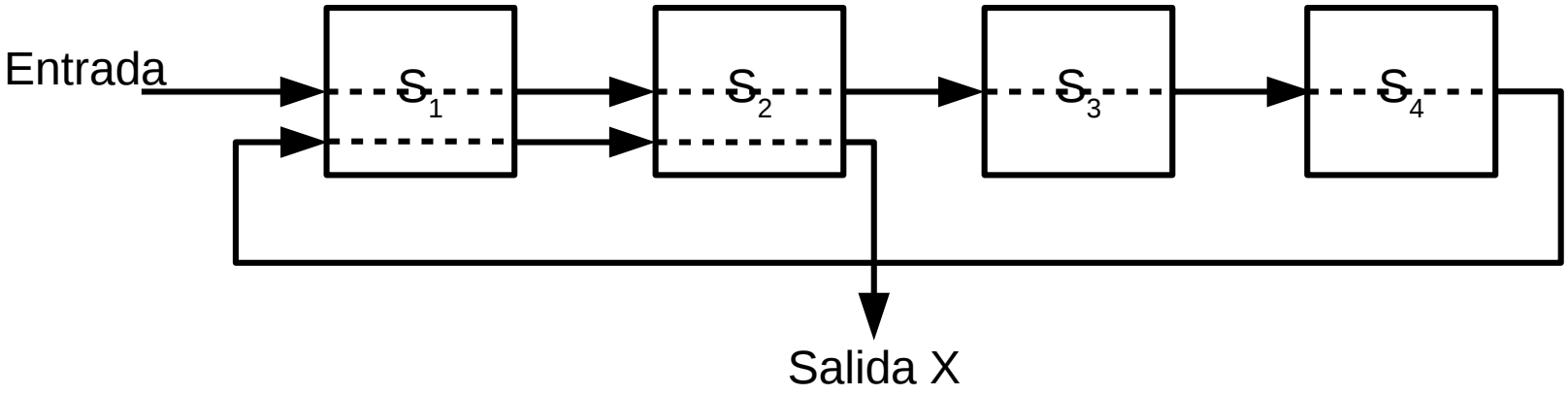
1) Identificar el vector de colisiones
2) Generar el diagrama de estados
3) Detectar ciclos *greedy*
4) El ciclo *greedy* de mínima latencia promedio determina la secuencia de iniciaciones del pipeline.

Mínima latencia promedio

- Está demostrado que la mínima latencia promedio está acotada por:
  - Cota inferior: máximo número de marcas en cualquier fila de la tabla de reservación.
  - Cota superior: mínima latencia promedio de los ciclos *greedy*.

    La cota superior para la latencia promedio de los ciclos *greedy* es uno más de la cantidad de 1s en vector de colisión inicial.
- La mínima latencia promedio obtenida se puede optimizar aún más agregando etapas de delay al pipeline original.

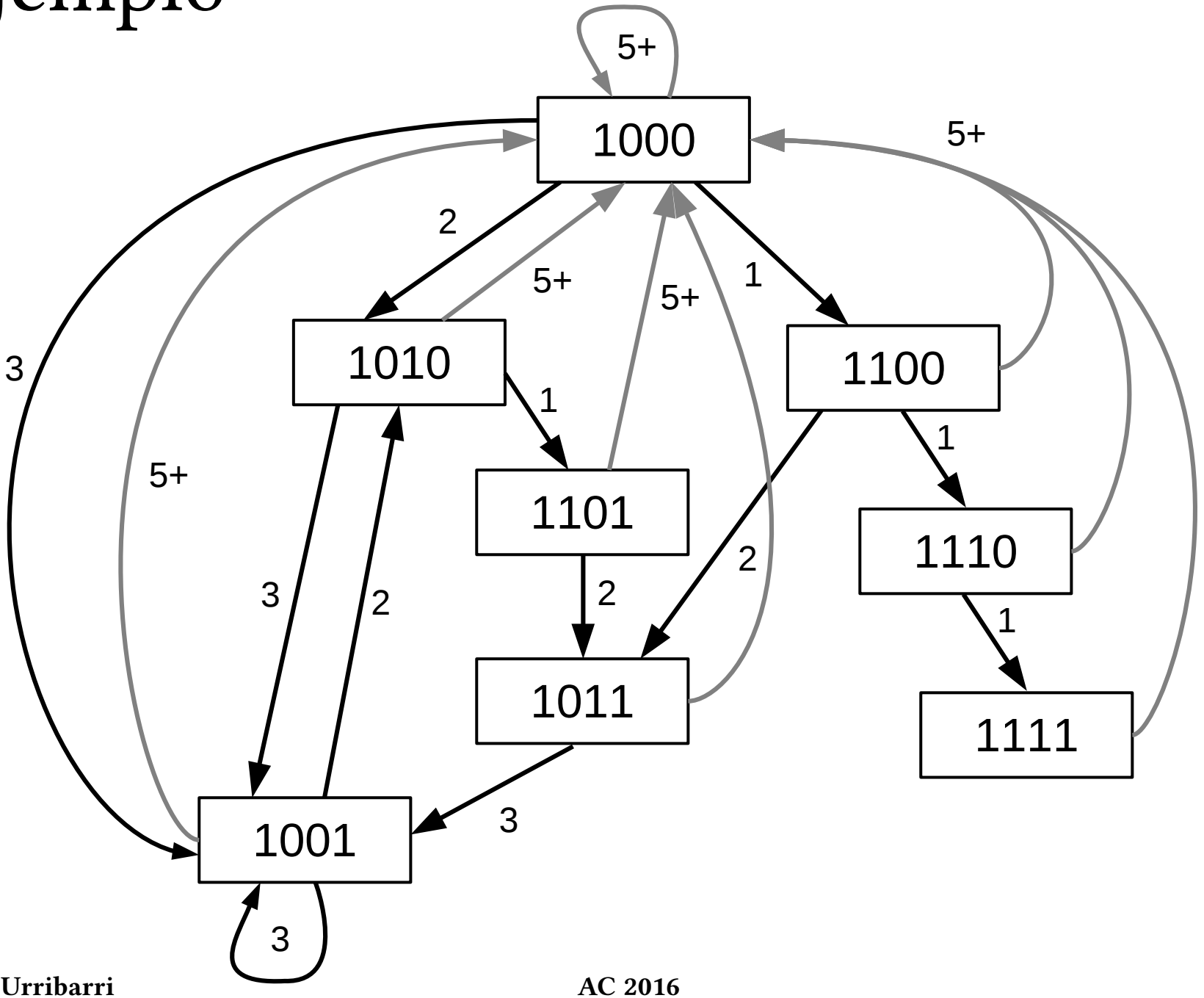
Ejemplo



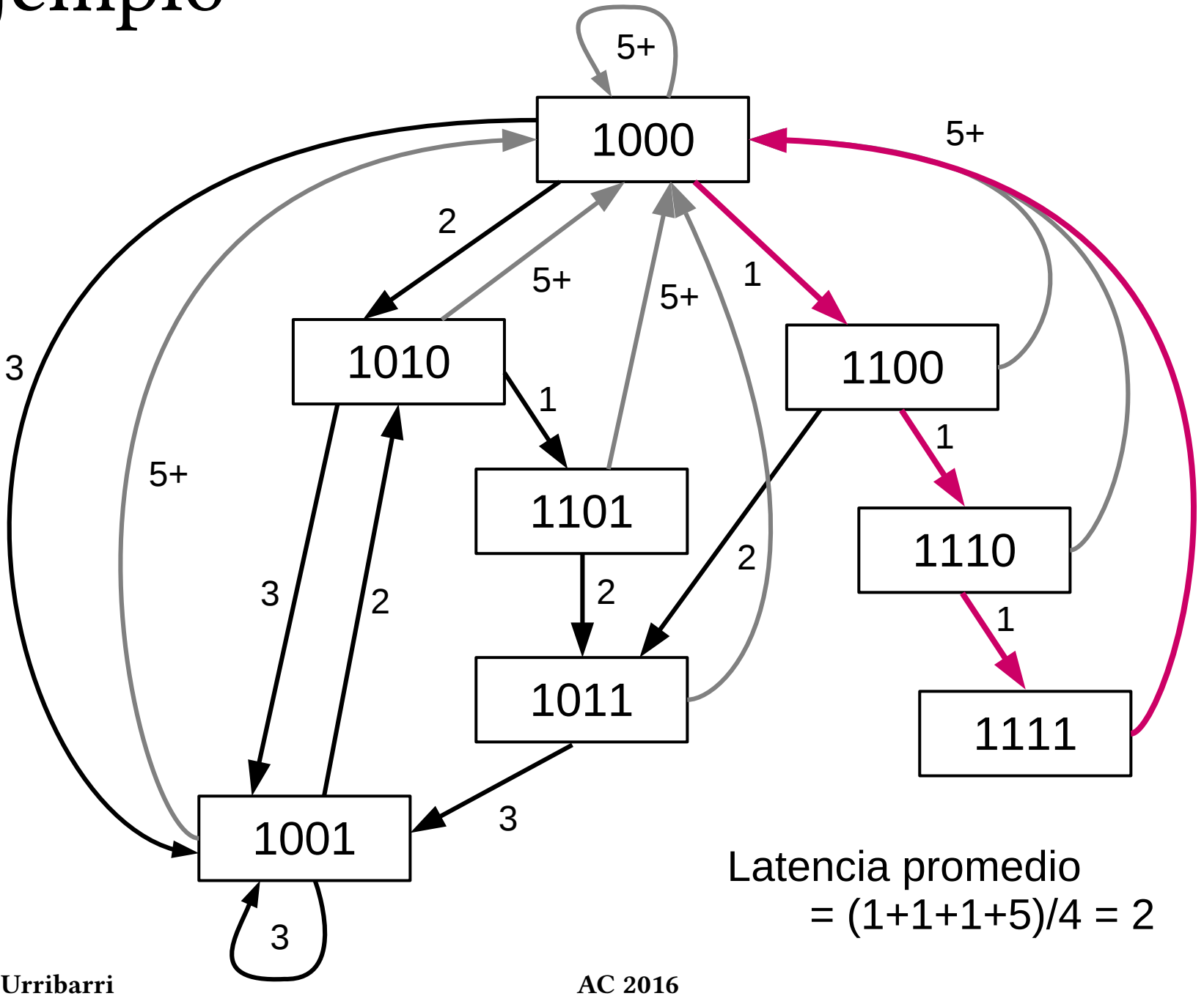
	1	2	3	4	5	6
S_1	X				X	
S_2		X				X
S_3			X			
S_4				X		

C = (1000)

Ejemplo

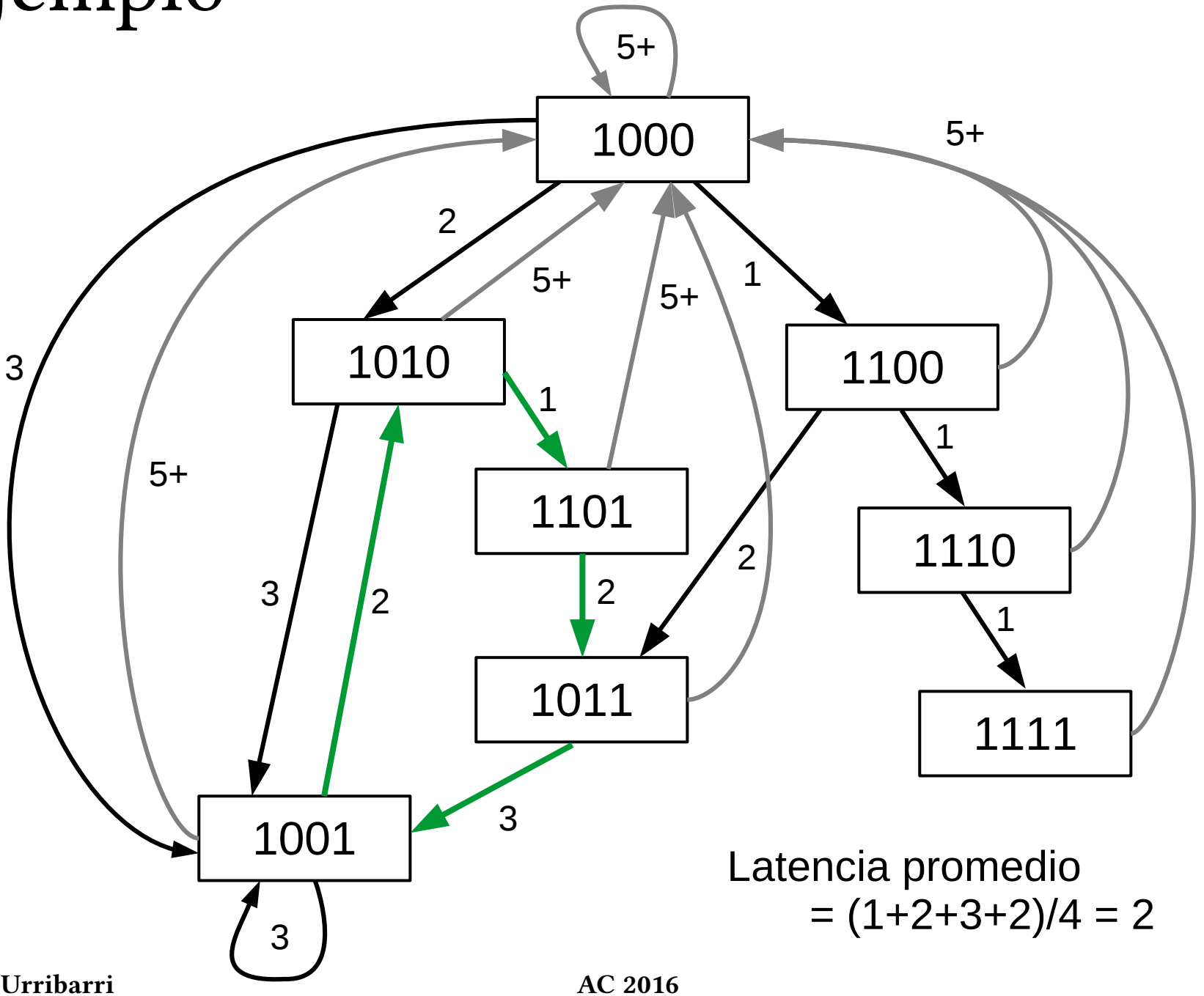


Ejemplo



Latencia promedio
= (1+1+1+5)/4 = 2

Ejemplo



Latencia promedio
= (1+2+3+2)/4 = 2

Ejemplo

- Cotas para la latencia promedio:
  - Cota inferior: máximo número de marcas en cualquier fila de la tabla de reservación → 2
  - Cota superior: uno más de la cantidad de 1s en vector de colisión inicial → 2

- Ciclo 1: Latencia promedio 2
- Ciclo 2: Latencia promedio 2

Bibliografía

- *Capítulo 3. Computer Architecture and Parallel Processing*. K. Wang y F. Briggs. McGraw-Hill, 1984.
- *Capítulo 6. Advance computer architecture. Parallelism. Scalability. Programmability*. Kai Hwang y Naresh Jotwani. McGraw-Hill, 2010.