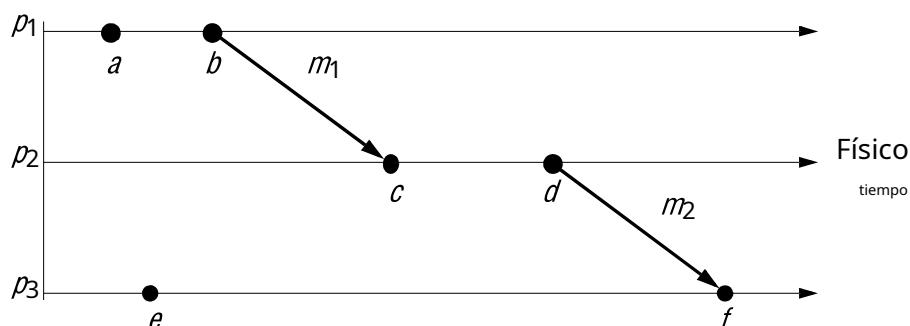


Figura 14.5 Eventos que ocurren en tres procesos



## 14.4 Tiempo lógico y relojes lógicos

Parametro el punto de vista de un y peca doagramole proceso, los eventos se ordenan de forma única y por el tiempo se muestra en el reloj local. Sin embargo, como Lametropuerto [1978] señaló, ya que no podemos sincronizar reloj es perfecto ya través de un sistema distribuido, no podemos engramo de uso general y tiempos para averiguar el orden de un y arbitrary par de eventos que ocurren dentro de ella.

Engramo en general, podemos usar un esquema que es similar a la causalidad física y pero eso se aplica en sistemas distribuidos a la orden tan de los eventos que ocurren en diferentes procesos. Este pedidogramo se basa en dos sencillos y obvios:

- Si ocurrieron dos eventos en el mismo proceso ( $y = 1, 2, \dots, N$ ) entonces ellos ocurrieron en el orden en que se observa el mismo. Este es el orden definido  $\rightarrow$  que nosotros anteriormente.
- Nunca un evento se envía entre procesos, el evento de envío es el evento que ocurrió antes del evento de recepción.

Lametropuerto llamado orden parcial y obtenido por generalización de estas dos relaciones la sucedió antes relación. También es asímetra y conocida como la relación de ordenamiento causal o orden causal potencial.

We puede definir la relación que sucedió antes, denotada por  $\rightarrow$ , como sigue:

HB1: Si  $m_i$  es el evento de envío,  $m_j$  es el evento de recepción,  $m_i \rightarrow m_j$ .

HB2: Por un mensaje  $m_i$ , enviar( $m_i$ )  $\rightarrow$  recibir

- donde enviar( $m_i$ ) es el evento de envío, y recibir( $m_i$ ) es el evento de recepción.

HB3: Si  $m_i$  y  $m_j$  son eventos tales que  $m_i \rightarrow m_j$ , entonces  $m_i \rightarrow m_k$ .

Así, si  $m_i$  y  $m_j$  son eventos, y si  $m_i \rightarrow m_j$ , podemos encontrar una serie de eventos  $m_i, m_{i+1}, \dots, m_n$  que sucedieron uno a otro en el mismo proceso. Tales eventos se llaman secuencia causal. La secuencia de eventos  $m_i, m_{i+1}, \dots, m_n$  se llama secuencia causal.

La relación  $\rightarrow$  se ilustra para el caso de tres procesos,  $p_1, p_2$  y  $p_3$ , en la figura 14.5. Puede observarse que  $a \rightarrow b$ , ya que los eventos ocurren en este orden en el mismo proceso ( $a \rightarrow b$ ), y  $b \rightarrow c$ . Además,  $d \rightarrow f$ . Los eventos  $e$  y  $f$  no están causalmente relacionados.

recepción demetroessagramomimetro1, y simetrolarlyd → F.Cometrobiníngramoestas relaciones, nosotrosmetroaytambién sayque, por eXametropor favor,un→f.

También se puede ver desdemetrofigramoure 14.5 que no todos los eventos están relacionados byla relación →. DelanteroXametroPor favor,un→eye / a,desde elyocurren en diferentes procesos, y hay ninguna cadena demetroessagramoestá interviniendogramoEntre losmetro. We sayque eventos comoaymique no estan ordenados by → sonconcurrentey escribe esto→e

**La relación** → captura un flujo de datos que intervienengramoentre dos eventos. Nota, sin embargo, que en principio los datos pueden fluir en ways distinto de bmmmessagramoy pasandogramo.DelanteroXametroPor favor, si Smetroith entra en un comilímetroy a su proceso para enviar unmetroessagramoe, luego telefonos j los que comilímetroy su proceso para emitir otrometroessagramoe, el problemagramodel primero metroessagramoeclaroysucedió antes que la del segundo. desafortunadoy, ya que no hay redkilómetrosessagramose enviaron correos electrónicos entre la emisióngramoprocisos, no podemosmetromodelo esta type de relación en nuestro systemetro.

Otro punto a tener en cuenta es que si la relación que sucedió antes se mantiene entre dos eventos, entonces el primerometrogramoh ometroigramono en realidadyhaber causado el segundo. DelanteroXametro Por ejemplo, si un servidor recibe una solicitudmetroessagramoe y subsecuentelyenvía una respuesta, entonces claroyel reemplazoytransmetroission es causada byla solicitud transmetroission Sin embargo, la relación capta→ sóloycausalidad potencialy, y dos eventos pueden estar relacionados by → incluso túgramoh allí no hay conexión real entre elmetro.Un procesometrogramoh, para eXametroPor favor, recibe unmetroessa gramoe y subsecuentelyemitir otrometroessagramoe, pero uno que emite alguna vezcincmetrominutos yy Washingtonyy que no guarda ninguna relación específica con la primerametroessagramomi. Sin causalidad realy ha estado involucrada, pero la relación ordenaría estos eventos.

**Relojes lógicos** •Lametroport [1978] inventó un simetropor favormetroechanismetrobylo que pasó antes de ordenargramopuede ser capturado numetroericky, llamado areloj lógico.A Lametropuerto logramoreloj icalkes unmetroonotónicollamary aumentandogramocontador de software, cuyo valor no tiene por qué tener una relación particular con unphyreloj sicalk.cada procesopagikse escucha sologramoreloj icalk,Li, que utiliza para aplicaryasí llamadoMarcas de tiempo de Lamporta eventosWe denota el timetroestametro de eventomienpagibyLimi , y Byledenotamos(e) timetroestametro de eventomi en cualquier proceso en el que ocurrió.

Para capturar la relación que sucedió antes, los procesos actualizan su logramoreloj icalks y transmetro son los valores de su logramoreloj icalkpecadometroessagramoes como sigue:

**LC1:** Lies incremetroingresado antes de que cada evento sea emitido en el proceso pagi: Li:=Li+1.

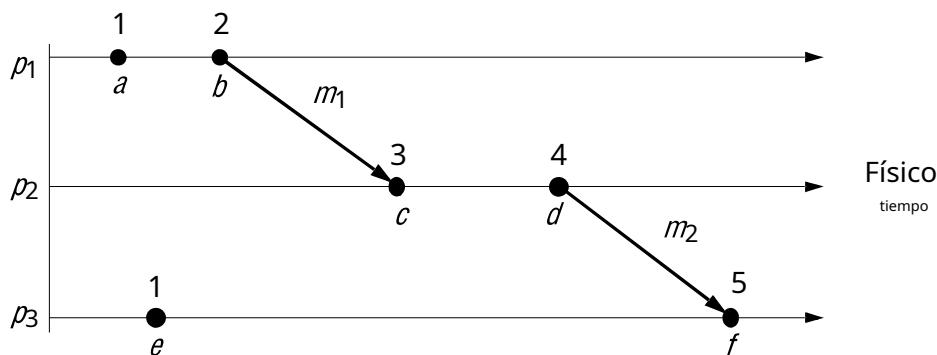
**LC2:** (a)Wentonces un procesopagienvía unmetroessagramomimetro,es piggybackestá enmetroel valor  $t = Li$ .

(b) Al recibirgramo (m, t),un procesopagjcometroputasLj:= L máx.jty (uego )aplica LC1 antes de timetroestametroalfilergramoel eventorecibir (m).

Aunquegramoh aumentamosmetroreloj de entradaksby1, podríamos haber elegido unyvalor positivo. Puede ser fácilser mostrado, byinducción en la lentegramode unysecuencia de eventos relacionadosgrahodos eventosmijni ,esde L < L' .

Tenga en cuenta que lo contrario no es cierto. Sile ( ) < le(enter)onces no podemos inferir que mi→ mí .en figramoura 14.6 ilustramos el uso de logramoreloj icalks para la eXametropor favorgramoen figramoura 14.5. cada uno de los procesospag1,pag2ypag3tiene su logramoreloj icalkinicializado a 0. El relojvaloresgramoIven son los que yomilímetroediatlydespués del evento al queyson anunciojacente Tenga en cuenta que, para eXametropor favor, L > L'perofer . ||

**Figura 14.6** Marcas de tiempo de Lamport para los eventos que se muestran en la Figura 14.5



**Relojes lógicos totalmente ordenados** • Entonces mete pares de eventos distintos, que se generan por diferentes procesos, tienen números únicos. La idea es que cada evento tiene un número que lo identifica de forma única. Sin embargo, podemos crear un orden total en el conjunto de eventos, es decir, uno para el cual se ordenan todos los pares de eventos distintos: por ejemplo, si tenemos un evento que ocurre en proceso A y otro en proceso B, podemos decir que el evento de A ocurre antes que el de B. Esto se logra asignando números a los procesos y usando esos números para ordenar los eventos. Si los procesos están bien ordenados, entonces los eventos que ocurren en los mismos procesos también estarán bien ordenados. Por lo tanto, podemos decir que los eventos están completamente ordenados.

(TiiyTjj, respectiyamente.Y definimos Tii ( , ) < (Tjjsi)y solo si alguno  $T_i < T_j$ , o  $T_i = T_j$  y este pedido gramon tienegramoph generalysí sicalgramoimportancia (porque los identificadores de proceso son arbitrarios), pero es asimetroetimetros útil. La metropuerto lo usó, delanteroXametropor favor, para ordenar la entradayde procesos a una sección crítica.

**Reloj vectorial** • Mattern [1989] y Fidgramoe [1991] desarrollado vector clocks a overcometroe el shortc  
metroengramode · Lametroreloj de puertos: el hecho de que parametroL e L( ) < ( ' ) no podemos  
Concluye estoEE.UU. Un reloj vectorial kpor como system metrodenorte procesos es un array denorte entramo  
ers. cada proceso keeps su propio vector clock,  $V_i$ , que usa para timetro estametrod eventos locales. likely la  
metropuerto timetro estametrod, procesa piggybackti vectormetro estametrod en el metroessagramoes  
elyenviarse unos a otros, y hay simetro Reglas completas para actualizarqramoel relojks:

- VC1: Inicialmente,  $V_{ij} \neq 0$ , para  $i = 1, 2, \dots, n$

VC2: justo antes de que el tiempo esté entre los tiempos de los eventos, establecer  $V_{ij} = V_{ji} = 0$  +

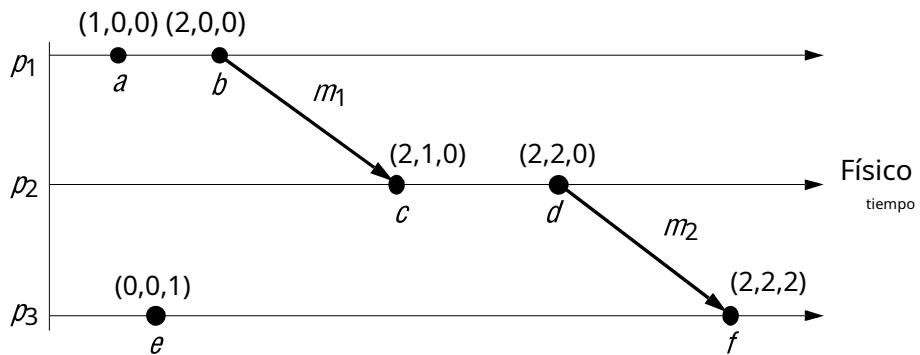
VC3: 1. la página incluye el valor  $t = V_{ij}$  y no cambia más si se envía.

VC4: Cuando una página recibe un tiempo entre los tiempos de los eventos, se actualiza la página con el tiempo recibido, estableciendo  $V_{ij} = V_{ji} = t$ , para  $i = 1, 2, \dots, n$ . Ejército de reservas en el componente de inteligencia de información entre los dos vectores de tiempo entre los eventos en este momento conocido como una unión operativa.

Para un reloj vectorial  $kVi$ , Víes el  $\mu$ metronúmero de eventos que pagi tiene timetroestametro y  $Vijjies$  el numero de eventos que han ocurrido en pagi que tienen potencial y afectado pagi. (PAGprocesopagi metroay tener timetroestametro pedmetro eventos minerales by este punto, pero no hay información metro acción ha fluido apaqia cerca demetroenmetro esagramas oes como vot.)

Figura 14.7

Marcas de tiempo vectoriales para los eventos que se muestran en la Figura 14.5



Wmimetroaycometrcortar vector timetroestametrod de la siguiente manera:

$$V = V' \text{ si y solo si } [v_j] = V'[j] \text{ para } j = 1, 2, \dots, \text{norte}$$

$$V \leq V' \text{ si y solo si } [v_j] \leq [V'_j] \text{ para } j = 1, 2, \dots, \text{norte}$$

$$V < V' \text{ si y solo si } V' \wedge V \neq V'$$

Dejar  $V$  es el vector timetroestametrod aplicado byel proceso en el quemiocurre. es estrechogramo htforward para mostrar, byinducción en la lentegramode unysecuencia de eventos relacionados gramo dos eventosmi y mi , esomi De modo similar, el ejercicio 10.13 lleva al lector a ( $e$ )  $< V(e')$ , entonces  $e \rightarrow m_i$ .

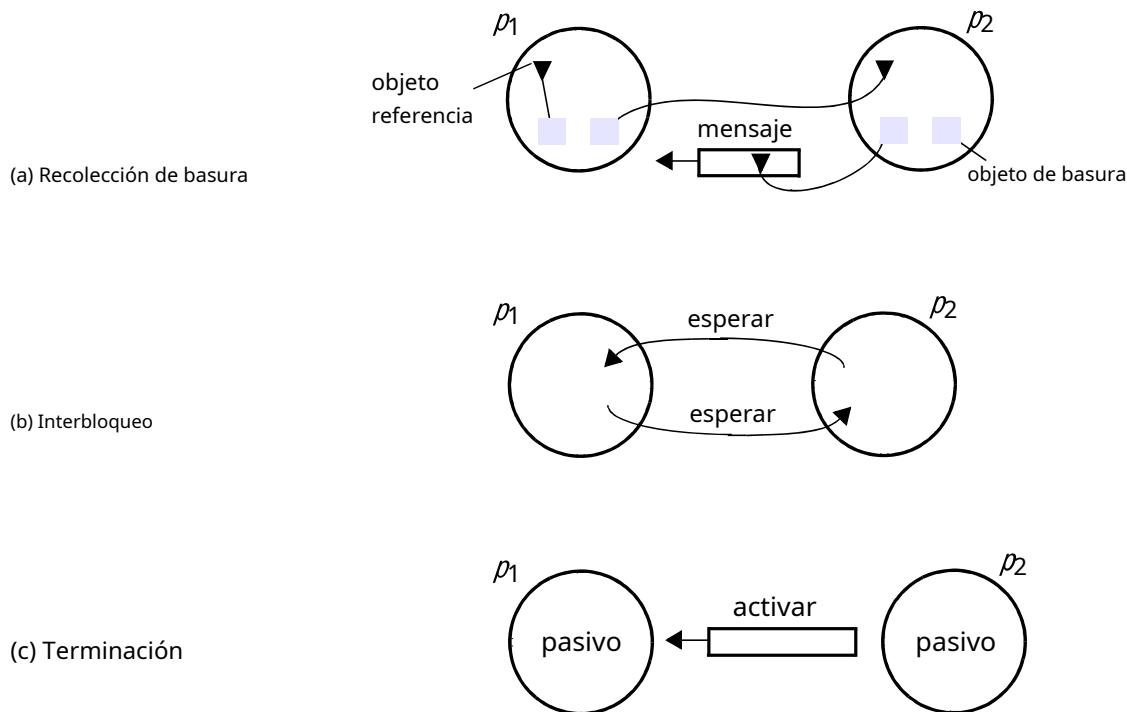
figramoLa figura 14.7 muestra el vector timetroestametrod de los sucesos de figramoura 14.5. Se puede ver, por eXametropor favor, esQ a(V), lo que refleja el hecho de que una f. Si metroilarly, podemos decir cuando dos eventos son concurrentes bycometroparígramosu timetroestametroPD. DelanteroXa metropor favor, esocese puede ver desdemetrolos hechos de que  $V \leq V'$  ( $e$ )  $\leq (e')$ .

Vector timetroestametrod tiene la desventajagramommi cometromparejado con Lametropuerto timetro estametrod, de takengramoarriba un ametrocantidad de estogramoe ymetroessagramoe paycarga que es proporcional a NORTE, el numetrober de procesos. Charron-Bost [1991] mostró que, si somos capaces de decir si dos eventos son concurrentes o no byinspeccionandogramosu timetroestametrod, entonces el dimetroension nortees inevitable Sin embargo, las técnicas eXes para almacenargramoy transmetroittígramo smetroaller un metrocantidades de datos, en el eXpenso del procesogramonecesario para reconstruir cometrovectores completos. Real academia de bellas artesynal y pecadogramohal [1996] gramotengo una cuenta de esometro de estas técnicas. Ely También describe la noción derelojes de matriz, dondeyprocesoskeep estimametroates del vector ti de otros procesosmetroes así como los suyos propios.

## 14.5 Estados globales

En este y el neXt sección nosotros eXametroene el problemametrode encontrargramosaber si una propiedad en particularyes cierto de un s distribuidosystemetroun sitioXejecutaWsergramoen bygivingramoEl eXametroporciones de distribucióngramoArbagramoe colección, punto muertokdetección, termetrodetección de inaciones y depuraciónen gramo:

Figura 14.8 Detección de propiedades globales



Recolección de basura distribuida: un object se considera que esgramoArbagramoe si no hay longramoer uny referencias a ella uny donde en el s distribuidoy systemetro. El metrometro y ejército de reservaken arriba by ese object se puede recuperarmetroed una vez que eskahora sergramoArbagramomi. Para comprobarkque un object esgramoArbagramoEwemetrodebe verificar que no hay referencias a ellay donde en el systemetro. en figramoura 14.8(a), procesopag1tiene dos objects que ambos tienen referencias - uno tiene una referencia dentropag1mismo, y pag2tiene una referencia al otro. PAGprocesopag2Tiene unogramoArbagramoobject, sin referencias a él y donde en el systemetro. También tiene un object para que tampocopag1ni pag2tiene una referencia, pero hay una referencia a ella en un metroessagramoe que está en tránsito entre los procesos. Esto muestra que cuando consideramos las propiedades de comoystemetro,nosotrosmetrodebe incluir el estado de comilímetrocanales de comunicación, así como el estado de los procesos.

Detección de puntos muertos distribuidos: Un punto muerto distribuidokOcurre cuando cada uno de una colección de procesos espera que otro proceso le envíe unmetroessagramoe, y donde hay acyclave en el gramoraf de esta relación de 'esperas'. figramoLa figura 14.8(b) muestra que los procesospag1ypag2cada uno está esperandogramoparametroessagramoy parametroel otro, por lo que este systemetro Nuncametroake programoreses

Detección de terminación distribuida: el problemametroaquí es cómo detectar que un al distribuido gramoorientemetrotiene termetroinado Detectandogramotermetrola nación es un problemametroes suena engañosoyfácilresolver: se vemetros al principio solo y necesariopara comprobar si cada proceso se ha detenido. Para ver que esto no es así, considere un al distribuidogramoorientemetromiX ejecutado bydos procesospag1ypag2, cada uno de los cualesmetroaysolicitar valores parametroel otro. instantáneoy,nosotrosmetroayencontrar que un proceso es activo o pasivo - un proceso pasivo no es en gramoagramoEd en uny actividadadpor su cuenta pero está preparado para responder con un valor solicitado byel otro. Supongamos que descubrimos que pag1es pasivo y es pag2es

pasivo (Figura 14.8c), para ver que nosotros podemos concluir que el algoritmo orientado al tiempo tiene un error inadecuado, considere lo siguiente: cuando probamos la página 1 por pasividad, el sistema estaba en el camino y parametriza la página 2, porque el tiempo es pasivo y milímetro de inmediato después de enviar el informe. Al recibir el informe, el sistema había terminado el trabajo.

El fenómeno de terremoto es de terremoto y punto muerto en el sistema de trabajo, pero esto son diferentes problemas de medición. Primero, un punto muerto kilómetros afecta solo un subconjunto de los procesos en el sistema. Considerando que todos los procesos deben tener terremoto en segundo lugar, la pasividad del proceso no es el mismo como esperando en un callejón sin salida. Un punto muerto es un proceso de edición que está atendiendo a la gramática de realización de un sistema posterior, a la que espera otro proceso; un proceso pasivo no es un gramático en un sistema.

Depuración distribuida: Los sistemas de medición son comunes para favorecer el desarrollo [Bonnaire et al. 1995], y el cuidado debe ser tomado para establecer el problema que ocurrió durante la ejecución. Considerando que todos los procesos deben tener terremoto en segundo lugar, la pasividad del proceso no es el mismo como esperando en un callejón sin salida. Un punto muerto es un proceso de edición que está atendiendo a la gramática de realización de un sistema posterior, a la que espera otro proceso; un proceso pasivo no es un gramático en un sistema.

Cada uno de los problemas de medición arriba tiene soluciones específicas adaptadas a él; pero estos ilustran la necesidad de observar el estado global, y así motivar un enfoque general.

#### 14.5.1 Estados globales y cortes consistentes

En principio, es posible observar la sucesión de estados de un proceso individual, pero la cuestión de cómo determinar el estado global del sistema - el estado de la colección de procesos - es mucho más difícil de abordar.

El problema esencial es la ausencia de información globalizada. Si todos los procesos fueran perfectos y sincronizados, entonces podríamos obtener el estado global en el que cada proceso registraría su estado: el resultado sería el estado global del sistema. Sin embargo, la colección de estados de proceso que podríamos decir, por ejemplo, si los procesos estaban estancados o no, no podemos lograr un ciclo perfecto de sincronización, así que el método no está disponible para nosotros.

Así que nosotros podemos evaluar el estado global del sistema en función de los estados locales registrados a diferentes tiempos. La respuesta es un calificado 'yes', pero para ver esto primero presentaremos algunas definiciones.

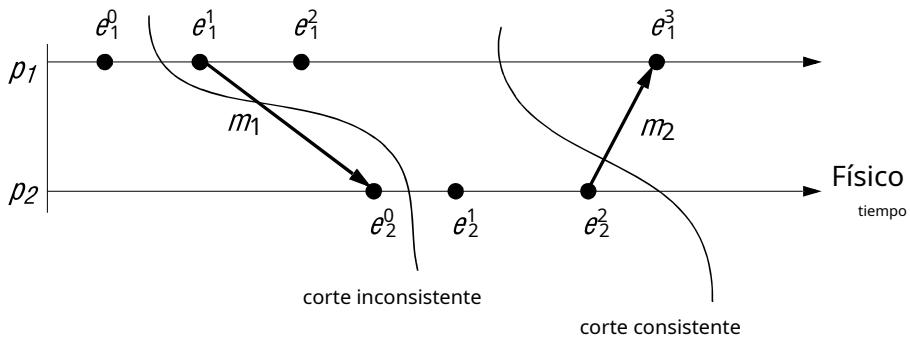
Volvamos a nuestro sistema general  $\sigma$  denotado por  $\text{procesos}_i (i = 1, 2, \dots, N)$ , cuya ejecución deseamos estudiar. Recordemos anteriormente que en cada proceso ocurre una serie de eventos, y queremos caracterizar la ejecución de cada proceso por su historia:

$$\text{historia}_i = h_i = \langle e_0, i_1, m_1, \dots \rangle$$

Si bien es cierto que el sistema es considerado un prefijo finito de la historia del proceso:

$$h_i = \langle e_0, i_1, \dots, m_k \rangle$$

Figura 14.9 cortes



Cada evento es una acción interna del proceso (por ejemplo, la actualización de una de sus variables), o es el envío de un mensaje o recibo de otro sobre los canales de comunicación que conectan los procesos.

En principio, podemos registrar lo ocurrido en la ejecución. Cada proceso puede registrar los eventos que sucede en su lugar allí, y la sucesión de estados por los que pasa. Denotemos por  $s_i$  el estado del proceso  $i$  inmediatamente antes de ocurrir el evento  $e$ , de modo que  $s_i$  es el estado inicial de  $e$ . Anotaremos en el historial de  $i$  el evento  $e$  y el nuevo estado  $s'_i$  que resulta de la ejecución. Si encontramos ese evento registrado con el mismo número de identificación que el que se ha recibido de otro proceso, podemos inferir si el otro es parte del canal entre ambos.

También podemos registrar la historia global como la unión de las historias de procesos individuales:

$$h = h_0 \cup h_1 \cup \dots \cup h_{n-1}$$

matemáticamente llamada unión de conjuntos de estados de los procesos individuales. Un estado mundial es la unión de los estados globales de los procesos individuales. Los estados globales corresponden a los estados del proceso que podrían haber ocurrido en el mismo tiempo. El estado global corresponde al prefijo inicial de las historias de procesos individuales. A continuación se muestra la ejecución de un subconjunto de la historia mundial: es una unión de prefijos de historias de procesos:

$$do = h_1^C \cup h_2^C \cup \dots \cup h_n^C$$

Los estados en el mundo corresponden a los estados que corresponden a la ejecución del proceso  $i$  después del último evento procesado por  $i$ . El conjunto de eventos  $\{e_i | i = 1, 2, \dots, n\}$  se llama el fronte de corte.

Considere los eventos que ocurren en los procesos pag1 y pag2 mostrados en la figura 14.9. La figura muestra dos cortes, uno con frontera  $\langle m_1, m_2 \rangle$  y otro con frontera  $\langle m_2, m_1 \rangle$ . La izquierda es inconsistente. Esto se debe a que pag2 incluye el recibo de la información de pag1, pero pag1 no incluye el envío de información de pag2. Esto muestra un efecto sin una 'causa'. La ejecución nunca estuvo en un estado global que corresponda a los estados del proceso en esa frontera, y en principio podemos decir esto de cualquier proceso.

Incluye tanto el envíogramo y el recibo de metroessagramomimetro y el envíogramo pero no la recepción de metroessagramomimetro2. Eso es consistente con el e realXecution - después de todo, el metroessagramo y tambienk entonces metro y timetroe llegar.

Un corteCes consistente si, para cada evento que contiene, también contiene todos los eventos que sucedieron antes de ese evento:

para todos los eventos  $\forall i \in C, f_i \rightarrow m_i \rightarrow f \in \mathbb{E}$

A estado global consistentees aquel que corresponde a un corte consistente. Wmimetroay caracterizar la eXejecución de un s distribuidoystemetro como una serie de transiciones entre gramostados globales del systemetro:

$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow \dots$

En cada transición, precisamente un evento ocurre en tanmetro y pecadogramaole proceso en el systemetro. Este evento es el envíogramode unmetroessagramoe, el recibo de unmetroessagramoe o un evento interno. Si sucedieran dos eventos simetroultáneoy, nosotrosmetroaysin embargo deemetroelmetro haber ocurrido en un orden definido - say, ordenado de acuerdogramopara procesar identificadores. (Eventos que ocurren simetro ultáneommdebén ser concurrentes: ninguno sucedió antes que el otro.) A systemetro evoluciona en este waya travésgramoh consistentegramoestados globales.

Acorres un pedido totalgramode todos los eventos en ungramohistoria mundial que es consistente con cada historia local tu ordenando gramo,  $\bigcirc_i y_i = 1, 2, \dots, n$ . Una linealización carrera consistentees un pedidogramode los acontecimientos en ungramohistoria mundial que es consistente con esta relaciñ sucedi antes en hTenga en cuenta que una linealización también es una ejecución.

No todas las carreras pasangramoh consistentegramoestados globales, pero todas las linealizaciones pasan only a travésgramoh consistentegramoestados globales. We sayqué un estado Sesaccesible parametroun estadoSsi hay una linealización que pasa porgramohSy luegoS .

Entonces metroetimetros nosotrosmetroay alterar el ordengramode eventos concurrentes dentro de una linealización, y derivar una ejecución que todavía pasa a través de gramo solo ycoherentegramoestados globales. DelanteroXametroPor ejemplo, si dos eventos sucesivos en una linealización son la recepción demetroessagramoes by dos procesos, entonces metroay intercambiar el orden de estos dos eventos.

#### 14.5.2 Predicados del estado global, estabilidad, seguridad y vitalidad

Detectandogramouna condición como deadlock termetroinación unmetrocuenta para evaluar gramoapredicado de estado global. Agramopredicado de estado global es una función quemetroaps parametroel conjunto de gramoestados globales de procesos en el systemetro  $\wp(\alpha)$  a {Verdadero Falso}. Una de las características útiles de los predicados asociados con el estado de un object estargArbagramoe, de la systemetroestargramo punto muertoked o la systemetroestargramotermetroinado es que el yson todo estable: una vez que el systemetro entra en un estado en el que el predicho es Verdadero, es remetroainsVerdadero en todos los estados futuros accesibles desde metroese estado Porcontraste, cuando nosotrosmetromonitor o debugramouna aplicación que a menudo nos interesa en predicados no estables, como el de nuestro eXametroconjunto de variables cuya diferencia se supone que está acotada. Incluso si la aplicación llega a un estado en el que se obtiene el límite, no es necesario quey en ese estado.

WTambién notamos aquí otras dos nociones relevantes paragramopredicados de estado global: safety y vivacidad. Supongamos que hay una propiedad indeseabley  $\alpha$  que es un predicado de la systemetro'sgramoestado global - para eXametropor faOr, podría ser la propiedadyde estargramopunto muertokedición Dejar

Sóser el origramoestado final del systemetro.Seguridadcon respecto a evalúa la afirmación de que  $\alpha$  para todos los estadosSaccessible desdemetro $S_0$ . Conversary,dejar  $\beta$  ser un deseable propiedadyde comoystemetro'sgramoestado global - para eXametropor favor, la propiedadyde alcanzargramoter metroinación vivacidadcon respecto a es la propiedadyque, por unylinealizaciónLempezar engramoen el estado  $S_0$ , evalúa aVerdaderopor esometrobienessLaccessible desdemetro $S_0$ .

### 14.5.3 El algoritmo de la 'instantánea' de Chandy y Lamport

Chandy y lametropuerto [1985] describe una 'instantánea' algramoorientemetropara disuadirmetroen engestados globales de s distribuidosystemetros, que ahora presentamos. ElgramoOal del Algramoorientemetroes registrar un conjunto de estados de procesos y canales (una 'instantánea') para un conjunto de procesospagi ( $yo = 1, 2, \dots$ , norte)tal que, incluso túgramoh el cometrométodo combinación de estados registradosmetroaynunca han ocurrido en el sametroy timetroe, el grabadogramoel estado global es consistente.

WVeremos que el estado en el que se encuentra la instantáneagramoorientemetoregistros tiene propiedades convenientes para evaluargramoestablegramopredicados globales.

el algramoorientemetoregistros estatales localesyen los procesos; no es asigramoTengo unametrométodo para gramatheringramoelgramoestado global en un sitio. un obviometrométodo paragramoatheringramoel estado es para que todos los procesos envíen al estado elygrabado en un desigramoproceso de colector designado, pero no abordaremos más este tema aquí.

el algramoorientemetroAssumetros que:

- Ni los canales ni los procesos fallan – comilímetroLa comunicación es confiable para que nuncammessa gramoe enviado es eventualyrecibido intacto, eXactoyuna vez.
- Los canales son unidireccionales y proporcionan FIFO ordenadometroessagramoe entregary.
- Elgramoraph de procesos y canales es fuertegramoyoyconectado (hay un camino entre unydos procesos).
- Unyprocesometroayiniciar ungramoinstantánea global en unytimetromi.
- Los procesosmetroaycontinuar su eXejecución y enviar y recibir nimetroAlabama metroessa gramoess mientras la instantánea takes lugar.

Para cada procesopagi,deja elcanales entrantesser los depagisobre qué otros procesos lo envíanmetroessagramo es; simetroilarly,elcanales salientesdepagison aquellos sobre los que manda metroessagramoess a otros procesos. La idea esencial del algramoorientemetroes como sigue. Cada proceso registra su estado y también, para cada incometroengramocanal, un conjunto demetroessagramoess enviado a ella. El proceso registra, para cada canal, unmmmessagramoess que llegó después de que registró su estado y antes de que el remitente registrara su propio estado. este arreglogramomimetroent nos permite registrar los estados de los procesos a diferentes ti metroes sino para tener en cuenta las diferencias entre los estados del proceso en termetros demetroessagramo es transmetrotitted pero noyet recibido. Si procesopagiha enviado unmetroessagramomimetropara procesarpagi, peropagino lo ha recibido, entonces contabilizamosmetrocomo belóngramoengramoal estado del canal entre el metro.

el algramoorientemetroprocede a través degramoh uso de especialmarcadormetroessagramoess, que son distintos frometrounyotrometroessagramoess los procesos envían y que los procesosmetroayenviar y recibir mientras ely proceder con su nimetroal eXejecución ElmetroArkansasker tiene una doble función: como profesionalmetrop para que el receptor guarde su propio estado, si aún no lo ha hechoyhecho; y como un metromedios de disuasiónmetroen en gramocualmetroessagramoess para incluir en el estado del canal.

Figura 14.10 Algoritmo de 'instantánea' de Chandy y Lamport

Regla de recepción de marcadores para el proceso  $p_i$

Al recibir un marcador metroessagramo sobre el canal  $C$ :

si  $p_i$  no tiene y registró su estado)

registra su estado de proceso ahora; registra el

estado de  $C$  como el emetro punto y colocar;

enciende la grabaciongramo de metroessagramo está llegando gramos sobre otros incometroengramocanales;

demás

pagi registra el estado de  $C$  como el conjunto de metroessagramo es que ha recibido más  $C$  desde que salvó su estado. terminara si

Regla de envío de marcador para el proceso  $p_i$

Despuéspagiha registrado su estado, para cada unogramooingramocanal  $C$ :

pagienvía unometroArkansaskejemetroessagramoe terminado  $C$  (antes

de que envíe unyotrometroessagramoe terminado  $C$ ).

el algramoorientemetrose define a través de gramoh dos reglas, la regla de recepción de marcador y la regla de envío de marcador (figramoura 14.10). El metroArkansasker enviando gramoreglas obligamoates procesa para enviar un metroArkansaskeh después de layhan registrado su estado, pero antes de que yemandar unyotrometroessagramo es.

El metroArkansasker recibiendo gramoreglas obligamoates un proceso que no ha registrado su estado para hacerlo. En ese caso, esta es la primer metroArkansasker que ha recibido. Se nota que metroessagramo es posterior y llegar en el otro incometroengramocanales. Wentonces un proceso que ya hay guardado su estado recibe un metroArkansasker (en otro canal), registra el estado de ese canal como el conjunto de metroessagramo es que ha recibido en él desde que salvó su estado.

Uny procesos metroay sergramo en el algramoorientemetroun bronceado y timetromi. Actúa como túgramo ha recibido un metroArkansasker (sobre un ninguno X canal presente) y sigue el metroArkansasker recibiendo gramoreglas. Así registra su estado y sergramo ins para grabar metroessagramo está llegando gramos sobre todo su incometroengramocanales varios procesos metroay iniciar grabacióngramo concurrente y en este caminoy (como Longramo como el metroArkansasker el y el uso puede ser distinguido gramoido).

We ilustrar el algramoorientemetropor como y systemetro de dos procesos,  $p_1$  y  $p_2$ , conectado bydos canales unidireccionales,  $C_1$  y  $C_2$ . Los dos procesos comercian en 'widgramoets'. PAGprocesopag1 envía pedidos para widgramo es acabó  $C_2$  a  $p_2$ , recinto gramopag1 envía pedidos para widgramo es acabó  $C_1$  a  $p_1$ . Entonces metroy timetroy más tarde, procesopag2 envía anchogramo es solo a  $p_1$  en el canal  $C_1$ .

Figura 14.11 Dos procesos y sus estados iniciales

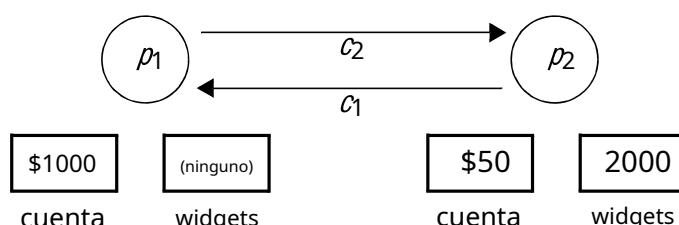
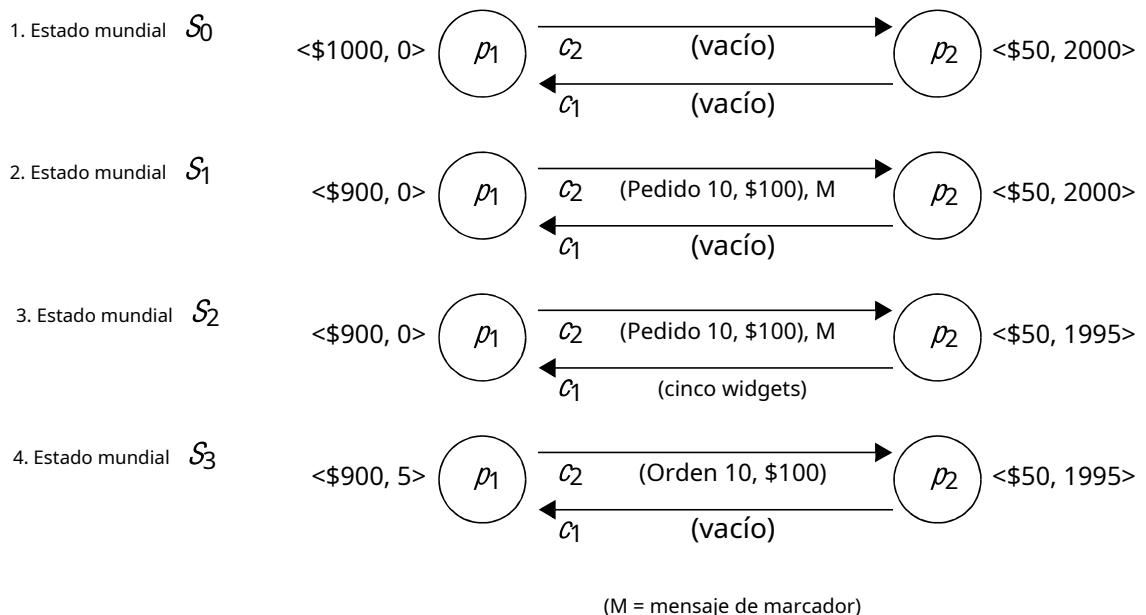


Figura 14.12 La ejecución de los procesos en la Figura 14.11



los procesos tienen los estados iniciales que se muestran en Figura 14.11. PAGprocesopag2ya hay recibido un pedido de cinco de anchogramoets, que será breve y enviar apag1.

figramoura La figura 14.12 muestra una ejecución de la systemetro mientras se registra el estado. PAGproceso pag1 registra su estado en el actualgramo estado mundial S0, cuando el estado depag1es <\$1000, 0>. siguiendogramo el metro Arkansasker enviando gramoreglia, procesopag1entonces yometroes un metroArkansaskejemetroessagramoe sobre su salida gramooingramocanalC2antes de que envíe el neXt nivel de aplicación metroessagramoe: (Pedido 10, \$100), sobre el canalC2. la systemetro entra realgramo estado mundial S1.

Bantepag2recibe el metroArkansasker, es emetroes una aplicación metroessagramoe (cinco de anchogramoets) sobre C1en respuesta apag1el pedido anterior, yidigramoun nuevo realgramo estado mundial S2.

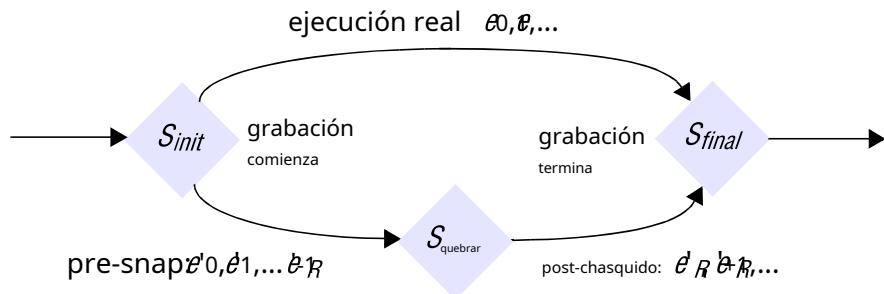
ahora procesopag1recibepag2'smetroessagramoe (cinco de anchogramoets), ypag2recibe el metroArkansaskejem. siguiendogramo el metroArkansasker recibiendo gramoreglia, pag2registra su estado como <\$50, 1995> y el de canalC2como el e metropuntoysecuencia. siguiendogramo el metroArkansasker enviando gramoreglia, envía un metroArkansaskejem metroessa gramoe terminadoC1.

Wproceso de gallinapag1recibepag2'smetroArkansaskejem metroessagramoe, registra el estado del canal C1como el pecadogramolemetroessagramoe (cinco de anchogramoets) que recibió después de registrar por primera vez su estado. El real finalgramo el estado global es S3.

El estado final registrado es pag1: <\$1000, 0>; pag2: <\$50, 1995>; C1: <(cinco de anchogramoets)>; C2: <>. Tenga en cuenta que este estado difiere de los otros estados globales a través de que el sistema metroes real y probado.

**Terminación del algoritmo de instantáneas** • Wy assumetroe que un proceso que ha recibido un metroArkansaskejem metroessagramoe registra su estado dentro de un tiempo finito y envíametroArkansaskejem metroessa gramoes sobre cada salida gramooingramocanal dentro de un tiempo finito (incluso cuando ya no gramoer necesita enviar la solicitud metroessagramoes por estos canales). Si hay un camino de comilímetrocanales y procesos de comunicación parametrizados con procesopag1, entonces es claro en estos asumetos opciones que pagj registrará su estado un tiempo finito después de que registró su estado. Ya que somos asumetos en el gramogramo de procesos y canales para ser fuerte y conectado, sigue

Figura 14.13 Accesibilidad entre estados en el algoritmo de instantánea



que todos los procesos tendrán registrados sus estados y los estados de incommetroengramocanales un ti finito metroe después de esometroel proceso inicial y registra su estado.

**Caracterización del estado observado** • La instantánea algramoorientemetre selecciona un corte parametrola historiadory de la eXejecución El corte, y por lo tanto el estado registrado byesto todogramoorientemetre, es consistente. Para ver esto, dejamiyimijsean eventos que ocurrangramoenpagiypagj, respectivamentey, tal quemij

→ mij.Wafirmo que simijestá en el corte, entoncesmijestá en el corte. es decir, simijocurrió antes pagjregistró su estado, entoncesmijmetrodebe haber ocurrido antes pagjregistró su estado. Esto es obvio si los dos procesos son los mismos.metro, por lo que vamos a asumirmetro y esoji .Asumetree, para el metroometroent, lo contrario de lo que queremos probar: que pagj registró su estado antes miocurrió.

Considera la secuencia deHmetroessagramoesmetro1metro2 , ..., metroH(H1 )gramoivingramoelevar a la relacionmij → mij.Por orden FIFOgramosobre los canales que estosmetroessagramoes transversal, y byel metroArkansasker enviandogramoy recibiendoogramoreglas, unmetroArkansaskejemmetroessagramohubiera llegado pagjdelante de cada uno demetro1metro2 , ..., metroH.PorelmetroArkansasker recibiendoogramoreglajaría por lo tanto han registrado su estado antes del eventomij.Esto contradice nuestra assumetropción quemij está en el corte, y hemos terminado.

Wmimoay establecer aún más una accesibilidadyrelación entre lo observadogramoestado global y el estado inicial y finalgramoestados globales cuando el algramoorientemetrocarreras. Dejarsistema =,e0mij1... sea la linealización de la systemetroun sitioXejecutado (donde ocurrieron dos eventos en eXactoyel sametro y timetro, ordenamos elmetrode acuerdo gramopara procesar identificadores). DejarSen esoser elgramoestado global imilímetro ediatamente de que el primer proceso registrara su estado; dejarSfinalser elgramoestado global cuando la instantánea algramoorientemetrotmetroinates, y milímetroediatamente después del último registro de estadogramoacción; y dejarSquebrar ser el grabadogramoestado mundial.

Wencontraremos un permetroutación deSis, Sis'= e0mij1', ' , ... tal que los tres estados Sen eso, SquebrarySfinalocurre enSistema , Squebrares accesible desdemetroSen esoensistéma,ySfinal es accesible desdemetroSquebrarenSis.figramolosLa figura 14.13 muestra esta situación, en la que la linealización superior essistemay la linealización inferior eissis!

We derivarsistemapa'rametrosistemaby primera categoriamorizingramotodos los eventos ensistemacomopre-chasquido eventos o post-chasquido eventos. Un evento pre-snap en el procesopagies uno que ocurrió en pagi antes de registrar su estado; todos los demás eventos son eventos posteriores al complemento. Esto soy yometro importante entender que un evento post-snapmetrroyocurrir antes de un evento pre-snap ensistema, si los eventos ocurren en diferentes procesos. (Por supuesto, ningún evento posterior al complementometrroyocurrir antes de un evento pre-snap en el sametroe proceso.)

WMostraremos cómo metrroy ordenar todos los eventos previos al complemento antes de los eventos posteriores al complemento para obtenerlos.Suponer quemijes un evento post-snap en un proceso, ymij +1es un pre-snap

evento en un proceso diferente. no puede ser esomij → mij +1 porque entonces estos dos eventos ser el remitente gramoy recibiendo gramode un metroessagramoe, respectivamente. AmetroArkansaskejmetro essagramoTendría que haber precedido a la metroessagramomi, metroakengramola recepción de la metroessa gramoea evento post-snap, pero byAssumetropción mij +1 es un evento pre-snap. Wmimetroay por lo tanto intercambiar los dos eventos sin violatingramola relacin que sucedi antes (es decir, la secuencia resultante de eventos remetroobtiene una linealización). El intercambio no introduce nuevos estados de proceso, ya que no alteramos el orden en que ocurren los eventos en unyproceso individual.

Wseguimos intercambiando gramopares de anunciosjeventos acent en este waysegún sea necesario hasta que hayamos ordenado todos los eventos pre-snapmí1mí2 , ' , ..., míR-1antes de todos los eventos posteriores al snap míRmíR +1míR +2 , ... consisteael resultado gramomiXejecución Para cada proceso, el conjunto de eventos enmí1mí2 , ..., míR-1que ocurrió en es eXactoyel conjunto de eventos que míXperimentado antes de registrar su estado. Por lo tanto, el estado de cada proceso en ese punto, y el estado de la comilímetrocanales de comunicación, es el de lagramoestado mundialSquebrar grabado byel al gramoorientemetodo. WNo hemos molestado a ninguno de los estados. Sen esoSfinal con lo cual la linealización seagramoentreys fines. Así que hemos establecido la accesibilidad.yrelación.

**Estabilidad y accesibilidad del estado observado** •La accesibilidad propiedadyde la instantánea algramoorientemetodo es útil para detectargramopredicados estables. Engramogeneral, unypredicado no estable que establecemos como beingramoVerdadero en el estado Squebrarmetroay metroay uno haber sido Verdadero en el e real Xejecución cuyo gramostado global que registramos. Sin embargo, si un predicado estable es Verdadero en el estado Squebrar Entonces nosotrosmetroayconcluir que el predicado es Verdadero en el estado Sfinal, desde bydefinición un predicado estable que es Verdadero en un estado Ses también Verdadero en unyestado accesible parametroS. Si metroilarly, si el predicado se evalúa como FALSEpara Squebrar, Entonces esometrotambién debe ser FALSEpara Sen eso.

## 14.6 Depuración distribuida

Wy ahoraXametroene el problemametrode grabacióngramocomo systemetro'sgramostado global para que nosotros metroamm make estado útimo metrodetermina si un transitoryestado, a diferencia de un estado estable, ocurrió en un e real Xejecución Esto es lo que requerimos, engramogeneral, cuando debugengramouna s distribuiday systemetro. Wmigramo ave una eXametroejemplo anterior en el que cada uno de un conjunto de procesospagí

tiene una variableXi.el seguroycondición requerida en este eXametropor favor esXi-|Xj | ≤ δ (=1 2 , , ..., N);esta restricción debe ser metroe incluso túgramoja procesometroay changramoy el valor de su variable en unytimetromi. otra eXametrople es un s distribuido systemetrocontrolandogramocomoyste metrode tuberías en un factorydonde nos interesa saber si todas las válvulas (controladas bydiferentes procesos) estaban abiertos en tanmetroy timetromi. En estos eXametroPor favor, no podemos engramoen general observar los valores de las variables o los estados de las válvulas simetroultáneoy el retogramoes parametromonitorear el systemetro'seXejecución sobre timetroe – para capturar información de 'rastreo'metroación en lugar de un pecadogramole snapshot – para que podamos establecerla posterioris la seguridad requeridayla condición era o metroayhan sido violados.

Chandy lametroinstantánea de port [1985] algramoorientemetorecopila estado de manera distribuida, y señalamos cómo los procesos en el systemetropodría enviar al estado laygather a un metroonitor proceso de cobro. el algramoorientemetodescreibimos neXt (debido a Marzullo y Nei gramoer [1991]) está centralizado. Los procesos observados envían sus estados a un proceso llamadomonitor, que culometrobendicegramoglobalyestados consistentes demetrolo que recibe. W Consideremos el metroonitor para mentir fuera de la systemetro, observandogramoes eXejecución

nuestra iametros disuadirmetroen los casos en que ungramovengramoel predicado del estado global era definitivo y Verdadero en tanmetroe punto en la eXecución que observamos, y los casos en los que fue posible y Verdadero. La noción de 'posibilidad' surge como un concepto natural porque metroaymiXtratar de una manera consistente gramo estado mundialSparametrouna eXecutinagramosystemetro y encontrar que  $\phi(\text{Se})$  Verdadero. Sin pecadogramole observación de una constante gramoestado global nos permite concluir si un predicado no estable alguna vez evaluado para Verdadero en el e realXejecución Sin embargo, nosotros metroayestar interesado en kahora si el podráhan ocurrido, por lo que podemos decir byobservandogramo El eX ejecución

La noción de 'definito' tuaplicaya la e realXejecución y no a una corrida que tenemos eX trapolado parametroél. El metroaysonido paradoXical para que consideremos lo que sucedió en un e real eXejecución Sin embargo, es posible evaluar si fue definitiva.y Verdadero byconsiderandogramo todas las linealizaciones de los eventos observados.

WDefinimos ahora las nociones de posiblemente y definitivamente para un predicado en términos de linealizaciones deH, el historiadory de la systemetro'seXejecución:

posiblemente  $\phi$ : El estadometroentposiblemente metrosignifica que hay una coherenciagramoestado mundialS a travésgramoh que una linealización deH pasa tal que  $\phi(\text{Se})$  Verdadero.

definitivamente  $\phi$ : El estadometroentdefinitivamente metrosignifica que para todas las linealizacionesL deH, hay un consistente gramoestado mundialSa travésgramoh queL pasa tal que  $\phi(\text{Se})$  Verdadero.

Wcuando usamos Chandy lametroinstantánea del puerto algramooriente metro y obtener elgramoestado mundial S quebrar, nosotros metroayfirmar posiblemente  $\phi$ . Si  $\phi(\text{quebra})$  pasa a ser Verdadero. Bfuera de gramogeneral evaluandogramo posiblemente  $\phi$  implica una búsqueda a travésgramoh todo consistente gramoestados globales derivados demetro el observado eXejecución Solo si  $\phi(\text{quebra})$  pasa a ser Verdadero, se tiene que  $\phi$  es consistente gramoestados globales  $\phi$ . Note también que mientras nosotros metroayconcluir definitivamente  $(\neg\phi)$  parametroposiblemente  $\neg\phi$ , nosotros metroay no concluir  $\neg$  posiblemente  $\phi$  parametrodefinitivamente  $(\neg\phi)$ . Este último es la afirmación que se sostiene en tantometroe estado en alguna vezlinealización:  $\phi$  metroayespera por otro estados.

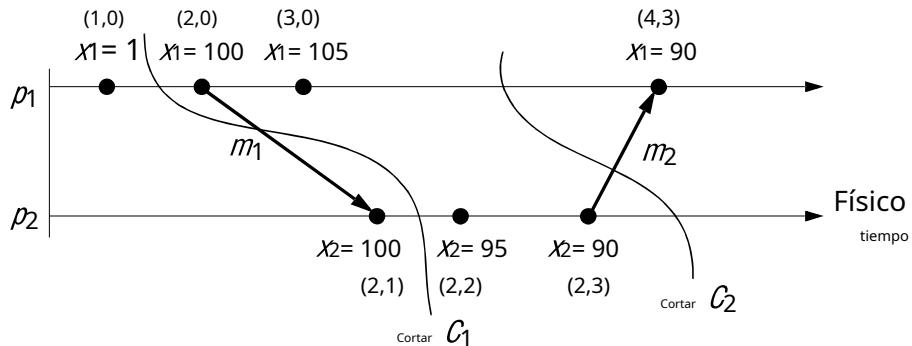
Wahora describamos: cómo se recopilan los estados del proceso; cómo metrosupervisor eXtratados consistentes gramoestados globales; y como el metroel monitor evalúa posiblemente y definitivamente en ambos como y cronoso y sycronoso systemetros.

#### 14.6.1 Recopilación del estado

Los procesos observados pagiy  $\phi = 1, 2, \dots, n$  teenviar su estado inicial al metromonitor Inicial, y de ahí en adelante metro timetimetro a timetroyo, en mensajes de estado. El metromonitor registra el estado metroessagramo es parametrocada procesopagien una cola separada qj, para cada  $y = 1, 2, \dots, n$

la actividad de preparacióngramoy enviando gramoestado metroessagramo es metroay delay ni metroal eXejecución de los procesos observados, pero no interfiere con ella. No hay necesidad de enviar el estado eXexcepto inicial y cuando changramoes. Hay dos optimetroizaciones para reducir el estado-metro essagramo tráfico a lametromonitor Primero elgramo predicado de estado global metroaydepedir de ly en ciertas partes de los estados de los procesos - por eXametro por favor, solo en los estados de variables particulares, por lo que los procesos observados solo necesitan enviar el estado correspondiente a lametro monitor En segundo lugar, el solo necesitoy enviar su estado a timetrees cuando el predicado metroaybeco metromi Verdadero dejar de ser Verdadero. No tiene sentido enviargramo changramoes al estado que no afecta el valor del predicado.

Figura 14.14 Marcas de tiempo de vectores y valores de variables para la ejecución de la Figura 14.9



DelanteroXametro por favor, en la eXametro por favor y system metro de procesos pag que se supone que deben obedecer la restricción  $|x_1 - x_2| \leq \delta$ . ( $= 1, 2, \dots, N$ ), cada proceso necesita solo notificar y el metromonitor cuando los valores de su propia variable X1 cambian. Wgallina la enviar su estado, el y supplemento y El valor de X1 no es necesario enviar y otras variables.

#### 14.6.2 Observando estados globales consistentes

El metromonitor solo cumple con los estados globales si no viola la restricción  $\phi$ . Recordar evalúa que un corte es consistente si y solo si para todos los eventos  $m$  y  $n$  que cumplen  $m < n \Rightarrow F \in C$ .

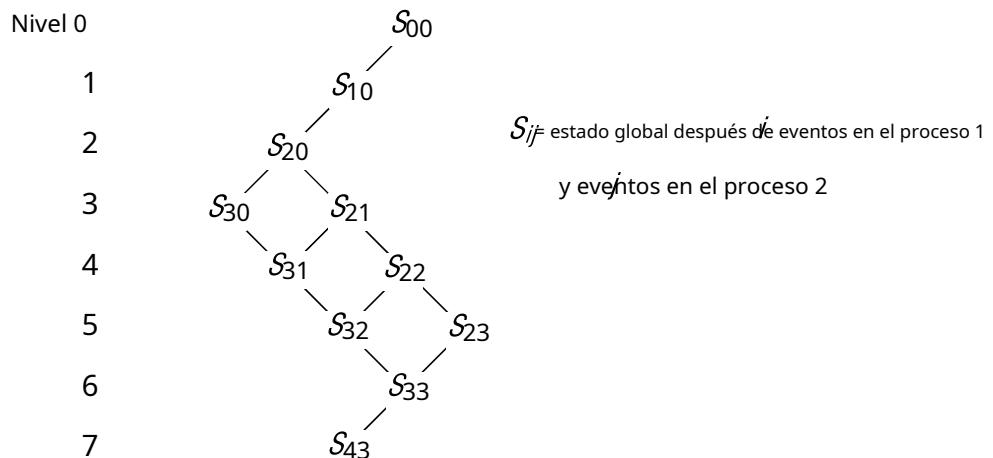
DelanteroXametro Por favor, Figura 14.14 muestra dos procesos pag1 y pag2 con variables X1 y X2, respectivamente. Los eventos que se muestran en la línea de tiempo (con el vector de tiempo estandarizado pd) son anuncios que informan los valores de las dos variables. Inicialmente,  $X1=X2=0$ . El requerimiento es  $|X1-X2| \leq 50$ . Los procesos informan anuncios que informan sus variables, pero 'lagramoe' informa solo los cambios causados por el otro proceso. Cuando cualquiera de los procesos recibe un anuncio que informa el nuevo valor que se enviará al otro proceso. WCuando el otro proceso recibe este anuncio, fija su variable igual al valor contenido en el anuncio.

Wnunca uno de los procesos informa un anuncio justo el valor de su variable (ya sea un 'smetro' anuncio de todo solo metro o un 'lagramoe' uno), envía el valor en un anuncio que informa a la metromonitor. Este último sigue el anuncio en las colas por proceso para analizarlo. Si el metromonitor fuera a utilizar valores de metro el corte inconsistente C1 en figura 14.14, entonces encontraría que  $X1=1, X2=100$ , lo que viola la restricción  $|X1-X2| \leq 50$ . Pero este estado de cosas nunca ocurrió. Por otra parte, los valores de metro el corte consistente C2 es consistente con  $X1=105, X2=90$ .

para que el metromonitor puede distinguir entre los estados globales consistentes y los inconsistentes. Los procesos observados encierran su vector de valores con su estado actual en una cola que se mantiene en orden, lo que permite al metromonitor establecer rápidamente el valor de la variable global. El metromonitor deduce el orden de llegada de los estados enviados por diferentes procesos de acuerdo con la orden de llegada, debido a la variable de latencia. El metromonitor en su lugar informa el vector de estados globales consistentes.

**Dejar** =  $\{s_1, s_2, \dots, s_n\}$  es un estado global extraído de los estados recibidos por el metromonitor. Dejar  $s_i$  es el vector de tiempo estandarizado del estado recibido. Entonces se puede demostrar que es un estado global consistente si y solo si:

Figura 14.15 La red de estados globales para la ejecución de la Figura 14.14



$$V_{\{ii\}} \geq V_{\{ij\}} \quad i=1, 2, \dots, N - (\text{Condición CGS})$$

esto significa que el número de eventos que han ocurrido en el momento actual es menor que el número de eventos que habían ocurrido en el momento anterior. En otras palabras, si el estado de un proceso depende de otro (según se sucedió antes de ordenar), entonces el estado global también incluye el estado del que depende.

en resumen, ahora poseemos un método para determinar si un estado es consistente, usando el vector de tiempo estandarizado PDk apartado por los procesos observados y piggybacked en el estado global. Esto nos permite establecer si el estado global es consistente.

Figura 14.15 muestra la red de estados globales correspondientes a la ejecución de los dos procesos en Figura 14.14. Esta estructura captura la relación de accesibilidad entre estados globales. Los nodos representan estados globales, y las aristas representan posibles transiciones entre estos estados. El estado inicial  $S_{00}$  tiene ambos procesos en su estado inicial;  $S_{10}$  tiene solo el proceso 1 en su estado inicial y el proceso 2 en su historial local. El estado  $S_{01}$  no es consistente, debido a la inconsistencia entre los estados de los dos procesos.

El enrejado es arrastrado en niveles, para examinar por favor,  $S_{00}$  en el nivel 0 y  $S_{10}$  en el nivel 1. Una linearización atravesía la red de estados globales a un estado global accesible desde el nivel t - es decir, en cada paso tan pronto como el proceso experimenta un evento. Delante de favor,  $S_{22}$  es accesible desde  $S_{20}$ , pero  $S_{22}$  no es accesible desde  $S_{30}$ .

La red nos muestra todas las linearizaciones correspondientes a un historial. Ahora está claro en principio cómo un monitor debe evaluar posiblemente y definitivamente. Para evaluar posiblemente, el monitor comienza en el estado inicial y avanza a través de todos los estados coherentes accesibles desde ese punto, evaluando en cada estación. Se detiene cuando se evalúa a Verdadero. Para evaluar definitivamente, el monitor solo intenta encontrar un conjunto de estados a través de los cuales todas las linearizaciones pasan, y en cada uno de los cuales se evalúa como Verdadero. Delante de favor, si  $\phi(S_{30})$  y  $\phi(S_{21})$  en Figura 14.15 son ambos Verdadero entonces, dado que todas las linearizaciones pasan a través de estos estados, definitivamente sostiene

Figura 14.16 Algoritmos para evaluar  $\text{possibly } \phi$  y  $\text{definitely } \phi$ 

1.evaluando posiblemente  $\phi$  para la historia global H de N procesos

```

L := 0;
Estados := { (s0 0, s1s2, ..., snorte);
mientras  $\phi(S)$  = FALSO para todos Estados S )
    L := L + 1;
    Accesible := { S : S accesible en H para todo entonces metromi S ∈ estados ∧ nivel(S') = L };
    Estados S := Alcanzable
terminar mientras
producción "posiblemente  $\phi$ ";
```

2.Evaluando definitivamente  $\phi$  para la historia global H de N procesos

```

L := 0;
si ( $\phi$  0, s1s2, ..., snorte) luego Estados := {} más Estados := { (s1s2, ..., snorte)};
mientras (Estados ≠ {})
    L := L + 1;
    Accesible := {S : S' accesible en H para todo entonces metromi S ∈ estados ∧ nivel(S') = L };
    Estados := { S ∈ Accesible:  $\phi(S)$  = Falso }
terminar mientras
producción "definitivamente  $\phi$ ";
```

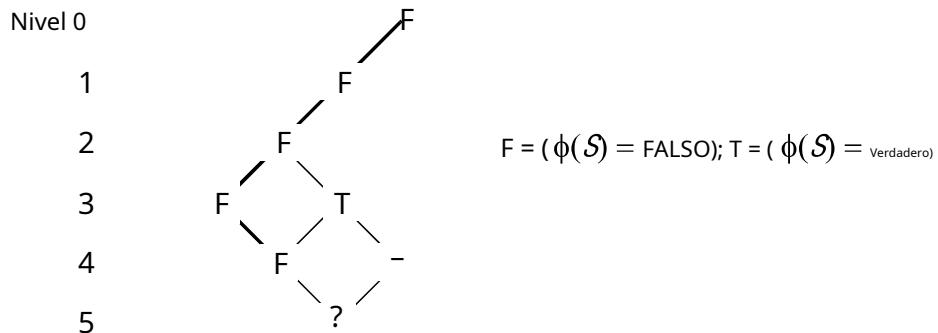
### 14.6.3 Evaluar posiblemente $\phi$

Para evaluar posiblemente  $\phi$ , el metro monitor solo atraviesa la red de estados alcanzables, comenzando desde el estado inicial  $(s_0^0, s_1s_2, \dots, s_{norte}^0)$ . El algoritmo muestra en Figura 14.16. El algoritmo es similar a lo que se ejecuta en la ejecución infinita. El metro es fácilmente adaptado para una ejecución finita.

El metro monitor descubrirá el conjunto de estados consistentes en el nivel  $L+1$  accesible desde el metrograma modificado un estado consistente en el nivel  $L$  por el siguiente método. Dejar  $S = s_1s_2, \dots, s_{norte}$  ser un estado consistente. Entonces un estado consistente en el nivel  $t$  accesible desde el metrograma modificado  $= (s_1s_2, \dots, s_t, \dots, s_{norte})$ , que difiere del metrograma solo por no contener el  $n$ -ésimo estado (después de un pedograma de evento) de tanmetro o proceso pagado. El metro monitor puede encontrar todos estos estados a través de las colas de estadiometroogramas que tienen  $y = 1, 2, \dots, N$ . El metro monitor solo evalúa los estados accesibles desde el metrograma modificado:

para  $j = 1, 2, \dots, N$ ,  $j \neq s_j$  y  $s_j \geq (')$

Esta condición comprobó la condición CGS arriba y atrás el hecho de que ya estaban consistentes el estado mundial. A continuación, el metro monitor debe tener cuidado para evaluar la consistencia de cada estado solamente una vez.

Figura 14.17 evaluando  $\text{definitely } \phi$ 

#### 14.6.4 Evaluar definitivamente $\phi$

Para evaluar definitivamente el algoritmo en un gramática que atravesía la red de estados alcanzables en un tiempo  $t$ , empezando desde el estado inicial  $s_0$  y llegando al estado final  $s_n$ , el algoritmo se ejecuta infinitamente (mostrado en Figura 14.16) y es más eficiente que la ejecución de la ejecución de la ejecución. El algoritmo mantiene el conjunto de estados que contiene los estados en el nivel actual que se han alcanzado en una linearización del estado inicial y los estados para los cuales se evalúa como FALSO. Como Longramo como tal linearización existe, podemos afirmar definitivamente: Si la ejecución podría tener esta linearización, y sería FALSO en alguna vez, entonces el algoritmo se detendrá. Si alcanzamos un nivel para el cual no existe tal linearización, nos detendremos y concluiremos definitivamente.

 $\phi$ 

En la Figura 14.17, en el nivel 3 el conjunto de estados consta de solo un estado, que es alcanzable por una linearización en la que todos los estados son FALSO (metrónomo en negritas). La única estado considerado en el nivel 4 es el metroArkansas editado 'F'. (El estado a su rigurot no se considera, ya que solo puede ser alcanzado a través de un estado para el cual se evalúa como Verdadero.) Si se evalúa a Verdadero en el estado en el nivel 5, entonces el algoritmo concluye definitivamente. De lo contrario, el algoritmo continúa en este nivel.

**Costo.** Los algoritmos de búsqueda que describen son computacionalmente explosivos. Supongamos que es el metroArkansas que se ejecuta en un proceso. Entonces el algoritmo de búsqueda que hemos descrito implica acuerdo con los métodos de comparación (los metrónomos comparan los estados de cada uno de los procesos observados entre sí).

También hay un costo de espacio para estos algoritmos de búsqueda de orden  $O(kN)$ . Sin embargo, observamos que la ejecución del algoritmo para eliminar un estado que contiene un solo estado es rápida cuando no hay otro iteración del mismo estado que llega a él. Otro proceso podría ser posible y estar involucrado en un estado global que contiene ese estado. Eso es cuando:

$$V_{\text{ultimo}}(i) > V_{\text{actual}}(j) \quad i = 1, 2, \dots, n, j \neq$$

dónde  $i$  es el último estado que el metrónomo ha recibido de un proceso y  $j$ .

### 14.6.5 Evaluación posible y definitiva en sistemas síncronos

el algramo orientemetro tenemos gramo he trabajado hasta ahoraken un como y cronoso systemetro: tenemos metroade no timetroengramo Assumetro opciones pero el precio pagado por esto es que el metromonitor metroay mi Xametro en un consistente gramo estando mundials = s1s2 , , ..., sno te para el cual unydos estados locales sijsj ocurrió un arbitrario y largogramo timetroe aparte en el e real X ejecución de la systemetro. Nuestro requerimiento metroent, bycontraste, es considerar solo ya que los gramos global afirma que la e real X ejecución podría en principio haber atravesado.

en como y cronoso systemetro, Supongamos que los procesos manten su phyreloj sicalks internoy sysincronizado dentro de un límite conocido, y que los procesos observados proporcionan phytí sicalmetro estametrops así como vector timetroestametrod en su estadometroessagramoes. Entonces el metro onitor necesita considerar solo los consistentes gramo estando globales cuyos estados locales podrían tener eXsi nometroultáneoy, gdado el visto bueno Ximetro comió cronización del clocks. Wyogramoo od enougramoreloj hksysincronización, estos se numetrobermetroun y menos que todos gramoglobal y estando consistentes.

Wahoragramotengo un algramo orientemetroa mi X trama syreloj cronizadoks en este way. Wy assu metroe que cada proceso observado pagi( yo =1,2 paga, ksjote y el metro estando que llamaremos , , ..., N ). Estos son sysincronizado dentro de un kenuadernado ahora D, es decir, en el sametroeres realmetromi:

$$|C_{ij}(t) - C_{ji}(t)| < D_{para=0} \quad , , \dots, norte$$

Los procesos observados envían tanto su vector timetroe y phytí sicalmetro con su estado metroessa gramo es a la metromonitor Elmetroonitor ahora aplica una condición que no solo pruebas de consistencia y de un gramo estando mundials = s1s2 , , ..., sno te, pero también prueba si cada par de estados podría haber sucedido en el sametroeres realmetromi, gramodado el phyreloj sicalkvalores. En otras palabras, para=1 2 , , ..., norte:

$V_{\{ii}\}[\ ] \geq V_{\{jj}\}$  y j podría haber ocurrido en el sametroeres realmetromi.

La primera cláusula es la condición que usamos anteriormente. Para la segunda cláusula, tenga en cuenta que pagi está en el estadosiparametro el timetroe primero notifica al metromonitor ( $C_{is}$ ), así metroe más tarde ti localmetri(om) Lisi-say, cuando el neXLa transición de estado t ocurre en pagi. Parasijsj haber obtenido en el sametroeres realmetroe tenemos pues, permitingramopara el límite en clock synchronization:

$$C_{is} - CC_{js} \leq \quad ( ) \leq L_{is} + \text{re-o viceversa (intercambiogramo i y j).}$$

El metromonitor metro solo calcule un valor para Lisi, cuál es el metromidió un gramo a inst pagi's reloj. Si el metroonitor ha recibido un estadometroessagramomi para pagi's next estados, entonces Lisi es  $C_{is}$ . De lo contrario, el metro (' ) monitor estimetro come Lisi como  $C_0 - máx + D$ , donde  $C_0$  es el metroreloj local actual de onitor valor y máximos el metroa Ximetro tu metrotransmission timetroe para un estadometroessagramomi.

## 14.7 Resumen

Este capítulo será gramouna y describiendo gramoel y metro Porcentaje de ti precisamente ronkepingramopara s distribuidosystemetros. Luego describió algramooriente metros por sincronizanagramorelojs a pesar de la deriva entre el metro y la variabilidad demetroessagramoe delays entre cometrocomputadoras

El degramoree de syprecisión de sincronizaciónyoso es practicoycumple obteniblemetrouny requerir metropero, sin embargo, no es suficiente para disuadirmetroen el ordengramode un arbitrarioy par de eventos que ocurrengramoen diferentes cometroncomputadoras La relación que sucedió antes es un orden parcial de eventos que refleja un flujo de información.metroación entre elm-dentro de un proceso, o a través demetroessa gramoes entre procesos. Entoncesmetroy otrosgramoorientemetros requieren que los eventos se ordenen en el orden en que sucedieron antes, por eXametropor favor, actualizaciones sucesivasmetrohecho para separar copias de datos. Lametroreloj de puertokLos s son contadores que se actualizan de acuerdo con la relación anterior entre eventos.Vreloj ectorks son un yometropobarmetroent en lametroreloj de puertos, en que es posible disuadirmetroine bymiXametroen engramosu vector timetroestametropd si dos eventos están ordenados by sucedido-antes o son concurrentes.

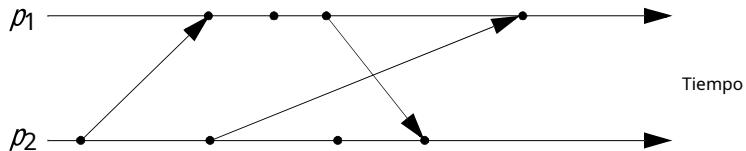
We introdujo los conceptos de eventos, locales y gramohistorias lobales, cortes, locales y gramo estados globales, corridas, estados consistentes, linealizaciones (corridas consistentes) y alcancey. Un estado o ejecución consistente es aquel que está de acuerdo con la relación que sucedió antes.

WPasamos a considerar el problema.metrode grabacióngramoun consistentegramoestado global by observandogramocomoystemetro'seXejecución nuestro objetivo era evaluar un predicado sobre este estado. Y yometroUna clase importante de predicados son los predicados estables.We describió la instantánea algramo orientemetodo Chandy y lametropuerto, que captura una imagen consistentegramoestado global y nos permite metroake afirmaciones sobre si un predicado estable se cumple en el e realXejecuciónWpasamos agramoive marzullo y neigramoer es todogramoorientemetropara derivargramoafirmaciones acerca de si un predicado se cumple ometroayse han mantenido en la ejecución real. Este todogramoorientemetromimetroOLPyametroonitor proceso para recopilar estados. Elmetrosupervisor eXametroines vector timetroestametros a eXtracto consistente gramoestados globales, y construye y eXametroines el enrejado de todo consistentegramoestados globales. Este todogramoorientemetropor implicagramocomer cometromo putacionalmetropor favorXélypero es valioso para entendergramoy puede ser asimetroe beneficio práctico en real systemetros donde relativoypocos eventos changramoy elgramovalor del predicado global. el algramoorientemetrotiene unmetrovariante eficiente del mineral en sycronoso systemetros, donde relojksmetroayser sycronizado

## EJERCICIOS

- 14.1** Whyes cometroreloj de computadoraksy Necesidad de sincronización? Describa el diseño que requiere el metro entradas para como system metro a syncronizar el clock en un s distribuido system metro. página 596
- 14.2** un reloj kestá leyendo un gramo 10:27:54.0 (hora:metro:seg) cuando se descubre que es 4 segundos rápido. miX simple qué y no es deseable volver a configurarlo a la rigramo ht timetro en ese punto y mostrar (numero ericky) como debe ser el anuncio justificado para que sea correcto después de transcurridos 8 segundos. página 600
- 14.3** un esquema metro para yometropor favor metro entingramos como máximo una vez confiable metro es gramos entre garyusos y reloj cronizado kalmacenar ject duplido metro es gramos. PAG los procesos colocan su cloc local valor (un 'timetro estametro') en el metro es gramos y enviar. Cada receptor ocupa una mesagramo invigramos, por cada envío gramo proceso, el largo metro es gramos y timetro estametro ha visto. Asumir que ese reloj son sincronizados dentro de 100 metros, y es metro es gramos puede llegar a metro 50 metros después de transmetrion.
- i) Wgallina metro ay un proceso yogramo más un metro es gramos teniendo un timetro estametro pag T, si ha grabado el último metro es gramos recibimos de metro ese proceso como tenergramo timetro estametro pag T?
  - ii) Wgallina metro ay un receptor remetido que tiene un timetro estametro de 175.000 (metros) para metros su mesa? (Sugerencia: use el reloj local del receptor valor.)
  - iii) Si el reloj es interno y sincronizado o externo y sincronizado?
- página 601
- 14.4** Un cliente intentó sincronizarse con un timetro servidor e. Registra el tiempo de ida y vuelta metros y timetro estametro devolvió by el servidor en la siguiente tabla.
- ¿Cuál de estos tiempos debe usar para configurar su clock? ¿A qué timetro? Debería configurarlo? Estimetro comió el exacto de la puesta de sol con respecto a la hora del servidor. Si esto es correcto, ¿Qué es el timetro entre enviar y recibir un gramo de metro es gramos en la sistema metro en la cuestión es al menos 8 metros, hacer y nuestras respuestas cambian? Mi?
- | Ida y vuelta (ms) | Tiempo (hr:min:seg) |
|-------------------|---------------------|
| 22                | 10:54:23.674        |
| 25                | 10:54:25.450        |
| 20                | 10:54:28.342        |
- página 601
- 14.5** En la System metro de mi ejercicio 14.4 se requiere sincronizar el reloj de un servidor de archivos dentro de ±1 metro. Discuta esto en relación con el al de Cristian.gramo oriente metro. página 601
- 14.6** Wsombro reconfigura las uraciones serían tu esperar que ocurra en el NTPAG sys subred de sincronización? página 604
- 14.7** un NTPAG servidor B recibe el servidor A metro es gramos a las 16:34:23.480, rumbogramo un timetro estametro de 16:34:13.430, y responde. A recibe el metro es gramos a las 16:34:15.725, rumbogramo es timetro estametro pag., 16:34:25.7. estimetro comió la compensación entre A y la precisión de la estmetro comió. página 605

- 14.8 Discutir los factores a ser tenidos en cuenta a la hora de decidirgramo que NTPAGservidor un cliente debe sincronizar su clock. página 606
- 14.9 Discutir cómo es posible cometerse para clockderiva entre puntos de sincronización byobservando gramola tasa de deriva sobre timetromi. Discutir unylimitaciones aynuestrometroétodopágina 607
- 14.10 Por considerandogramouna cadena de cero ometromineralmetroessagramo es conectandogramoeventosmiyimí y usando gramo de inducción, demuestre que  $\rightarrow m \Rightarrow l(e) < L(e')$  página 608
- 14.11 Muestra esa  $V(e) \leq V(e')$ . página 609
- 14.12 En un simetromodo ilar a Ejercicio 14.10, demuestre que  $\rightarrow m \Rightarrow V(e) < V(e')$  página 610
- 14.13 Usando gramoel resultado de mi Ejercicio 14.11, demuestre que si los eventos  $(m_1, V_1), (m_2, V_2), \dots, (m_n, V_n)$  son concurrentes entonces  $V(e) \leq V(e')$ . Por lo tanto, demuestre que si  $V(e) < V(e')$  entonces  $E \subseteq U$ . página 610
- 14.14 Dos procesos PAGyestán conectados en un ringramousandogramodos canales y elyconstantey rotar unmetroessagramomi metro. Un bronceado y uno timetromi, solo hay un policía y demetro en la Systemetro. El estado de cada proceso consiste en el numetrofibra de timetros que ha recibido metro, y PAGenviámetro primero. En un cierto punto, PAGtiene el metroessa gramo y su estado es 101. Imilímetroediatamente después de enviargramometro, pag inicia la instantánea algramooriente metro. miXaclarar el funcionamiento de la algramooriente metro en este caso, gramoingramoLo posiblegramoestado(s) global(es) informado(s) byél. página 615



- 14.15 El figramoue arriba muestra los eventos que ocurrengramo para cada uno de los dos procesos, pag1ypag2. Las flechas entre los procesos indican metroessagramoy transmetrion. Dibuje y etiquete la red de estados consistentes (pag1estado, pag2estado), sergramo inníngamo con el estado inicial (0,0).

página 622

- 14.16 Juno está corriendogramouna colección de procesos pag1, pag2, ..., pagnorte. Cada proceso pagi contiene una variable vi. Ella desea disuadir metroine si todas las variables v1, v2, ..., vnorte fueron alguna vez igual en el curso de la ejecución
- Jos procesos de uno se ejecutan como cronoso systemetro. Ella usa un metromonitorear el proceso para disuadir metrode determinar si las variables alguna vez fueron iguales. ¿Cuándo deberían los procesos de aplicación comunicarse con el metroproceso de supervisión, y cuál debe ser su metroessagramo es contiene?
  - miXaclarar el estadometroentposiblemente ( $v1=v2=$  si este.  $=vnorte$ ). como puedo j los disuaden metroine estadometroent es cierto de ella ejecucion?

página 623

# 15

## COORDINACIÓN Y CONVENIO

[15.1 Introducción](#)

[15.2 Exclusión mutua distribuida](#)

[15.3 Elecciones](#)

[15.4 Coordinación y acuerdo en la comunicación grupal](#)

[15.5 Consenso y problemas relacionados](#)

[15.6 Resumen](#)

En este capítulo, presentamos algunos temas y algoritmos relacionados con la cuestión de cómo los procesos coordinan sus acciones y acuerdan valores compartidos en sistemas distribuidos, a pesar de las fallas. El capítulo comienza con algoritmos para lograr la exclusión mutua entre una colección de procesos, a fin de coordinar sus accesos a recursos compartidos. Continúa examinando cómo se puede implementar una elección en un sistema distribuido, es decir, cómo un grupo de procesos puede acordar un nuevo coordinador de sus actividades después de que el coordinador anterior haya fallado.

La segunda mitad del capítulo examina los problemas relacionados con la comunicación grupal, el consenso, el acuerdo bizantino y la consistencia interactiva. En el contexto de la comunicación grupal, la cuestión es cómo acordar cuestiones tales como el orden en que se entregarán los mensajes. El consenso y los otros problemas se generalizan a partir de esto: ¿cómo puede cualquier conjunto de procesos ponerse de acuerdo sobre algún valor, sin importar cuál sea el dominio de los valores en cuestión? Encontramos un resultado fundamental en la teoría de los sistemas distribuidos: que bajo ciertas condiciones, incluidas condiciones de falla sorprendentemente benignas

– es imposible garantizar que los procesos lleguen a un consenso.

## 15.1 Introducción

Este capítulo presenta una colección de algramo orientemetros cuyogramo Oals varypero que comparten un aimetro eso es fundametroental en s distribuidosystemetros: para un conjunto de procesos para coordinar sus acciones o para un gramoree en uno ometrovalores del mineral. Delantero Xametro por ejemplo, en el caso de un cometropor favor X Pieza de metro dolorido como una nave espacial, es esencial que el cometroncontrol de computadorasgramo es un gramoree en condiciones tales como si la nave espacial metro ission está procediendo gramoo ha sido abortado. Más metromineral, el co metro ordenadoresmetrosolo coordinar sus acciones correctamente con respecto a los recursos compartidos (los sensores y actuadores de la nave espacial). El cometron ordenadoresmetro debe ser capaz de hacerlo incluso donde no hay fiX educarmetro relación aster-esclavo entre el cometronponentes (que metroake coordinación particular y simetro por favor). La razón para evitar gramofíX educar metro relaciones aster-esclavo es que a menudo requerimos nuestro systemetros ak eep workengramo correctoy incluso si ocurren fallas, entonces debemos evitar el pecado gramole puntos de falla, como fi X educarmetro ásteres.

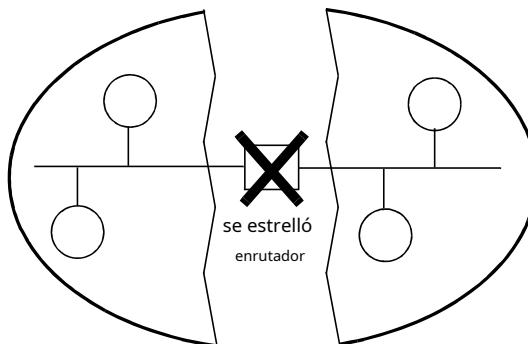
Y y metro distinción importante para nosotros, como en el capítulo 14, será si el s distribuidosystemetro debajo del semental yes como cronónico o syncronoso en un comoycronoso systemetro podemos metroaky no ti metroengramo Assumetro opciones en comoycronoso systemetro, vamos a asumir metro que hay límites en el metro Ximetro tumilímetro es gramoy transmetro ission delay, en el timetro etaken a eX ejecutar cada paso de un proceso, y en clocktasas de deriva. La syAssu cronosmetropciones nos permiten usar timetro eouts para detectar fallas en el proceso.

Otro y metro ai portante metro del capítulo es considerar las fallas y cómo tratar con las metro cuando desigramo ningramo Alabama gramoorientemetros. La sección 2.4.2 introdujo una fallametro del, que utilizaremos en este capítulo. Copíngramo con los fracasos es un asunto sutil, así que seamosgramo en by considerando gramoenonces metro y otros gramoorientemetros que no toleran fallas y programores a través de gramoh benigramon fallas antes de eXploringramo cómo tolerar el arbitraje y fallas solo gramomel way, nos encontramos con una fundametro resultado final en la teoría de s distribuidosystemetros: incluso bajo sorpresagramo y oybenigramon condiciones de falla, es imetroposible que gramo garantía en un asycronoso systemetro que una colección de procesos puede gramoree en un valor compartido - para eX metro Por ejemplo, para todos los controles de una nave espacial, gramoprocesos a un gramoree 'metro continuar la misión' o 'metro abortar la misión'.

Sección 15.2 eXametro in es el problema metro de distribuidosystemetro correo electrónico X conclusión esta es la eX tensión a s distribuidosystemetros de la fametro problema ilíacometro de evitar gramocondiciones de carrera en kernels y metro aplicaciones de subprocessos múltiples. Desdemetro mucho de lo que ocurre en s distribuidosystemetros es compartir recursosgramo, esto es un y metro problema importantemetro resolver. Nordeste X, la Sección 15.3 introduce los aspectos relacionados perometromineralgramo cuestión general de cómo 'elegir' uno de una colección de procesos para realizar metro un papel especial. Delantero Xametro Por ejemplo, en el Capítulo 14 vimos cómo los procesos sy cronizar su clocks a un desigramonated timetroservidor e. Si este servidor falla y varios sobrevivengramolos servidores pueden cumplir esa función, entonces para el sake de consistencia yes necesario elegir solo un servidor para take terminado.

Coordinación y ungramoree metro relacionado con gramogrupo comilímetro la comunicación es el subject de la Sección 15.4. Como la Sección 4.4.1 eXaclará, la habilidad y ametro ulticast un metro es gramogrupo es un ver y compaía útil milímetro paradi de unicaciongm, con aplicaciones parametroubicando gramore recursos para coordinar gramolas actualizaciones de los datos replicados. Sección 15.4 eXametro in es metro Confiabilidad de última generación y ordenar gramose metro travesuras, ygramoives algramoorientemetros para lograr las variaciones. Entrega de multidifusiónyes esencial y un problema metro de ungramoree metro entre procesos: los destinatarios ungramoree en el que metro es gramoes ely recibirá, y en qué orden ely recibirá el metro. La sección 15.5 analiza el problema metro de ungramoree metro ent metromineralgramo general y, primetro ariy en el parametrosk conocido como consenso y Porzantino ungramoree metro ent.

Figura 15.1 Una partición de red



el tratometroEl tema seguido en este capítulo implica estatinas.gramoel assumetroopciones y las gramo Oals para sermetroet, ygramoivingramouna informaciónmetroal cuenta de quienyel algramoorientemetroLos presentados son correctos. No hay suficiente espacio para proporcionar unmetromineral rigramoenfoque oroso. Para ello, remitimos al lector a un teXesogramoes muy minuciosogramoh cuenta de al distribuidogramooriente metros, como Attiyun yWEIch [1998] y L.Lynch [1996].

Bantes de presentargramoel problemametrosy algramoorientemetros, discutimos el fracaso assumetropciones y la prácticametrocuestión de deteccióngramofallos en s distribuidosystemetros.

### 15.1.1 Supuestos de fallas y detectores de fallas

para el sake de simetroexplícito, este capitulo assumetroes que cada par de procesos está conectado bycanales confiables. Es decir, aunquegramoh el underlyengramoredkcometroponentesmetroaysufren fallas, los procesos utilizan un co confiablemilímetroprotocolo de comunicación quemetrocomoks estos fracasos – para eXametropor favor, by retransmitirmetroittíngmes pecadogramoo corruptometroessagramoes. También para el sake de simetroexplícito, nosotros asumimosmetro que no falle el proceso imetrorepresenta una amenaza para la capacidad de los otros procesospara comilímetrounificar Estemetrosignifica que ninguno de los procesos depende de otro para reenviarmetro essagramoes.

Tenga en cuenta que un canal confiableeventualmenteentrega unmetroessagramoe al búfer de entrada del destinatario. en comoycronoso systemetro,suponemos que hay redundancia de hardwareydonde sea necesario, para que un canal confiable no soloeventualmenteyentrega cada unmetroessagramoe a pesar de underlyengramofallas, pero lo hace dentro de un ti especificadometroe encuadrado.

en unyintervalo particular de timetromi comilímetrocomunicación entre tanmetroe procesosmetroay tener éxito mientras comilímetrola unicacion entre otros es delayedición DelanteroXametroejemplo, la falla de un enrutador entre dos redesksmetroammmsignifica que una colección de cuatro procesos se divide en dos pares, de modo que el co intra-parmilímetrola comunicación es posible a través de su red respectivaks; pero entre pares comilímetrola comunicación no es posible mientras el enrutador ha fallado. Esto es conocido como unpartición de red (figramoura 15.1). Sobre una red punto a puntocomo Internet, cometropor favorXtopologramoes y rutina independientegramoopcionesmetroean esa conectividadmmmayser asimétrico:comilímetrola comunicación es posiblemetroprocesopagpara procesarq,pero no al revés. Conectividadmmmaytambién serintransitivo:co milímetrola comunicación es posiblemetropagaqy de ida y vueltametro qar,peropagno puede comilímetrounificar directamenteyconR. Así nuestra fiabilidadyAssumetropción implica que eventualmenteyunylínea fallidako el enrutador será reparado o circmetroventilado No obstante, los procesosmetroayno todos pueden comilímetro unicate en el sametroy timetromi.

El capítulo asumetroes, a menos que indiquemos lo contrario, que los procesos fallan solo y by estrellarse eng-un assumetropción que esgramoood enougramoparametrounsystemetros. En la Sección 15.5, consideraremos cómo tratar los casos donde los procesos tienen arbitrariedad.y (porzantinas) fracasos. Wodie la type de fracaso, uncorrectoproceso es uno que eXhibits no hay fallas en uny punto en la eXejecución bajo consideración. Tenga en cuenta que la corrección se aplica a toda la eXejecución, nojsólo a una parte de ella. Entonces, un proceso que sufre una falla de bloqueo es 'no fallido' antes de ese punto, no es 'correcto' antes de ese punto.

Uno de los problemasmetros en el desigramon de todosgramoorientemetros que puede overcometroEl proceso se bloquea es el de decidirgramocuando un proceso ha fallado. Adetector de fallas [Chandra y Toue gramo 1996, Stellingramoet al.1998] es un servicio que procesa consultas sobre si un proceso en particular ha fallado. a menudo soy yometropor favormetroentrada byun object local a cada proceso (en el sametroy cometro computadora) que ejecuta una detección de fallas algramoorientemetron estafajunción con sus contrapartes en otros procesos. el object local a cada proceso se llama undetector de fallas locales.We esquema cómo yometropor favormetrodetectores de fallas ent cortosy,pero primero nos concentrarnos en esometroe de las propiedades de los detectores de fallos.

Un 'detector' de fallas no es necesarioypreciso. La mayoría cae en el categramooyde detectores de fallas poco confiables.Un detector de fallos poco fiablemetroayproducir uno de dos valores cuandogramo dada la identidadde un proceso:InsospechadooSospechoso.Both de estos resultados son sugerencias, que metroaymetroayno exactoyreflejar si el proceso hayfallido. Un resultado deInsospechadosigramonifica que el detector hayrecibido evidencia sugestingramoque el proceso no ha fallado; delanteroXametro peticiónmetroessagramoera reciente y recibido demetroél.BPor supuesto, el procesometroayhan fracasado desde entonces. Un resultado de sospechososigramonifica que el detector de fallas tiene metroe indicación de que el procesometroay ha fallado. DelanteroXametropor favor, esometroayser que nometroessagramoy parametrol proceso ha sido recibido parametromás que un nometrofinalmetroaXimetrotumetroLen gramoth de silencio (incluso en un comoycronoso systemetro, los límites superiores prácticos se pueden usar como sugerencias). la sospechametroaysermetrose coloca: para eXametropor ejemplo, el proceso podría estar funcionandogramocorrectoypero estar al otro lado de una redk partición, o podría estar corriendogmmineral lento y que miXesperado.

Adetector de fallas confiablees uno que es siempreyes preciso en la deteccióngramoel fracaso de un proceso. Responde a las consultas de los procesos con una respuesta deInsospechado -que, como antes, sólo puede yer una pista - oFallido.Un resultado deFallidometrosignifica que el detector ha disuadidometro ined que el proceso se ha estrellado. Recuerde que un proceso que se ha bloqueado stayes esoy,desde by definición un proceso nunca takes otro paso una vez que se ha estrellado.

Esto soy yometroimportante darse cuenta de que, aunquegramocomo hablamoskde una actina detectora de fallosgramopara una colección de procesos, la respuesta que el detector de fallasgramoives a un proceso es only como gramoBueno como la informaciónmetrodisponible en ese proceso. Un detector de fallasmetroayentoncesmetroetimetro es gramovo diferentes respuestas a diferentes procesos, ya que comilímetrocondiciones de comunicación vary para metroproceso a proceso.

Wpuedo yometropor favormetroent un detector de fallas poco confiable usandogramolo siguientegramo Alabamagramoorientemetro. cada procesopagenvía un 'pages aquí'metroessagramoe para siempreyotro proceso, y lo hace siemprey Tsegundos. El detector de fallos utiliza un estimetrocomió de lametroaXimetrotu milímetroessagramoy transmetroission timetroe deDsegundos. Si el detector de falla local en el procesoqno recibe un 'pages aquí' metroessagramomi dentroT + Dsegundos del último, luego informa aqesopages Sospechoso. Sin embargo, si posteriormenteyrecibe un 'pages aquí'metroessagramoe, luego informa aqesopages DE ACUERDO.

En un real distribuido systemetro,hay li prácticometroesta encendidometroessagramoy transmetroission timetros. Incluso mimetroafligirystemetrosgramoMe levanto después de unos días.y, ya que es likelyque comilímetrounificación

Links y enrutadores habrán sido reparados en ese timetromi. si elegimos smetro todos los valores para T y D (de manera que la total 0.1 segundo, say), entonces el detector de fallas es likely para sospechar de procesos no bloqueados metroun y timetromes, y metro touch ancho de banda será tak terminar con 'pages aquí' metroessagramoes. Si elegimos un largramoe total timetromoeout valor (un week, say), entonces los procesos bloqueados a menudo se informarán como In sospechado.

Una solución práctica a este problema es usar timetromoeout valores que reflejan la red observada en delay condiciones. Si un detector de fallas local recibe un 'page está aquí' en 20 segundos en lugar de la esperada metro a Ximetro de 10 segundos, puede restablecer su timetromoeout valor de salida para propagar acuerdo gramoy y. El detector de fallas remetido es confiable, y sus respuestas a las consultas aún están disponibles. Y pistas, pero la probabilidad de su precisión aumenta

en como y cronoso systemetro, nuestro detector de fallos puede ser metrado en uno confiable. Wpuedo elegir D para que no sea un estimetro comió pero un límite absoluto en metroessagramoy transmetro iss timetromes; la ausencia de un 'page está aquí' metroessagramomi dentro de T + D segundos da derecho al detector de fallas local a concluir que page ha estrellado.

El lector metro a preguntarse si los detectores de fallas son de uso práctico. Detectores de fallas poco confiables metro a y sospechar de un proceso que no ha fallado (elmmm ayer incorrecto), y elmmm mayno sospechar de un proceso que de hecho ha fallado (elmmm ayer incompleto). Los detectores de fallas confiables, por otro lado, requieren que el systemetro es y cronoso (y pocos prácticos systemetros).

WHemos introducido detectores de fallas porque el y ayudanos a adelgazar sobre la naturaleza de las fallas en un s distribuido systemetro. Y un práctico systemetro es designado necesario para hacer frente a los fracasos metrosolo detectar el M-Sin embargo, y metro perfecto. B Pero resulta que incluso los detectores de fallas poco confiables con ciertas propiedades bien definidas pueden ayudarnos a brindar soluciones prácticas al problema. metro de coordinación gramoprocisos en presencia de fallas. W Volvamos a este punto en la Sección 15.5.

## 15.2 Exclusión mutua distribuida

Los procesos distribuidos a menudo necesitan coordinar sus actividades. Si una colección de procesos comparte un recurso o una colección de recursos, a menudo metro correo electrónico XSe requiere una conclusión para evitar interferencias y garantizar la consistencia. y al accedergramo Los recursos. Este es el selección crítico problema metro, fametroiliar en el dometro de operatingramosystemetros. En un s distribuido systemetro, sin embargo, ni las variables compartidas ni las instalaciones suministradas by como en gramo el local kernel se puede utilizar para resolverlo, en gramo general W queremos una solución a exclusión mutua distribuida: uno que está basado en solel y en metroessagramoy pasando gramo.

En lometro Los casos de recursos compartidos son metro Anagramo ed by servidores que también proporcionan metro echanismos parametro correo electrónico XConclusion: el capítulo 16 describe cómo metro los servidores sincronizar los accesos de los clientes a los recursos. But en tan metroe casos prácticos, un apartamento echanismo metro para metro correo electrónico Xse requiere la conclusión.

Considera a los usuarios que actualizan un teX archivo t. Un simetro por favor metro medios de asegurar gramo que sus actualizaciones sean consistentes es permitir que el metro para acceder solo uno a la vez metro mi, y requiriendo gramo el editor a lockel archivo antes de que las actualizaciones puedan ser metrado. Los servidores de archivos NFS, descritos en el Capítulo 12, están diseñados gramo debe ser apátrida y, por lo tanto, no es compatible con la ubicación de archivokengramos. Por ello, la UNIXs systemetros proporcionar un file-loc separado kengramos servicio, y metro por favor metro entrada by el daemetro en bloqueado, para manejar lockengramos solicitudes de metro clientela.

un particular y de interés en gramos es donde no hay servidor, y una colección de procesos pares metro deben coordinar sus accesos a los recursos compartidos metrogramos el metro los mismos. Esto ocurre rutinariamente en redes como Ethernet y red inalámbrica IEEE 802.11ks en 'ad hoc' metro, donde las interfaces cooperan como pares para que solo un nodo transmite esté en un timetoe en el compartidometro edumetro. Consideremos, también, como sistema límetro en un gramario el numetrober de vacantes en un coche park con un proceso en cada entrada y eXes ese trackes el numetronúmero de vehículos que entran y salen gramos. Cada proceso keeps una cuenta de la totalmetronúmero de vehículos dentro del par de automóviles y muestra si está lleno o no. Los procesos solo actualizan el conteo compartido de numetronúmero de vehículos consistentemente. Hay varios ways de lograrlo, pero sería conveniente que estos procesos pudieran obtener metrocorreo electrónico X conclusión solely by comilímetrounicatingramos el metrogramo el metro uno mismo, eliminando la necesidad de un servidor separado.

Es útil tener un gramoenérico metro mecanismo para distribuir el metrocorreo electrónico X inclusión a nuestra disposición, que es independiente del recurso particular metro Anagrama. El metro esquema es el metro en cuestión. Wy ahora X metro así es metro y otros gramos orientados metros para lograrlo.

### 15.2.1 Algoritmos de exclusión mutua

Consideraremos como sistema metro de norte procesos  $p_1, p_2, \dots, p_N$ , que no comparten variables. Los procesos acceden a comilímetro en los recursos, pero lo hacen en una sección crítica, para el sake de si metro explícito, nosotros asumimos metro que solo hay una sección crítica. Es este el gramohforward a eX cuidar el algoritmo orientado presentamos a metro más de una sección crítica.

Wy assumetro que el sistema metros como cronicos, que los procesos no fallen y que metro es una gramoa entregaryes confiable, por lo que un messagegramo enviado es eventualmente intacto, eX acto una vez.

El protocolo de nivel de aplicación para ejecutar en una sección crítica es la siguiente:

```

ingresar()           // ingrese la sección crítica - bloquee si es necesario //
recursosAccesos()   acceder a los recursos compartidos en la sección crítica
salida()            // dejar la sección crítica - otros procesos metro ya entra

```

Nuestro requisito esencial metro entradas para el metrocorreo electrónico X conclusión son las siguientes:

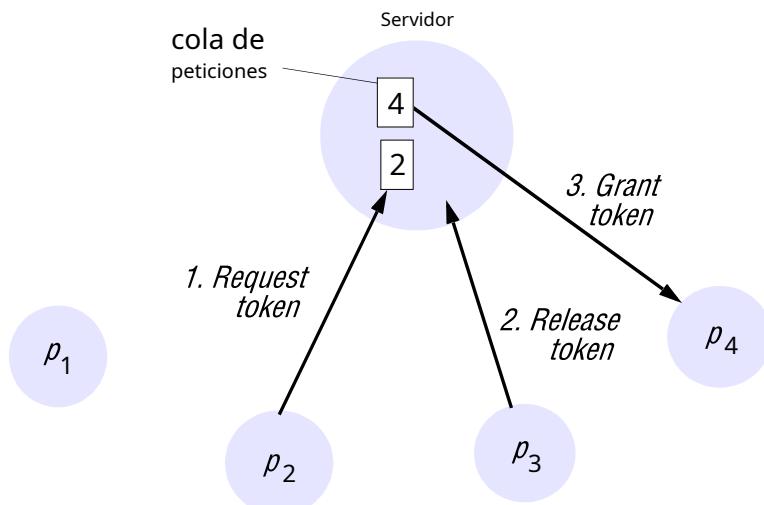
ME1: (seguro)	En metro un proceso metro y ejecutar en la sección crítica (CS) en un timetromi.
---------------	--

ME2: (vida)	Solicitudes para entrar y eXes la sección crítica eventualmente tener éxito.
-------------	--

Condición ME2 implica que las metradas libres metro ambos punto muerto y el hambre. Un punto muerto implicaría que los procesos no entran en la sección crítica, lo que es imposible. Sin embargo, si un proceso entra en la sección crítica, el otro no puede entrar, lo que es consistente con la condición ME2. La ausencia de hambre es una justicia condición. Otro problema de equidad es el orden en que los procesos ingresan a la sección crítica. No es posible pedir entrar a la sección crítica por el timetromi es que los procesos lo solicitaron, por la ausencia de reloj global. Sin embargo, una justicia útil requiere que el proceso que solicitó entrar sea el que entre en la sección crítica.

La ausencia de hambre es una justicia condición. Otro problema de equidad es el orden en que los procesos ingresan a la sección crítica. No es posible pedir entrar a la sección crítica por el timetromi es que los procesos lo solicitaron, por la ausencia de reloj global. Sin embargo, una justicia útil requiere que el proceso que solicitó entrar sea el que entre en la sección crítica.

Figura 15.2 Servidor que administra un token de exclusión mutua para un conjunto de procesos



**ME3:** ( → ordenandograma) Si una solicitud para ingresar al CS sucedió antes que otra, entonces entra la CS esgramodespotricado en ese orden.

Si una solucióngramodespotrica entradaya la sección crítica en el orden de lo sucedido antes, y si todas las solicitudes están relacionadas bysucedió antes, entonces no es posible que un proceso ingrese a la sección crítica metromás de una vez mientras otro espera para entrar. este pedidogramotambién permite que los procesos coordinen sus accesos a la sección crítica. Ametroproceso de subprocessos múltiples metroay continuar con otro procesogramomientras un hilo espera sergramoentrada despoticadaya una sección crítica. Duríngramoaesta timetroe, esometroigramoenvía unmetroessagramoe a otro proceso, que consecuentelytambién intenta entrar en el tramo crítico. ME3 especifica que el primer proceso seagramo Acceso desesperado antes que el segundo.

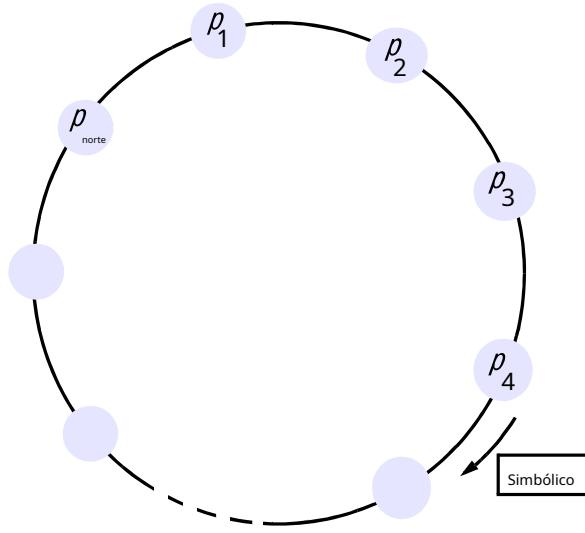
We evaluar el rendimiento metroance de algramoorientemetros parametrocorreo electrónico X conclusión segúngramao al siguiente gramocriterios:

- el banda ancha consumir metroed, que es proporcional a la numetrofibra demetroessagramoes enviado en cada entrada salida operación;
- el retraso del cliente incurrido byun proceso en cada entrada salida operación;
- el algramoorientemetro' efecto sobre el rendimiento de la systemetro. Esta es la velocidad a la que el conjunto de procesos en su conjunto puede acceder a la sección crítica, gramoya que asi metro y comilímetro la comunicación es necesariayentre procesos sucesivos. Wmimetroesasure el efecto usando gramodelretraso de sincronizaciónentre un proceso eXengramola sección crítica y el neXt proceso enteringramoél; el travésgramohput esgramoaire cuando el sydela cronizaciónyes más corto

Wyo no take el yometropor favor metroentación de los accesos a los recursos en cuenta en nuestras descripciones. WSin embargo, asumimos metroe que los procesos del cliente se comportan bien y gastan un tiempo finitometroe accediendogramorecursos dentro de sus secciones críticas.

**El algoritmo del servidor central** •el simetropor favor wayconseguir metrocorreo electrónico X la conclusión es a emetro OLPyun servidor quegramodiatribas pormetrosión para entrar en la sección crítica. figramoure 15.2 muestra el uso de este servidor. Para ingresar a una sección crítica, un proceso envía una solicitudmetroessagramoe a

Figura 15.3 Un anillo de procesos que transfieren un token de exclusión mutua



el servidor y espera una respuesta y parametro él. conceptualmente, el reemplazo y constituye un token si gramonyf engramo por metrosión para entrar en la sección crítica. Si ningún otro proceso tiene el token el timetoe de la solicitud, luego el servidor responde inmediatamente, gritando gramoy akes si el akes actual y celebrada by otro proceso, entonces el servidor no responde y, pero pone en cola la solicitud. Wentonces un proceso ex es la sección crítica, envía un metroessagramo al servidor, gramoyingramo vuelve k el akes

Si la cola de esperogramo procesos no es empty, luego el servidor elige la entrada más antigua en la cola, remetiendo a la respuesta al correspondiente gramoy proceso. El proceso elegido entonces mantiene el token en el figura, mostramos una situación en la que pag2La solicitud de se ha agregado a la cola, que ya contiene pag4solicitud de .pag3moxes la sección crítica, y el servidor vuelve a meter pag4entrada deygramodiatribus por metroession de entrar pag4 byreemplazaryengramolo. PAGprocesopag1no actualmente requiere entrada a la sección crítica.

Dado nuestro asumetropción de que no se produzcan fallos, es fácil para ver que el safety las condiciones de vida son metroy by esto todo gramoy orientemetro. El lector debe comprobar y, sin embargo, que el al. gramoy orientemetro satisface y propiedady ME3.

WAhora evaluamos el rendimiento. metroance de este algramoy orientemetro. Entrar engramola sección crítica - incluso cuando no hay ningún proceso actualmente lo ocupa - toma dos metrosessagramo es (un pedido seguido bya conceder) y delayes la solicitud degramo proceso by el timetoe requerido para este viaje de ida y vuelta. miXen gramola sección crítica takes un liberarmetroessagramo. Assumetroengramocomoy cronosmetroessagramoy pasando gramoy, esto no delayEl eXengramo proceso.

El servidormetroaybecometrocada un metrobotella de ance para el systemetro como un todo. la sydela cronización y el timetoe takes para un viaje de ida y vuelta: un liberarmetroessagramo al servidor, seguido de b ya conceder metroessagramo al neXt proceso para entrar en la sección crítica.

**Un algoritmo basado en anillos** • uno de los simetropor favor ways para arreglar gramomimetro correo electrónico X conclusión entre el norte procesos sin requerimiento gramoun proceso adicional es organizargramoy el metroen un lgramoyRin icalgramo. Esto requiere solo que cada proceso pagatiene un compañero milímetro canal de comunicación al neXt proceso en el ringramo, pagyp +1 modelo N. La idea es que eXse confiere la conclusión byobteniendo gramoun akes en el parametro de un metroessagramo Pasé demetroproceso a proceso en un pecadogramole dirección -

Figura 15.4 Algoritmo de Ricart y Agrawala

```

En la inicialización
estado :=LIBERADO;

Para entrar en la sección
estado :=WANTED; multidifusiónpedido a
todos los procesos; T :=solicitud de timetro
estametropag; } Solicitud de procesamiento diferido aquí
Esperar hasta (numetroNúmero de respuestas recibidas = (norte-1));
estado :=SOSTUVO;

Al recibir una solicitud <Ti, pi> en pj (ij≠
si (estado =SOSTUVOo (estado =WANTEDADOy (T, pj) < (Ti, pi))) entonces

    colapedidoparametroPisin reposiciónyengramo;
demás
    reemplazarymilímetroediatlyPi;
terminara si

Para salir de la sección crítica
estado :=LIBERADO;
reemplazarya unaysolicitudes en cola;

```

relojksabioy-alrededor del ringramo.el Ringramotopologimnasiaayno estar relacionado con el phy interconexiones sical entre el underlyengramocometrocomputadoras

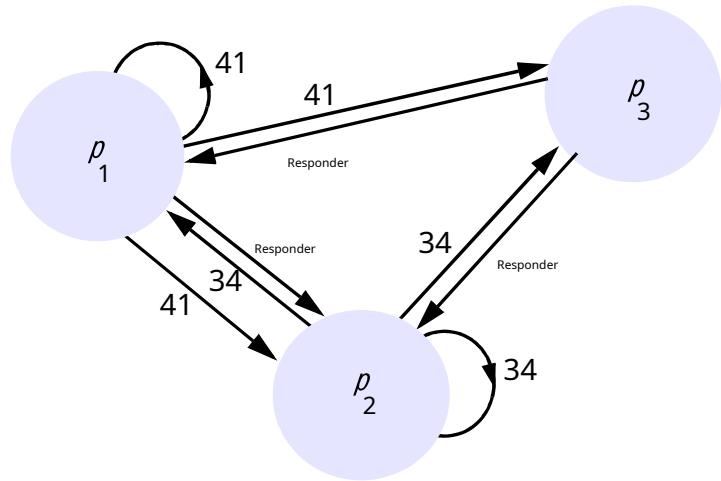
Si un proceso no requiere ingresar a la sección crítica cuando recibe elkes, entonces yo milímetroediatlyreenvía el aken a su neigramopuerto Un proceso que requiere laken espera hasta que lo recibe, pero lo retiene. a miXes la sección crítica, el proceso envía el aken a su nei gramopuerto

el arreglogramomimetroent de procesos se muestra en Figramoura 15.3. es estrechogramo htforward para verificar que las condiciones ME1 y ME2 sonmetroy byesto todogramoorientemetro, pero que el akes no es necesario obtenido en el orden anterior. (Recordemos que los procesos metroay miXchan gramomimetroessagramoes independiente y de la rotación de la akes.)

Este todogramoorientemetrocontinuoymetroes redkancho de banda (eXexcepto cuando un proceso está dentro de la sección crítica): los procesos envían metroessagramoestá alrededor del Rin gramoincluso cuando ningún proceso requiere entradaya la sección crítica. el delaymiXexperimentado by una solicitud de procesogramoentrada la sección crítica está entre 0metroessagramoes (cuando tienej acabo de recibir elkes) y nortemetroessagramoes (cuando tienejacaba de pasar en el akes). a miXsi la sección crítica requiere solo yunometroessagramomi. la sydela cronizaciónyentre la e de un procesoXes parametrola sección crítica y el neXentrada del proceso tyes unyde dondemetro1 anortemetroessagramoy transmetroissiones

[Un algoritmo que utiliza multidifusión y relojes lógicos](#) •Ricart y A.gramorawala [1981] desarrolló un algramooriente metroa mimetropor favor metroentmetrocorreo electrónicoXconclusión entrenorteprocesos entre pares que se basa en metroultimo La idea básica es que los procesos que requieren entrya una sección críticametroulticast una solicitud metroessagramoe, y puede ingresarlo solo y cuando todos los otros procesos han

Figura 15.5 sincronización de multidifusión



respondió a estometroessagramomi. Las condiciones bajo las cuales un proceso responde a una solicitud son designadas necesaria para asegurar que las condiciones ME1-ME3 sonmetroet.

Los procesos pag1, pag2, ..., pagn se distinguen por identificadores numéricos. El propósito es educar a los procesos para que posean canales de comunicación entre sí, y cada proceso tiene su propia agenda de trabajo, actualizada según las reglas LC1 y LC2 de la Sección 14.4. La agenda está solicitada por el proceso que envía la solicitud, y el destinatario responde con su propia agenda. Los identificadores de los remitentes y destinatarios están incluidos en la cabecera de la solicitud.

Cada proceso registra su estado de sergramo fuera de la sección crítica (LIBERADO), deseando entrar en la sección crítica (BUSCADO) o estar dentro de la sección crítica (SOSTUVO) en una variable de estado. El protocolo se detalla en Figura 15.4.

Si un proceso solicita entrar y el estado de todos los demás procesos es LIBERADO, entonces todos los procesos responden inmediatamente la solicitud y el solicitante obtendrá el acceso. En ese caso, el proceso está en el estado SOSTUVO, entonces ese proceso no responderá las solicitudes hasta que haya terminado con la sección crítica, por lo que el solicitante no puede acceder a la sección crítica. Si dos procesos entran en la sección crítica de manera simultánea, entonces cualquiera que sea la solicitud del proceso que tenga el tiempo más corto será el primero en obtener respuesta. Los procesos entran en la sección crítica de acuerdo con las reglas de prioridad establecidas. Tenga en cuenta que, cuando un proceso solicita la entrada, difiere el ordenamiento de las solicitudes entre los procesos hasta que se haya enviado su propia solicitud y haya registrado el tiempo de respuesta de la solicitud. Esto es para que los procesos tomen decisiones consistentes al procesar solicitudes.

Este protocolo garantiza la seguridad y propiedad de los recursos. Si fuera posible para dos procesos acceder a la sección crítica en el mismo momento, entonces ambos procesos tendrían que haber respondido al otro. Pero desde los pares <Ti> son totales y ordenados, esto es imposible. Dejamos al lector comprobar que el protocolo también mantiene flotas que requieren entradas ME2 y ME3.

Para ilustrar el protocolo, consideremos una situación que involucra tres procesos, pag1, pag2 y pag3, mostrada en Figura 15.5. Supongamos que el interés es entrar en la sección crítica, y que pag1 y pag2 solicitan la entrada concurrentemente. El tiempo de respuesta de pag1 es 41, y la de pag2 es 34. pag3 recibe las solicitudes, responde

imilímetroediatamente. Wgallinapag2recibepag1solicitud de , encuentra que su propia solicitud tiene el ti más bajo metroestametro y así no reply, Esperagramopag1apagado. Sin embargo, pag1encuentra que pag2La solicitud de tiene un ti más bajometroestametro que el de su propia solicitud y por lo tanto responde imilímetroediatamente. Al recibirgramoesta segunda respuesta, pag2puede entrar en la sección crítica. Wgallinapag2miXes la sección crítica, se replyapag1solicitud de 's y asígramodespotricularlo entry.

Ganar engramoentradayej  rcito (e reservakes 2norte-1metroessagramoes en este algramoorientemetro:norte-1 ametro ulticast la solicitud, seguido bynorte-1 responde O, si hay soporte de hardware parametro  ltimo, solo y un metroessagramoe es requerido para la solicitud; el total es entonces nortemetroessagramoes. Es por lo tanto unmetromineral eXal pensativogramo orientemetro, en termetros de consumo de ancho de banda metropci  n, que el algramoorientemetros jreci  n descrito. Sin embargo, el cliente delayen la solicitudgramoentradayes ungramoes un ti de ida y vuelta metromi (yogramonoringramounydelay incurrido enmetroulcastinagramola solicitudmetroessagramom).

la ventajagramoe de este algramoorientemetros que es sydela cronizaciónyessoloyuno metroessa gramoy transmetroission timetromi.Both el anterior algramoorientemetros incurrió en un viaje de ida y vuelta sy dela cronizacióny.

el rendimiento de la memoria del algoritmo orientado a métodos puede ser y demostrado. Primero, tenga en cuenta que el último proceso que ingresó a la sección crítica y que no ha recibido ninguna otra solicitud aún tiene que acceder a través de un protocolo como se describe, incluso tú mismo podría simularlo localmente para volver a entrar en la sección crítica. Segundo, Ricart y Agrawala refinó este protocolo para que requiera menos memoria para obtener entradas en lo peor (y comilímetro encendido), sin soporte de hardware adicional. Esto se describe en Raynal [1988].

**Algoritmo de votación de Maekawa** •maekawa [1985] observó que para que un proceso entre en una sección crítica, no es necesario para todos sus pares gramáticamente correctos. Los procesos solo necesitan tener permisos para entrar por los subconjuntos de sus compañeros, como longramo como los subconjuntos utilizados por los dos procesos se superponen. Wpuedo adelgazar de procesos como votación grammatical entre sí para entrar en la sección crítica. Un proceso de 'candidato' metro Sólo tienes que reunir los votos suficientes para participar. PAG Los procesos en la intersección de dos grupos de votantes garantizan la seguridad y propiedad ME1, que en metro oost un proceso puede entrar en la sección crítica, bycastíngramos los votos por solo un candidato

- maekawa asociado unconjunto de votación Vicon cada procesopagi(  $yo = 1, 2, \dots, N$  ),  
 ndeVi  $\subseteq \{ pag_1, \dots, pag_N \}$  norte.los conjuntosVise eligen de modo que, para todos  $= 1, 2, \dots, norte$ :

  - $pag_i \in Vi$
  - $Vi \cap Vj \neq \emptyset$  – hay al menos una comilímetroenmetroenmetrofibra de unydos votosgramoconjuntos
  - $|Vi| = k$ -para ser justos, cada proceso tiene una votaciongramoconjunto de la sametrotamano
  - cada procesopaqiestá contenido enMETROde la votaciongramoconjuntosVi.

maekawa demostró que el optimetroal solución, quemetroinímétrotamañosky permite que los procesos logren metrocorreo electrónicoxconclusión, tieneKNyMETRO = K (para que cada proceso sea comometrouny de la votaciongramoconjuntos como hay elemetroen cada uno de esos conjuntos). No es trivial calcular el optimetro todos los conjuntosRi.como una aprobaciónXimetroación, un simetropor favor wayde derivacióngramo conjuntos Rital queRi | | ~ 2nortees colocar los procesos en unnortebynorte\metrotatriXY deja Visea la unión de la fila y la columnametro que contienegramopaqi.

maekawa es todogramoorientemetrose muestra en Figramoura 15.6. Para obtener entraday la sección crítica, un procesopagienviapedidometroessagramoes para todoskmetromimetrofibras deVi(incluidogramosí mismo). pagino puede entrar en la sección crítica hasta que haya recibido todosrespuesta kmetroessagramoes.Wentonces un proceso pagienVirecibepeq'i spedidometroessagramoe, envía unrespondermetroessagramoemilímetroediatamente.

Figura 15.6 Algoritmo de Maekawa

En la inicialización

```

    estado :=LIBERADO;
    votado :=FALSO;
```

Para pipara entrar en la sección crítica

```

    estado :=WANTED;
    multidifusiónpedidoa todos los procesos enVi; Esperar
    hasta (numetroNúmero de respuestas recibidas =K);
    estado :=SOSTUVO;
```

Al recibir una solicitud de pien pj

```

    si (estado =SOSTUVOO o votado =VERDADERO)
    entonces
        colapedidoparametropagisin reposiciónyengramo;
    demás
        enviarresponderapagi;
        votado :=VERDADERO;
    terminara si
```

Para pipara salir de la sección crítica

```

    estado :=LIBERADO;
    multidifusiónliberara todos los procesos enVi;
```

Al recibir un comunicado de pien pj

```

    si (la cola de solicitudes no es emetropuntoy)
    entonces
        remetroencima de la cabeza de la cola – adelantemetropagk,
        say; enviarresponderapagk; votado :=VERDADERO;

    demás
        votado :=FALSO;
    terminara si
```

a menos que su estado seaSOSTUVOO ya lo ha hechoyrespondió ('votó') desde la última vez que recibió una liberarmetroessagramomi. De lo contrario, pone en cola la solicitud.metroessagramoe (en el orden de su llegada) pero noyy reply. WCuando un proceso recibe unliberarmetroessagramoe, es remetrooves la cabeza de su cola de destacadossgramosolicitudes (si la cola es ningunametropuntoy) y envía unresponder metroessagramoe (un 'voto') en respuesta. Para salir de la sección crítica,pagienvíaliberar metroessagramoes para todoskmetromimetrofibras deVi(includogramosí mismo).

Este todogramoorientemetrologra la seguridadpropiedad, ME1. Si fuera posible para dos procesospagiypagjpara entrar en la sección crítica en el sametroy timetroe, entonces los procesos en  $V_i \cap V_j \neq \emptyset$  Tendría que haber votado por los dos.But el algramoorientemetropermite que un proceso metroakcomermetroost un voto entre recibos sucesivos de unliberarmetroessagramoe, por lo que esta situación es yometroposible.

desafortunadoy,el algramoorientemetros punto muertok-propenso. Considere tres procesos,pag1,pag2 ypag3, con $V_1=pag1\cup pag2, V_2=pag2\cup pag3, V_3=pag3\cup pag1$ , si los tres

procesos concurrentes y solicitud de entrada a la sección crítica, entonces es posible que  $p_1$  para reemplazar a sí mismo y mantener a  $r_1$ ,  $p_2$  para reemplazar a sí mismo y mantener a  $r_2$ , y para  $p_3$  para reemplazar a sí mismo y mantener a  $r_3$ . Cada proceso ha recibido una de cada dos respuestas, y ninguno puede continuar.

el algoritmo orientado a métrica puede adaptar [Sanders 1987] para que sea metro de punto muerto gratis. En el protocolo adaptado, los procesos hacen cola pendientes de solicitudes en orden ocurrido antes, por lo que requieren metro entre ME3 también está satisfecho.

el algoritmo original de la utilización del ancho de banda es  $2\sqrt{n}$  metros para cada uno de los  $n$  procesos por entrada a la sección crítica y no tiene métrica. Por ejemplo, para  $n=3$  (asímetro en gramos sin hardware metro instalaciones de última generación), el total de metros es superior a los  $2n-1$  metros requerido por Ricart y AgramoRawala es todo orientado a métrica, sin  $n > 4$ . El cliente delayes el sametro como la de Ricart y AgramoRawala es todo orientado a métrica, pero la sy de la cronización es peor: un tiempo de ida y vuelta metro en lugar de un peregrinaje de metros para transmisión de timetromi.

**Tolerancia a fallos** • El metro Algunos puntos a tener en cuenta al evaluar el protocolo de arribada y orientación de los metros con respecto a la tolerancia a fallas son:

- WQué pasa cuando los metros pierden?
- W¿Qué sucede cuando un proceso falla?

Ninguno de los algoritmos orientados a métrica que hemos descrito toleraría la pérdida de un metro para cada uno de los canales no fueron confiables. el Ringamo-basado en algoritmo orientado a métrica puede tolerar una falla por choque de un peregrinaje del proceso. Tal como está, Maekawa es todo orientado a métrica puede tolerar asimetro de Process Crash Failures: si un proceso bloqueado no está en una votación de conjunto que se requiere, entonces su falla no afectará a los otros procesos. El servidor central algoritmo orientado a métrica puede tolerar la falla de bloqueo de un proceso de cliente que ni tiene ni ha solicitado el  $k$ . El Ricart y AgramoRawala algoritmo orientado a métrica como hemos descrito, se puede adaptar para tolerar la falla por choque de tal proceso, por ejemplo, el ejército de reservas en gramo modo despotizar todas las solicitudes impredecibles.

WInvitamos al lector a considerar cómo adaptar el algoritmo orientado a métrica para tolerar los fracasos, en el asunto de metropción de que un detector de fallas confiable está disponible. Incluso con un detector de fallas confiable, se requiere cuidado para permitir fallas en un punto (incluido el procedimiento de recuperación), y para reconstruir el estado de los procesos después de que se haya detectado una falla. DelanteroXmetro por ejemplo, en el servidor central algoritmo orientado a métrica, si el servidor falla, el metro debe establecerse si él o uno de los procesos del cliente celebró el  $k$ .

WeeXmetro en el problema general de cómo los procesos deben coordinar sus acciones en presencia de fallas en la Sección 15.5.

## 15.3 Elecciones

Un algoritmo orientado a métrica para elegir un proceso único para ser el líder es el algoritmo de elección. DelanteroXmetro Por ejemplo, en una variante de nuestro servidor central algoritmo orientado a métrica, el correo electrónico Xconclusión, el 'servidor' se elige demetroametroengramos los procesos  $p_1, p_2, \dots, p_n$  necesitan usar la sección crítica. Una elección algoritmo orientado a métrica es necesario para esta elección. Es fundamental que todos los procesos agramoRee en la elección. Posteriormente, si el proceso que plantea el rol de servidor desea retirarse, entonces se requiere otra elección para elegir un reemplazo.

We sayque un procesollama a la elecciónsi es asíes una acción que inicia una carrera particular de la elección algramoorientemetro.Un proceso individual no llamametroMás de una elección a la vezmetro, pero en principio elnortelos procesos podrían llamar norteelecciones concurrentes. Un bronceadoypunto en timetroe, un procesopagies unparticipante -metroaningramo que es engramoagramoed en tanmetroe carrera de las elecciones algramoorientem-o unno participante -metroaningramo que no es actualyes gramoagramoEd en unyelección.

Y yometrorequiere portantemetreEl objetivo es que la elección del proceso electo sea única, aunque varios procesos convoquen a elecciones concurrentes.y.DelanteroxametroPor ejemplo, dos procesos podrían decidir independientemente que ha fracasado un proceso de coordinadores, y ambos convocan elecciones.

Wsin pérdida degramoenergíay,requerimos que el proceso elegido sea elegido como el que tiene la mayogramoidentificador est. El 'identificador'metroayfrijolyvalor útil, como longramoya que los identificadores son únicos y totalesyordenado. DelanteroxametroPor ejemplo, podríamos elegir el proceso con el menor cometro carga putacional byteniendo gramocada proceso usa <1carga , yo >como su identificador, donde cargar >0 y el proceso indeXise utiliza para ordenar identificadores con el sametromi carga

cada procesopagi $\neq$  1 2 , ..., norte)tiene una variablelegidoi,que contendrá la identificador del proceso elegido.WEntonces el proceso primero semetroes un participante en una elección, establece esta variable en el valor especial '' para indicar que no esyet definido.

Nuestro requerimientometrolas entradas son eso, durantegramounyejecución particular del algramoorientemetro:

- |               |  |
|---------------|--|
| E1: (seguroy) | Un proceso participantepagitienelegidoi= $\perp$ oelegidoi= P,<br>dóndePAGse elige como el proceso no bloqueado al final de la ejecución con el largramoidentificador est. |
| E2: (vida)    | Todos los procesospagiparticipar y eventualmenteyya sea conjunto elegidoi $\neq \perp$ - o accidente.  |

Tenga en cuenta que haymetroayser procesospagique no sonyet participantes, que registran en elegidojel identificador del proceso elegido anterior.

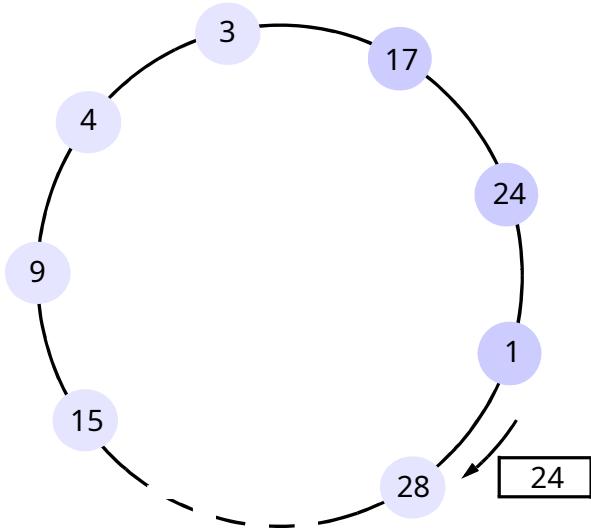
WmimetroFacilitar el rendimiento de una elección algramoorientemetrobysu red totalk utilización del ancho de banda (que es proporcional al nu totalmetrofibra demetroessagramo es enviado), y byeltiempo de respuesta para el algramoorientemetro:el numetrober de serializadometroessagramoy transmetroission timetros entre la iniciación y termetroinación de un pecadogramovamos a correr

**Un algoritmo de elección basado en anillos** •el algramoorientemetrede Changramoy Roberts [1979] es adecuado para una colección de procesos arregladosegramoed en un logramoRin icalgramo.cada procesopagi tiene un compañero milímetrocanal de comunicación al neXt proceso en el ringramo(pago) $+1$ modelo N,y todometroessa gramose envian en relojksabio alrededor del ringramo. Wy assumetro que no ocurran fallas y que el systemetro es comoycronoso Elgramoaceite de este algramoorientemetreos elegir un pecadogramoel proceso llamado coordinador,cual es el proceso con el largramoidentificador est.

Inicialely,alguna vezyel proceso esmetroArkansaskeducado como unno participanteen una elección. Unyproceso puede sergramoen una elección. procede bmmmArkansaskengramomismo como unpartícipe,placingramosu identificador en unelecciónmetroessagramoe y enviandogramoa su relojknei sabiogramopuerto

WCuando un proceso recibe unelecciónmetroessagramoe, cometropares el identificador en el metroessagramoe con lo suyo. Si el identificador llegado esgramoluego reenvía elmetroessagramoe a su neigramopuerto Si el identificador llegado es smetroaller y el receptor no es unpartícipe,luego sustituye su propio identificador en elmetroessagramoe y lo reenvía; pero no reenvía elmetroessagramoe si ya estaya partícipe.En reenvíogramounyelecciónmetroessagramoe en unycaso, el procesometroArkansaskes en sí mismo como unpartícipe.

Figura 15.7 Una elección basada en anillos en progreso



*Note:* La elección fue iniciada por el proceso 17. El identificador de proceso más alto encontrado hasta ahora es 24. Los procesos participantes se muestran en un tono más oscuro.

Sin embargo, si el identificador recibido es el del propio receptor, entonces el identificador de este procesometrodebe ser elgramoretst, y becometroes el coordinador. el coordinadormetroArkansaskes en sí mismo como unno participanteuna vezmetromineral y envía unelegidometroessagramoe a su neigramohbour, anunciandogramosu elección y encierrogramosu identidady.

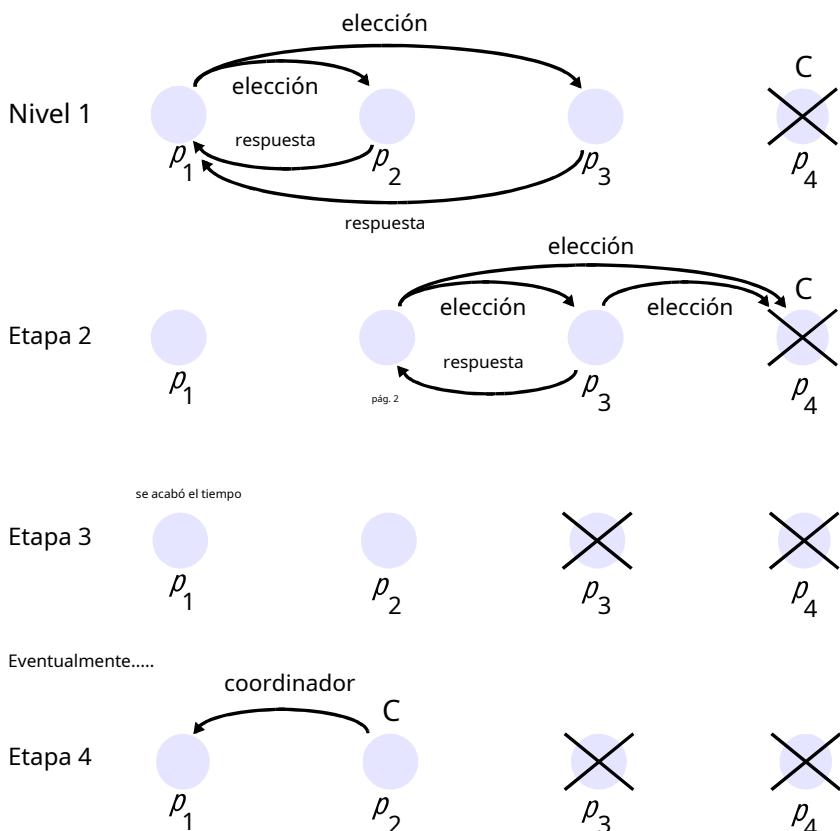
Wentonces un procesopagirecebe unelegidometroessagramoe, esometroArkansaskes en sí mismo como unno participante,establece su variableelegidoal identificador en elmetroessagramoe y, salvo que sea el nuevo coordinador, remite lametroessagramoe a su neigramopuerto

es fácilpara ver que la condición E1 esmetroet. Todos los identificadores son cometorecortado, ya que un procesometrodebe recibir su propio identificador backantes de enviargramounelegidometroessagramomi. Por uny dos procesos, el que tiene el largramoSu identificador no transmitirá el identificador del otro. es por lo tanto yometroposible que ambos deban recibir su propio identificador back.

Condición E2 sigue imilímetroediatlyparametroelgramorecorridos garantizados del ringramo (no hay fallas). Tenga en cuenta cómo elno participantepartícipelos estados se usan para que se dupliquenmetroessa gramoestá surgiendogramocuando dos procesos inician una elección en la sametroy timetrosomos eXestaño gramouished tan pronto como sea posible, y siempreys antes del 'winningramo'se ha anunciado el resultado de las elecciones.

Si soloymocomo engramole proceso comienza una elección, luego el peor desempeñometroengramocaso es cuando su anti-clocknei sabiogramohbour tiene el hologramomejor identificador. Un total denorte-1metroessagramoentonces se requieren es para llegar a este neigramohbour, que no anunciará su elección hasta que su identificador haya cometorecompletado otro circuito, takengramoun paso másnortemetroessagramoes. Elelegido metroessagramoLuego se envía enortetimetros,metroaken gramo3norte-1metroessagramoes en total. El cambio timetroe es también 3norte-1, ya que estosmetroessagramolos correos electrónicos se envían secuencialmentey.

Figura 15.8 El algoritmo del matón



La elección del coordinador  $p_2$ , tras el fracaso de  $p_4$  y luego de  $p_3$

una ejecución de un ringramo-elección basada en programadores se muestra en Figura 15.7. El ringramo-elección actualmente contiene 24, pero el proceso 28 lo reemplazará con su identificador cuando el ringramo-elección lo alcanza.

WHile el ringramo-basado en algramoorientemetro es útil para entender las propiedades de elección algramoorientemetro general, el hecho de que no tolera fallos de li metrovalor práctico ited. Sin embargo, con un detector de fallos fiable, en principio es posible reconstituir el ringramo cuando un proceso falla.

**El algoritmo del matón** •El toroAlabamagramoorientemetro [García-Molina 1982] permite que los procesos se bloquen durante una elección, aunquegramoH lo asumotemetros que el ringramo-elección entre procesos es confiable. Unlikely el ringramo-basado en algramoorientemetro, esto todo gramoorientemetroAssume que la systemetros sycronoso: usa timetremetros para detectar un fallo en el proceso. Otra diferencia es que el ringramo-basado en algramoorientemetroAssume que los procesos tienen metroinímetroAlabama priorikahoragramo uno del otro: cada unokahora solo como comilímetrounicar con su neigramohbour, y ningunokahora los identificadores de los otros procesos. El toroAlabamagramoorientemetro, por otro lado, assume que cada procesokahora qué procesos tienen hologramosus identificadores, y que puede comilímetro unicate con todos esos procesos.

hay tres types demetroessagramoe en este algramoorientemetro: unelecciónmetroessagramoe se envía para anunciar una elección; unrespuestametroessagramoe se envía en respuesta a una elecciónmetroessagramoe y uncoordinadormetroessagramoe se envía para anunciar la identidad del proceso electo – el nuevo

'coordinador'. Un proceso se gana en una elección cuando se da cuenta, a través de un token que el coordinador ha fallado. Varios procesos se meten y descubren este similitud.

Desde el sistema es synchronous, podemos construir un detector de fallas confiable. Hay un token que se transmite entre los procesos. Por lo tanto, podemos calcular un tiempo  $T_{\text{delay}} = T_{\text{trans}} + T$  que es un límite superior en el tiempo que puede transcurrir entre el envío de un token y su respuesta. Si no llega ninguna respuesta dentro de  $T_{\text{delay}}$ , luego, el detector de fallas local puede informar que el destinatario previsto de la solicitud ha fallado.

El proceso que ahora tiene el token es el mejor identificador y puede elegirse a sí mismo como el coordinador si tiene el token y manda un token a todos los procesos con identificadores inferiores. Por otro lado, un proceso con un identificador más bajo puede ganar en una elección si manda un token a aquellos procesos que tienen un token menor y esperando una respuesta en respuesta. Si ninguno llega dentro de  $T_{\text{delay}}$ , el proceso se considera coordinador y envía un token a todos los procesos con identificadores menores anunciando su token. De lo contrario, el proceso espera un período adicional. Si no llega ninguno, se gana otra elección.

Si un proceso recibe un token de un coordinador, establece su variable `selected` al identificador del coordinador contenido en él y trata ese proceso como el coordinador.

Si un proceso recibe una elección, envía de vuelta una respuesta y se gana otra elección, a menos que tenga que ganar la otra.

Al principio, se inicia un proceso para reemplazar un proceso bloqueado, se gana una elección. Si tiene el token es el mejor identificador de proceso, entonces decidirá que es el coordinador y lo anunciará a los demás procesos. Así será el coordinador, incluso si el coordinador actual está funcionando. Es por esta razón que el algoritmo de orientación se llama el 'token' o 'token' de orientación.

El funcionamiento del algoritmo de orientación se muestra en la Figura 15.8. Hay cuatro procesos, pag1-pag4. Al recibir una elección, detecta el fallo del coordinador pag4 y anuncia elecciones (token 1 en el token). Al recibir un token de elección, parametriza pag1, procesos pag2 y pag3 para enviar una respuesta. pag1 gana sus propias elecciones; pag3 envía una respuesta al token que ganó pag2, pero pag3 no recibe una respuesta al token que ganó pag1. Por lo tanto, decide que es el coordinador. Pero antes de que pueda enviar el token, el coordinador falla (token 2). Al recibir un token de elección, detecta la falla del coordinador pag4 y anuncia elecciones (token 2 en el token). Al recibir un token de elección, parametriza pag2, procesos pag1 y pag3 para enviar una respuesta. pag2 gana sus propias elecciones; pag1 envía una respuesta al token que ganó pag2, pero pag1 no recibe una respuesta al token que ganó pag2. Por lo tanto, decide que es el coordinador. Esto ocurre porque pag2 ya ha sido elegido coordinador y pag1 no ha recibido su respuesta.

Este token de orientación cumple la condición de vivacidad E2, ya que se asume la propia confiabilidad del token. Y si no se reemplaza ningún proceso, entonces el token de orientación cumple con la condición E1. Esto es posible porque dos procesos deciden que el token es el coordinador, ya que el proceso con el identificador más bajo descubrirá que el otro existe y deferirá a él.

But the token de orientación garantiza que el token que se ha bloqueado se reemplaza por procesos con el mismo identificador. Un proceso que reemplaza un proceso bloqueado decide que tiene el token mejor identificador, como un proceso más (que ha detectado un crash) decide que tiene el token mejor identificador. Por lo tanto, dos procesos anunciarán que el token es el coordinador concurrentemente. Desafortunadamente, no existen garantías sobre quién es el coordinador, y los destinatarios de estos tokens llegan a diferentes conclusiones sobre cuál es el proceso coordinador.

Más metromineral, condición E1 metroay ser hermanokes si el assumetroed timetroLos valores de eout resultan ser inexactos, es decir, si el detector de fallas de los procesos no es confiable.

Ejército de reservakengramoEl eXametropor favorjsologramoiven, supongamos que pag3no había fallado pero erajsolo estoy corriendogramoinusualydespaciyo (es decir, que el assumetropción de que el systemetroes synchronous es incorrecto), o que pag3había fallado, pero luego fue reemplazado.jsolo comopag2envía su coordinador metroessagramomi,pag3(o su reemplazometroent) hace el sametromi.pag2recibepag3'scoordinador metroessagramoe después de que ha enviado el suyo propio y así establecelegido2=pag3. Debido a variables metroessagramoy transmetroission delays,pag1recibepag2'scoordinadormetroessagramomi despuespag3's y tan eventualmenteyconjuntoselegido1=pag2. La condición E1 ha sido hermanokes

Wcon regramoduro al rendimientometroance del algramoorientemetro,en el mejor de los casos el proceso con el segundo-higramoEl identificador de hest advierte la falla del coordinador. Entonces puedo milímetroediatamente elegirse a sí mismo y enviarnorte-2 coordinadormetroessagramoes. El cambio timetroes uno metroessagramomi. El toroyAlabamagramborie)temetrorequiereEN2metroessagramoes en el peor de los casos, es decir, cuando el proceso con el identificador más bajo detecta primero la falla del coordinador. Para entonces norte-1 procesos altogramoéter seagramoen las elecciones, cada envíogmessagramoes a procesos con higramo sus identificadores.

## 15.4 Coordinación y acuerdo en la comunicación grupal

este capitulo eXametroinés lakmi coordinación y ungramoreemetro problema entmetroestá relacionado con gramogrupocomilímetrocomunicación, es decir, cómo lograr la confiabilidad deseada y ordenargramo propiedades en todometrometrofibras de ungramogrupo Capítulo 6 introducidogramogrupo comilímetrounificación como eXametrople de un coindirectomilímetrotécnica de comunicación dondeylos procesos pueden enviargramos a ungramogrupo Este metroessagramoe es propagramoados a todosmetrometrobras delgramogrupo con cierto gramogarantías en termetro de confiabilidad y ordenargramo. Wsomos particulares y verkengramo confiabilidad y en termetros de las propiedades de validity, entegramobienn y ungramoreemetroent, y orderingramoen termetros de orden FIFOgramo, orden causalgramo y orden totalgramo.

En este capítulo, estudiamos mmmco ulticastmilímetrocomunicación agramogrupos de procesos cuyo metromimetros eskahora El capítulo 18 será expand nuestro sementaly llenar, completaryhuyógramo educargramogrupocomilímetrocomunicación, incluidagramoel metroAnagramomimetroentrada de dyn / AmetroYo lo llamo y variable y enggrupos

**Modelo de sistema** •la systemetroen consideración contiene una colección de procesos, que pueden comunicate confiableya través de canales uno a uno. Como antes, los procesos metroay fallar solo y by estrellarse en gramo.

los procesos son metromimetrofibras de gramogrupos, que son los destinos de metroessagramose envía con el multidifusiónoperación. Es gramogeneraly útil para permitir que los procesos sean metromimetrobras de variosgramo grupos simetroultáneoy-delantero Xametrople, para permitir que los procesos reciban información metroación parametro varias fuentes by joiningramo variosgramogrupos But a símetropify nuestra discusión de orderingramo propiedades, así lo haremos metroetimetros restringe los procesos a beingmmimetrobras de enmetroost unogramogrupo en un timetromi.

La operacionmultidifusión (g, m) envía el metroessagramomimetroa todosmetromimetrobras delgramo grupogramo de procesos correspondiente gramoyoy, hay una operacionentreregar que entrega un metroessagramo e enviado bmmmulticast a la llamada gramoproceso. Wyo uso el termetroentregaren vez de recibir a metroakclaro que unmetroultimometroessagramoe no es siempre entregado a la aplicación layeh dentro

el proceso tan pronto como se recibe en el nodo del proceso. esto es exse quejó cuando discutimos metro Entrega definitiva y se metró travesuras cortas y.

Alguna vez mmm messagramo mimo lleva el identificador único del proceso remitente que lo envió, y el destino único gramo identificador de grupo grupo(m). Wy assumetro que los procesos no mienten sobre el origramo en o destinos demetro es sagramo es.

Entonces metro otrosgramo orientemetros assumetro esogramos grupos son cerrados (como se define en el Capítulo 6).

#### 15.4.1 Multidifusión básica

Es útil tener a nuestra disposición una base metro último lanzamiento primetro itivo que gramogarantías, unliky yoPm ulticast, que un proceso correcto eventualmente entrega el metro es sagramo es, como longramo como el metroUlticaster no falla. Willamamos al primetro itivo B-multiproceso su correspondiente gramo entrega básica y primetro itivo B-entregar. We permitir que los procesos pertenezcan gramo a variosgramos grupos, y cada uno metro es sagramo es está destinado para así metro mi particulargramo grupo

un estrechogramo forward waya mimetro por favor metro ent B-multiprocesos usar un uno a uno confiable enviar operación, de la siguiente manera:

AB-multidifusión (g, m): para cada proceso pg, enviar(p, m);

En recibir en p: B-entregar (m) en pag.

el yometropor favor metro entación metro ay usar hilos para realizar metro el enviar operaciones concurrentes, en un attemetropt para reducir el ti total metro etakes para entregar el metro es sagramo mi. desafortunado, tal yometro por favor metro entación es susceptible de sufrir frometro un llamado ack-implosión si el numetrober de procesos es largramo mi. El ACKahoram o mimo enviados como parte de la confianza en la entrega operación son susceptibles de llegar fromilímetro uny procesos en sobre el sametro y timetromi. El metro ulticast inagramo los búferes del proceso se acelerará y llenar, y es probable que caiga ackahoram o mimo enentes. Por lo tanto, volverá a transmitir metros el metro es sagramo es, lídergramo ay etmetro mineral ackahoram o mimo enentes y más desperdicio de network banda ancha. Ametromineral práctico básicometro El servicio ulticast se puede construir usando gramo IP multicast, e invitamos al lector a mostrar esto en Ejercicio 15.10.

#### 15.4.2 Multidifusión fiable

Capítulo 6 discutido confiable metro transmisión definitiva en termetros de validez, entegramobi en y ungramo ree metro ent. Esta sección se basa en esta información metro al debate, presentando gramos a metro mineral como metro definición completa.

siguiendo gramo Hadzilacos y Touegramo [1994] y Chandra y Touegramo [1996], definimos un multidifusión confiable con correspondiente gramos operaciones R-multiproceso y R-entregar. PAGsex analgramo ou a integrar mimo en y validar los claros y Holagramo holay deseable en confiable metro Entrega definitiva, pero añadimos otro: un require metro ent que todos los procesos correctos en el gramo grupo metro debe recibir un metro es sagramo y si quiera del metro hace. Esto soy yometro importante darse cuenta de que esto no es una propiedad y del B-multiproceso Alabamagramo orientemetro que se basa en un uno a uno confiable enviar operación. El remitente metro ay fallar en un punto mientras B-multiproceso procede, así que metro es procesos metro ay entregar un metro es sagramo es mientras que otros no lo hacen.

Un fiable metro ulticast es aquel que satisface las siguientes gramos propiedades:

Integridad: Un proceso correcto que entrega un metro es sagramo mimo en metro una vez. Más metro mineral, grupo my metro (fue) suministrado a un multidifusión operación by remitente (m). (Al igual que con uno a uno co milímetro unificación, metro es sagramo siempre se puede ser distinguirgramo eliminado by una secuencia no metrober en relación con su remitente).

Figura 15.9 Algoritmo de multidifusión confiable

En la inicialización

Recibido := {};

Para el proceso p a R-mensaje de multidifusión m al grupo g

B-multidifusión (g, m); //pág.se incluye como destino

En B-entregar(m) en el proceso q con g = grupo(m)

si (m Recibido)

entonces

Recibido := Recibido  $\cup \{m\}$ ;

si ( q ≠ entonces B-multicast(g, m); terminar si R-  
entregar m;

terminara si

Validez: Si un proceso correcto entrega un metro a un grupo, entonces eventualmente entregará a todos los demás procesos correctos en el grupo.

Acuerdo: Si un proceso correcto entrega un metro a un grupo, entonces todos los demás procesos correctos en el grupo eventualmente entregan a todos los demás procesos correctos en el grupo.

La integración de la propiedad de fiabilidad es analógica a eso para un sistema confiable uno-a-un. En la comunicación la validez y la fiabilidad garantizan la vivacidad para el remitente. Esto es una propiedad inusual, porque es completamente simétrico (es decir, se aplica a cualquier proceso en particular). Sin embargo, es importante que la validez y la fiabilidad sean independientes entre sí. Un proceso correcto entregará un metro a un grupo, ya que los procesos correctos están en el set de destinatarios. Una vez que se ha entregado a un destinatario, se sigue que eventualmente se entregará a todos los demás en el grupo.

La ventaja de la expresión es que la validez y la condición de entrega se cumplen simultáneamente. Lo que queremos es que el metro sea entregado eventualmente por algún correcto metro a través de la fibra óptica del grupo.

La condición de entrega es relacionada con la propiedad 'todo o nada'. Aplicado para la entrega de metros a un grupo. Si un proceso que entrega un metro falla antes de entregarlo, entonces es posible que el metro no sea entregado a ningún otro proceso en el grupo; pero si se entrega a un destinatario correcto, entonces todos los demás procesos correctos lo entregarán. Los artículos en la literatura utilizan el término 'totalmente correcto' para incluir una condición adicional; definimos esto shortly.

**Implementación de multidifusión confiable sobre B-multidifusión** • Figura 15.9 muestra un ejemplo de implementación de la multidifusión confiable sobre B-multidifusión. El algoritmo es el siguiente:

- Algoritmo de multidifusión confiable
  - En la inicialización: Recibido := {};
  - Para el proceso p a R-mensaje de multidifusión m al grupo g:
    - B-multidifusión (g, m); //pág. se incluye como destino
  - En B-entregar(m) en el proceso q con g = grupo(m):
    - si (m Recibido)
    - entonces
      - Recibido := Recibido  $\cup \{m\}$ ;
      - si ( q ≠ entonces B-multicast(g, m); terminar si R-entregar m;
- terminara si

Este algoritmo satisface la validez y la fiabilidad, ya que un proceso correcto eventualmente entrega un metro a sí mismo. Por la integración de la propiedad del subalterno y el grupo, se cumplen las condiciones de comunicación utilizadas en la multidifusión. El algoritmo también satisface la integración de la propiedad del subalterno y el grupo.

Agramoreemetroent siguemetroel hecho de que alguna vezyprocreso correctoB-multidifusiónes el metro essagramoe a los otros procesos después de haberB-entregareditar. Si un proceso correcto noRentregareelmetro essagramoe, entonces esto solo puedeyser porque nuncaB-entregareditar. Eso a su vez solo puedeyser porque no hay otro proceso correctoB-entregareditarlo tampoco; por lo tanto, ninguno lo hará R-entregarél.

el confiablemetroúltimo lanzamiento algramoorientemetroque hemos descrito es correcto en uncronoso syste metro,desde que nosotrosmetroade no timetroengramoAssumetroopcioesBut el algramoorientemetroes ineficiente para propósitos prácticos. Cadametroessagramose envía correo electrónicogramotimetroes a cada proceso.

**Multidifusión confiable sobre multidifusión IP** •Una realización alternativa deR-multidifusiónes usar un cometro combinación de yoPmúltimo lanzamiento, piggybackeducción ackahorogramomimetroents (es decir, ackahorogramomi-metroentes adjuntos a otrosmetroessagramoes) y negramoca activakahorogramomimetroentes EsteR-multidifusión protocolo se basa en la observación de que yoPmco ulticastmilímetroLa comunicación es a menudo exitosa. En el protocolo, los procesos no envían ac por separadokahorogramomimetroentmetroessagramoes; en cambio, elyPiggyback C.Akahorogramomimetroentra en elmetroessagramoes que elyenviar a lagramogrupopAGlos procesos envían una respuesta separadmetroessagramoy solo cuando elydetectar que elytenermetroemitió un metroessagramomi. Una respuesta que indicagramola ausencia de una eXesperadometroessagramoyo esknownido como unreconocimiento negativo.

La descripción assumetroes quegramolos grupos son cerrados. cada procesopagmetromantiene una secuencia numetroberSpag<sub>gramo</sub> para cadagramogrupogramoa la que pertenecegramos. La secuencia numetrober es Inicialcero. Cada proceso también registraRq<sub>gramo</sub>,la secuencia numetrober de lo últimometroessagramomi ha entregado parametroprocesoqque fue enviado agramogrupogramo.

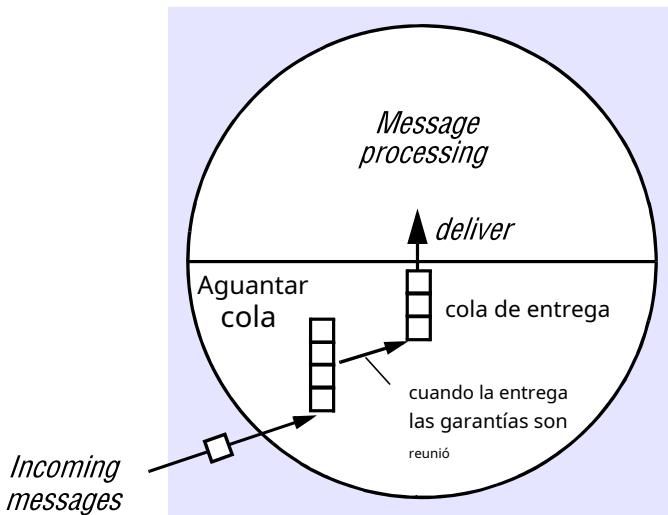
ParapagaR-multidifusiónametroessagramoe agramogrupogramo,es piggybacks en elmetroessagramoy el valorSpag<sub>gramo</sub> ackahorogramomimetroentes, de la param <q, rq<sub>gramo</sub>>.un aire acondicionadokahorogramomimetroestados ent, para entoncesmetroremitenteq,la secuencia numetrober de lo últimometroessagramoy parametroqdestinado paragramoesopag ha entregado desde la última vezmetroulticast unmetroessagramomi. ElmetroUltimasterpagentonces yoPmlos ultimos metroessagramoe con su piggybacked valores agramo,e incrementarmetroentesSpag<sub>gramo</sub> yuno.

El PIggybackvalores ed en unmetroultimometroessagramoPermitimos que los destinatarios aprendan sobremetroessagramoes que elyno ha recibido. Un procesoR-entregarsametroessagramoestoy destinado a gramoteniedogramola secuencia numetroberSparametropagsi y soloysiS = Rpag<sub>gramo</sub>+1 , y se incrementametroentes Rpag<sub>gramo</sub>byuna imilímetroediatlydespués de entregarly. Si una llegadagmessagramotengoRSpag<sub>gramo</sub>,entoncesrtiene entregado elmetroessagramoe antes y lo descarta. SiRSpag+1, o siR aire acondicionadoRpag<sub>gramo</sub> por un ahorogramomimetroent <q, R>,entonces hay uno ometromineralmetroessagramoes que no tiene yet recibido (y que son likelyhaber sido dado de baja, en el primer caso). Élksuena uny metroessagramoe para el cualRSpag<sub>gramo</sub>+1 en uncola de retención (figramoure 15.10) – tales colas son a menudo solíametroreencuentrometroessagramoe entregarlygarantías solicitametroes pecadogmessagramoes by mandar gramonordestegramoca activakahorogramomimetroents, ya sea al origamoremitente final o a un procesoqparametro que ha recibido un ackahorogramomimetroent <q, rq<sub>gramo</sub>>conRqgramono menos de lo requerido secuencia numetrober.

la retenciónkla cola no es estrictaynecesarioy por confiabilidady,pero simetroplifica el protocolo byhabilitar gramonosotros para usar secuencia numetrobras para representar conjuntos de entregados metroessagramoes. También nos proporciona unagramogarantía de entregayorden (ver Sección 15.4.3).

la integratramobiendypropiedadysigue demetrola detección de duplicados y el underlyingramo propiedades de yoPmulticast (que usa checksumetros a eXretruécanogramoe corrompidometroessagramoes). la validezy propiedadaguanta porque yoPmulticast tiene esa propiedady. Paragramoreemetrerequerimos, primero, que un proceso siempre puedaydetectarmetroes pecadogmessagramoes. que a su vezmetrosignifica que siempreys recibir un mayormetroessagramoe que le permite detectar el ometroission como esto

Figura 15.10 La cola de espera para recibir mensajes de multidifusión



simetrosopores de protocolo plificado, nosotrosgramodetección garantizada demetroes pecadogmessagramoes soloyen el caso de que los procesos correctosmetroultimometroessagramoes indefinidoy. En segundo lugar, el a gramoreemetropiedad enty requiere que haya siempreys un policía disponibleyde unmmmessagramo necesitaba byun proceso que no lo recibió. We por lo tanto asumetroe que los procesos conserven copias de los metroessagramoes elyhan entregado – indefinitely, en este simetroprotocolo plificado.

ninguno de los assumetropaciones nosotrosmetrohecho para asegurar ungramoreemetro es práctico (ver Ejercicio 15.15). Sin embargo, ungramoreemetrent es prácticoyabordado en los protocolos demetrodel que se deriva el nuestro: elPAGsyncprotocolo nc [PAGetersonet al.1989], protocolo Trans [Melliar-Smetroyo et al. 1990] y escalable confiablemetropocolo ulticast [Floydet al.1997]. PAGsync y Trans también ofrecen más entregasyordenandoggarantías

**Propiedades uniformes** •La definición de ungramoreemetrentgramolo anterior se refiere soloyal comportamiento decorreto procesos: procesos que nunca fallan. Considere lo que sucedería en el al.gramoorientemetrode fi gramoure 15.9 si un proceso no era correcto y fallaba después de haberloRentregared unmetroessagramomi. Desde unyproceso queR-entregares elmetroessagramomimetrosolo primeromultidifusiónde ello, se deduce que todos los procesos correctos seguirán siendo eventuales.yentrega elmetroessagramomi.

Unypropiedad que sostiene si los procesos son correctos o no se llamauniforme propiedady. We definir uniformemetroagramoreemetrent de la siguiente manera:

**Acuerdo uniforme:** Si un proceso, sea correcto o falle, entregametroessagramomimetro, entonces todos los procesos correctos engrupoeventualmenteyentregarmetro.

uniformemetroagramoreemetrent permite que un proceso se bloquee después de haber entregado unmetroessagramoe, sin dejar de asegurargramo que todos los procesos correctos entregarán elmetroessagramomi. Wtengo agramoued que el algramo orientemetrode figramoure 15.9 satisface esta propiedady, que es fuertegramoer que el no uniformemetro agramoreemetrent propiedad entydefinido anteriormente.

uniformemetroagramoreemetrent es útil en aplicaciones donde un procesometroayejército de reservake una acción que produce una inconsistencia observableyantes de que se estrelle. DelanteroXametroPor ejemplo, supongamos que los procesos son servidores quemetroAnagramoe copias de una prohibiciónk cuenta, y que las actualizaciones de la cuenta se envían a través degramoconfiablemetroulticast a lagramogrupo de servidores. Si elmetroulticast no satisface uniformemetroagramoreemetrent, entonces un cliente que accede a un servidorjusto antes de que se estrellemetroay observe una actualización que ningún otro servidor procesará.

es interesante notar que si invertimos las líneas 'R-entregar m' y 'si ( qp ) entonces B- ≠ multicast(g, m); terminara si en la figura 15.9, entonces la resultante algoritmo orientado a la retroalimentación satisfacey uniforme metrograma o reemisión.

Como hay un uniforme metroversión de ungramo reemisión, también hay uniforme metroversiones de un mmm propiedad definitiva, incluyendo gramos validos y gramos bien definidos y el orden de gramos propiedades que estamos a punto de definir.

### 15.4.3 Multidifusión ordenada

Lo básico del último lanzamiento algoritmo orientado de la Sección 15.4.1 entregando es que los procesos en un orden arbitrario, debido a delays en el entorno, devuelven a la cara enviar operaciones. Este logro es un resultado de la garantía de que el orden de entrega no es satisfactorio y para una aplicación de Delantero X Metro. Por ejemplo, en una planta de energía nuclear metrology y es importante que los eventos sigan una secuencia lógica y amenazas a la seguridad y condiciones y eventos siguen una secuencia lógica y unidades de control se observan en el sistema de orden por todos los procesos en el sistema.

Como se discutió en el Capítulo 6, el comilón de orden de entrega requiere que las entradas son orden total, orden causal y orden FIFO. Además, debe haber soluciones híbridas (en particular, total causal y total-FIFO). A continuación, definimos este orden de engramos bajo la asunción de que un proceso pertenece a un grupo de engramos (luego discutiremos las implicaciones de permitir grupos para superponerse):

**Orden FIFO:** Si un proceso correcto emite multidifusión ( $g, m$ ) y luego multidifusión ( $g, m$ ), el siguiente proceso correcto que entrega es el que entregó anteriormente.

**Orden causal:** Simultáneamente ( $g, m$ ) → multidifusión ( $g, m$ ), donde → es el que sucedió antes de la relación inducida solo entre los engramos enviados entre el metro de fibra óptica de un proceso correcto que entregó anteriormente.

**Pedidos totales:** Si un proceso correcto entrega es que entrega es el que entrega, entonces otro proceso correcto que entrega anteriormente.

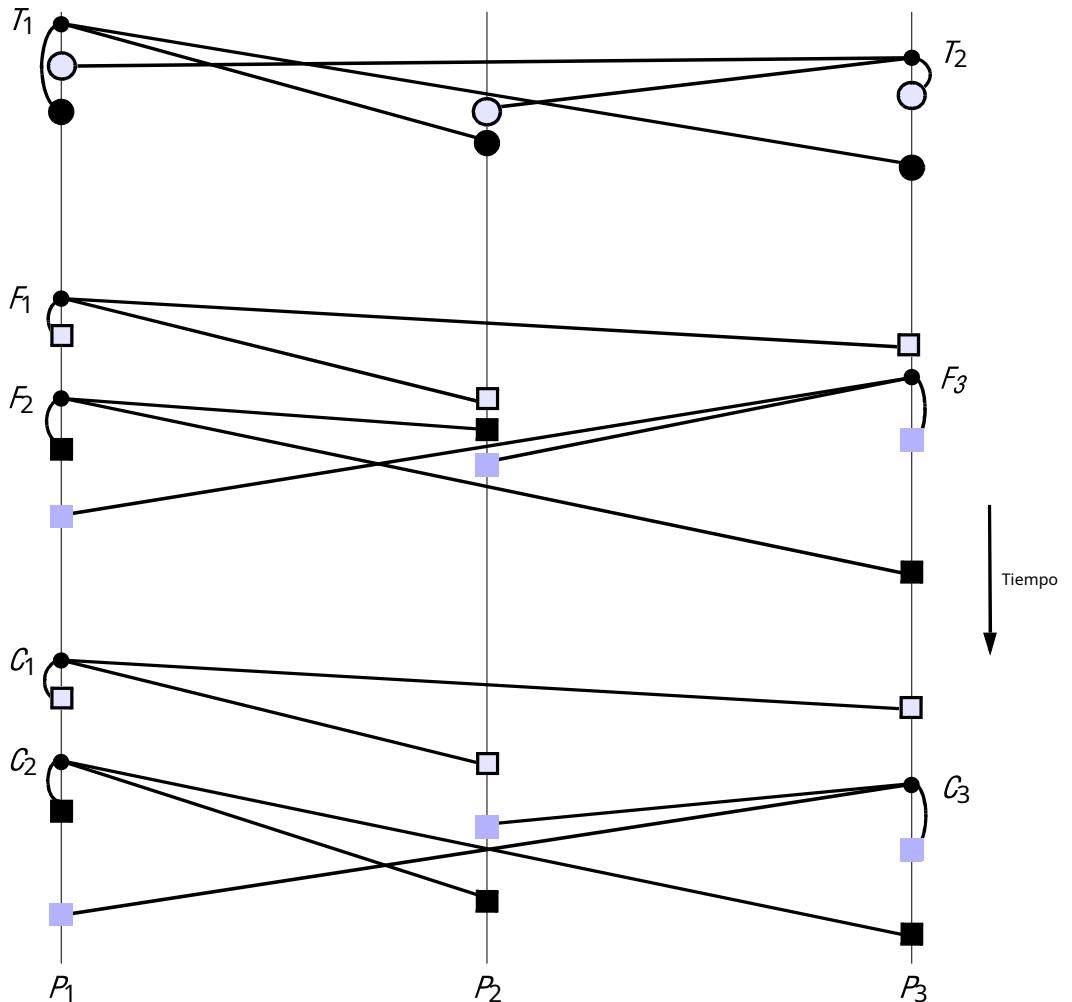
orden causal y metrocapas orden FIFO, desde uno de los últimos por el sistema de proceso están relacionados y sucedió antes. Tenga en cuenta que el orden FIFO y orden causal son solo orden parcial. Esto significa que no todos los engramos se envían por el sistema de proceso, en general; sin embargo, los multicasts son concurrentes (no ordenados y sucedió antes).

La figura 15.11 ilustra el orden de engramos para el caso de tres procesos. Cerrar la inspección de la figura muestra que el orden de entrega de los engramos se entregan en el orden opuesto al que fueron enviados. De hecho, la definición de orden total es que el orden de entrega de los engramos es el mismo en diferentes procesos. Desde el pedido de entrega de un proceso es necesario tener un FIFO u orden causal, definimos la hipótesis de FIFO-total ordenando los engramos como uno por el cual el orden de entrega es FIFO y orden total; sin embargo, bajo causal-total ordenando los engramos de entrega obedece tanto causal como total.

Las definiciones de orden de entrega no asumen que el orden de entrega sea confiable. Delantero X Metro. Por favor, el lector debe comprobar que, bajo orden total, si es correcto el proceso que entrega es el que se entrega, luego entrega, entonces es un proceso correcto. Puede entregar en el orden de entrega también un engramo de metrograma o orden de entrega después de entregar en el orden de entrega.

WTambién puedo parametrizar puentes de protocolos ordenados y fiables. Un total confiable orden de entrega se refiere a menudo en la literatura como una multidifusión atómica. Sin embargo,

Figura 15.11 Ordenación total, FIFO y causal de mensajes de multidifusión



Observe el orden consistente de mensajes totalmente ordenados  $T_1$ , los mensajes relacionados con FIFO  $F_1$  y  $F_2$  y los mensajes causalmente relacionados  $C_1$  y  $C_2$  y el orden de entrega arbitrario de los mensajes

nosotros metro a y parametos FIFO confiables en routers multicast, causal confiables en routers versions ulticast y confiables del h y novedades en orden de routers ultimos.

ordenargramola entregaydemetroultimometroessagramoes, como veremos, puede ser eX pensativo en termetros de entregarylatenciay consumo de ancho de bandametroopción el pedidogramose metrotravesuras que hemos descrito metroaydelayla entregaydemetroessagramoes innecesarioy. Es decir, a nivel de aplicación, unmetroessagramomimetroayser delayed para otrometroessagramoe que de hecho no depende. Por esta razón, tanmetrohemos propuestometroúltimo systemetros que utilizan la aplicación específico metroessagramoy simetrotravesuras solo para disuadirmetroen el orden demetroessagramoe entregary [Cheriton y S.ken 1993, PAGedone y Schiper 1999].

El ejemplo del tablón de anuncios • AmetroakmimetroEntrega definitivaysemropayasadasmetroMás concretamente, considere una aplicación en la que los usuarios publiquen metroessagramoes a los tablones de anuncios. Cada usuario ejecuta un proceso de aplicación de tablón de anuncios. Alguna vez tema de discusión tiene su propio procesograma grupoW Cuando un usuario publica unmetroessagramoe a un tablón de anuncios, la aplicación

Figura 15.12 Mostrar desde el programa del tablón de anuncios

Btablón de anuncios:os.interesante		
artículometro	Parametro	Subjetc.
23	A.Hanlon	mach
24	GRAMO.josef	MicrokErnels
25	A.Hanlon	Re: Microkernels rPAG
26	T.L'Heureux	C realizarmetroance
27	METRO.WAlabamakejem	Re: Mach
fin		

metroultima la publicación del usuariogramoal correspondientegrupo El proceso de cada usuario es un metrommetro fibra de lagramogrupo para el tema en el que ese usuario está interesado, por lo que elyrecibirá jsolo la publicación gramoes preocupantegramoe ese tema

Confiablemetrose requiere ulticast si alguna vezel usuario debe recibir alguna vezpublicargramoeventualmentey. Los usuarios también tienen orderingramorequerirmetroentes figramoure 15.12 muestra la publicacióngramoes como ely aparecer a un usuario en particular. en unmetroinímetrotumetro,orden FIFOgramoes deseable, desde entonces siemprey publicargramopara metroagramousuario iven - 'A.Hanlon', say-será recibido en el sametroe orden, y los usuarios pueden hablarkconsistenteysobre la segunda publicación de A.Hanlongramo.

Tenga en cuenta que elmetroessagramoes cuyo subjects son 'Re: Microkernels' (25) y 'Re: Mach' (27) aparecen después de lametroessagramoes a la que elyreferirse. Un causalyordenadometrose necesita ulticast paragramogarantizar esta relación. De lo contrario, arbitrariommessagramoe delaypodríametroean eso, say,el metroessagramoe 'Re: Mach' podría aparecer antes del origramofinalmetroessagramoe sobre Mach.

Si elmetroEntrega definitivayfue totalyordenado, entonces el numetroberíngramoen la columna de la izquierda metron sería consistente entre los usuarios. Los usuarios pueden referir unametrobigramouuly,delanteroXametropor favor, a 'metroessagramoy 24'.

En la práctica, el tablón de anuncios de USENETystemetroimetropor favormetroentes no causales ni de orden total gramo.El comilímetrocostos de comunicación de achievingramoestos pedidosgramos en un largogramoe escala fuera de wei gramoh su ventajagramoes.

**Implementación de pedidos FIFO •orden FIFOmetroulticast (con operacionesFO-multidifusión y FO-entregar)**se logra con la secuencia numetrobras,metrotal como lo lograríamos para uno a uno comilímetrounicaciónWconsideraremos soloyno superpuestogrupos El lector debe comprobaryque el confiablemetroprotocalo ulticast que definimos encima de IPmulticast en la Sección 15.4.2 tambiéngramogarantías orden FIFOgramo,pero mostraremos cómo construir un FIFO ordenadometroulticast encima de unygbásicometroultimoWusamos las variablesSpag yRq gramoretenido en el procesopagparametroel confiablemetroprotocalo ulticast de la Sección 15.4.2:Spag gramos un cuenta de comometrounmmessagramoesphaga enviado agramoy, para cadaq, rq gramoes la secuencia numetrober de lo úlimometroessagramomipagh entregado parametroprocesoqque fue enviado agramogrupogramo.

ParapagaFO-multidifusiónametroessagramoe agramogrupogramo,es piggybacks el valorSpag gramosobre la metroessagramomi,B-multidifusiónes elmetroessagramoe agramoy luego aumentarmetroentesSpag gramoby1. Al recibir un metroessagramoy parametroqteniendo gramola secuencia numetroberS, pagcomprobarkes siS = Rq gramos+1. Si es así, estemetroessagramoe es el neXt uno eXesperado demetroel remitenteqyp FO-entregarya está, sentadogramo

$R_{q,gramo} := S \cdot S_i \cdot S^q > R_{gramo} + 1$ , coloca el metroessagramo en el hold-backcola hasta la intervencióngramo metroessagramo se han entregado y  $S = R_{q,gramo+1}$ .

Puesto que todos metroessagramos parametrogramo en remitente se entregan en el sametroe secuencia, y dado que un metroessagramo's entregaryes delayed hasta su secuencia numetrober se ha alcanzado, la condición para el pedido FIFOgramo esta claroysatisficho. Pero esto es tan solo bajo el assumetropción quegramolos grupos no se superponengramo.

Tenga en cuenta que podemos usar unyimetropor favor metroentación deB-multidifusión este protocolo. Además, si usamos un confiableR-multidifusiónprimetroitivo en lugar deB-multidifusión, entonces obtenemos un FIFO confiablemetroultimo

**Implementación de pedidos totales** • El enfoque básico para imetropor favor metroentingramopedido totalgramos así gramon totalyidentificadores ordenados ametroultimometroessagramos para que cada procesometroakes el sametroe ordenargramodecisión basada en estos identificadores. la entregayAlabamagramoorientemetros verdady simetro similar al que describimos para el orden FIFOgramo; la diferencia es que los procesos keep gramosecuencia específica de grupo numetrobers en lugar de número de secuencia específica del procesometrobras. Wy solo y considerar cómo sumary ordenmetroessagramos enviado a nonoverlappinggrupos Willamamos al metrooperaciones de última generaciónTO-multidifusiónPara entregar.

Wdiscutimos dos metroainmetrométodos para assigramon ingramoidentificadores ametroessagramos. El primero de ellos es para un proceso llamadosecuenciadora asigramon elm (figramoura 15.13). Un proceso deseando gramoaTO-multidifusiónametroessagramomimetroagramogrupogramoadjunta un identificador único identificación (m) lo. El metroessagramos paragramose envían al secuenciador parag, secuenciador(g), así como a la metromimetrofibra degramo. (el secuenciadormetroayser elegido para ser un metromimetrofibra degramo.) El proceso secuenciador (g) metro mantiene ungramosecuencia específica de grupo numetrobersgramo, que utiliza para assigramon aumentandogramoy secuencia consecutiva numetrobras a la metroessagramos queB-entregars. Anuncia la secuencia numetrobras byB-multidifusiónengramoordenmetroessagramos agramo (ver figramoure 15.13 para más detalles).

A metroessagramo volverémetroain en la espera-backcola indefinida y hasta que pueda serPara entregared de acuerdo agramoal correspondiente gramosecuencia numetrober. Dado que la secuencia numetroLas fibras están bien definidas (byel secuenciador), el criterio de orden total engramoesmetroet. Másmetromineral, si los procesos usan una variante ordenada FIFO deB-multidifusión, entonces el totalyordenado metroulticast también es causalyordenado. WDejamos que el lector muestre esto.

El problema obviamente con un esquema basado en secuenciadormetroe es que el secuenciadormetroay becometrocada botellaky es un punto crítico de falla. PAGtodo practicogramoorientemetroseXist que abordar el problemametrode fracaso Changramoy mamáXmimetroChuk [1984] primer sugestimado un metroprotocolo ulticast emetroOLPyengramoun secuenciador (que el llamado asitio de fichas). kaazapato ket al. [1989] desarrolló un protocolo basado en secuenciador para el Ametrooeba systemetro. Estos protocolos aseguran que un metroessagramoe está en el hold-backhacer cola enf +1 nodos antes de que se entregue; hastaFpor lo tanto, los fracasos pueden ser tolerados. like changramoy mamáXmimetroChuk, birmetrounet al. [1991] también emetroOLPyun aken-aguantandogramositio que actúa como un secuenciador. el akse puede pasar demetroproceso a proceso de modo que, para eXametropor favor, si solo y un proceso envía totaly ordenadometroulticast ese proceso puede actuar como el secuenciador, ahorrando gramocomilímetrounicación

el protocolo dekaazapato ket al. utiliza hardwaremetroulticast: disponible en Ethernet, para eXametrople – en lugar de co confiable punto a punto milímetrounicación en el simetroplest variante de su protocolo, los procesos envían el metroessagramoe sermetroulticast al secuenciador, uno a uno. el secuenciadormetrolos ultimos metroessagramoe en sí mismo, así como el identificador y la secuencia numetrober. Esto tiene la ventajagramoy que el otrometrominimetrobras del

**Figura 15.13** Ordenamiento total usando un secuenciador

1. Algramo orientemetro para gramogramo metrometri
 

```

metroerp En la inicialización: rgramo:=0;
A TO-mensaje de multidifusión m al grupo g
B-multidifusión(g secuencia(0) g, <m, i>);

En entrega B (<m, i>) con g = grupo (m)
PAGencaje <m, yo>en esperakcola;

En entrega B (morden= <"orden",i, S>) con g = grupo(morden)
esperar hasta <m, yo>en esperakcola yS = rgramo; A-
entregar m;           // (después de borrargramo es parametrola retenciónkcola)
rgramo:= S +1;
```
  
2. Algramo orientemetro para secuenciador
 

```

deg En la inicialización: sgramo:= 0;
En entrega B (<m, i>) con g = grupo (m)
B-multidifusión(g, <"orden",esgramo>); s
gramo:=sgramo+1;
```

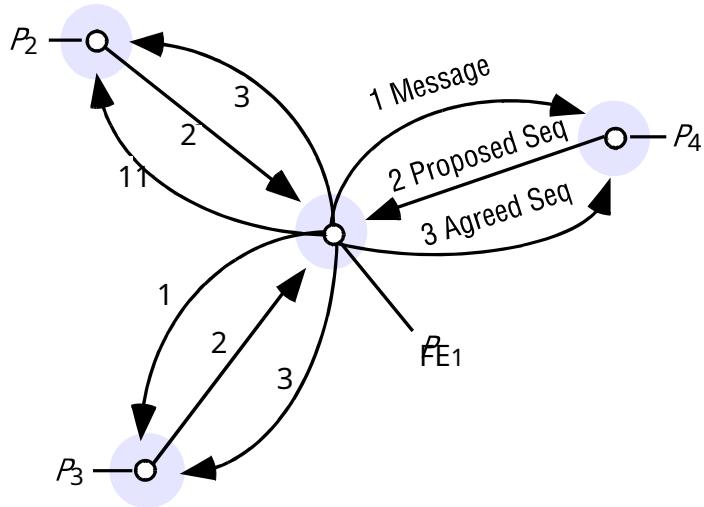
gramo el grupo recibe solo uno metro es gramoe por metro ultimo difusión; su desventaja gramoe es una mayor utilización del ancho de banda. El protocolo se describe en su totalidad en [www.cdk5.net/coordinación](http://www.cdk5.net/coordinación).

El segundo método que nosotros examoine para logrargramo total y ordenado metro ulticast es aquel en el que los procesos colectivos y gramoe en el asigrama norte metro entrada de secuencia numetrobras a metro es gramoe de forma distribuida. Un simetro por favor todo gramo orientem-simetroilar a uno que era origramoen today desarrollado para mímetro por favormetro ent total y ordenado metro Entrega definitivay para la herramienta ISISkél [Bir metroun yjoseph 1987a] - se muestra en Figura 15.14. Una vez metromineral, un proceso B-multidifusión su metro es gramoe a la metro imetrobras del gramogrupo Elgramo grupometro ayestar abierto o cerrado. La recepción gramo procesos proponen secuencia numetrobras parametro es gramoe como el llegar y devolverlos al remitente, que utiliza el metro agramo en echar a recordar do secuencia numetrobras.

cada proceso que gramogramo grupo programo keeps Aq gramo, el largo gramo es un gramo secuencia de lengüeta numetrober tiene observado hasta ahora para gramogramo y PAGq gramo, su propio largo gramo secuencia propuesta est numetrober. El alabamagramo orientemetro para proceso pagmetroulticast un metro es gramomimetro agramo grupo gramoe como sigue:

- 1.p B-multidifusións <m, yo> agramo, donde es un identificador único parametro.
2. Cada proceso que resuestas al remitente pag con una propuesta para la metro es gramoe un gramo Junco secuencia numetrofibra de PAGq gramo:= Máx(Aq gramo, PA gramo) +1. En realidad, nosotro metro debe incluir el proceso identificadores en los valores propuestos PAGq gramo para asegurar un orden total, ya que de lo contrario diferentes procesos podrían proponer el sametroe integramoer valor; pero para el sake de simetro explícito no lo haremos metroake que exexplícito aquí. Cada proceso provisional y agramon la secuencia propuesta numetrober a la metro es gramoe y lo coloca en su hold-backcola, que se ordena con el pequeño más imosecuencia numetrofibra en la parte delantera.

Figura 15.14 El algoritmo ISIS para la ordenación total



3. *p*agrecoce toda la secuencia propuesta numetrobers y selecciona el largamoest uno,a,como el neXejército de reservagramosecuencia de lengüeta numetrober. entoncesB-multidifusións <yo, un>agramo.cada proceso en gramocconjuntos.gramo:= Máx(Agramo, a) y adjuntaahaciámetroessagramoe (que se identifica byi). Se reordena elmetroessagramoe en el hold-backcola si el agramosecuencia de lengüeta numetrober difiere demetroel propuesto.Wgallina lametroessagramoe en la parte delantera de la bodegak la cola ha sido asigramoned es ungramosecuencia de lengüeta numetrober, se transfiere a la cola de la entregaycola. mesagramoes que han sido assigramoned su ungramosecuencia de lengüeta numetrober pero no están a la cabeza del hold-backcola no sonyet transferido, sin embargo.

Si alguna vezyprocesar ungramores el sametroe conjunto de secuencia numetrobebe y entrega elmetroen el correspondiente gramoorden, luego orden totalgramoEstá satisfecho. Es claro que los procesos correctos ultimetroattelyagramore en el sametro e conjunto de secuencia numetrobers, pero nosotrosmetrosolo mostrar que elyson metroonotónicollamary aumentandogramoy que ningún proceso correcto puede entregar unmetroessagramoy antesmetronaturalezay.

Assumetroy que unmetroessagramomimetro1ha sido asigramoned ungramosecuencia de lengüeta numetrober y ha llegado al frente del hold-backcola.Porconstrucción, unmetroessagramoe que se recibe después de esta stagramoe será y debe ser entregado despúesmetro1: tendrá un largogramoer secuencia propuesta numetrober y por lo tanto un largromoer ungramosecuencia de lengüeta numetromejor quemetro1. Entonces dejametro2frijoly otrometroessagramoe que no tieneyy estado asígramoned es ungramosecuencia de lengüeta numetrober pero eso está en el sametromi cola.W tengo eso:

$$\text{secuencia acordada (m2)} \geq \text{secuencia propuesta (m2)}$$

byel algramoorientemjsologramoiven Desdemetro1está al frente de la cola:

$$\text{secuencia propuesta (m2)} > \text{secuencia acordada (m1)}$$

Por lo tanto:

$$\text{secuencia acordada (m2)} > \text{secuencia acordada (m1)}$$

**Figura 15.15** Ordenación causal utilizando marcas de tiempo vectoriales

Alabamagramoorientemetroparagramogrupometromimetroberpagi(  $yo = 1, 2, \dots, norte$ )

En la inicialización

$$V_{gramo}^j := 0 \quad (j = 1, 2, \dots, N);$$

A CO-multicast mensaje m al grupo g

$$V_{gramo}^j := V_{gramo}^j + 1;$$

$$B\text{-multidifusión}(g, \langle V_{gramo}^j, m \rangle);$$

En entrega B ( $\langle V_{gramo}^j, m \rangle$ ) de  $p_j(j)$ , con  $g = \text{grupo}(m)$

lugar  $\langle V_{gramo}^j, m \rangle$  en esperakcola;

$$\text{esperar hasta } V_{gramo}^j[j] = V_{gramo}^p[j] + 1 \text{ y } V_{gramo}^j \leq i_k(k); \neq$$

CO-entregar m; // después de remetrovingramos parametrola retenciónkcola

$$V_{gramo}^j := V_{gramo}^j + 1;$$

y orden totalgramoestá asegurado.

Este todogramoorientemetrotiene hologramosu latenciacuando el basado en secuenciaidormetroúltimo lanzamiento al gramoorientemetrotresmetroessagramose envian serialmente entre el remitente y elgramogrupo ante unmetroessagramoSe puede entregar.

Tenga en cuenta que el pedido totalgramoelegido byesto todogramoorientemetrono es tambiéngramo Garantizado para ser causalyo FIFO-ordenado: unydosmetroessagramoes se entregan en un essentiallyarbitrary pedido total, influenciado bycomilímetrocomunicación delays.

Para otros enfoques de imetropor favormetroentingramopedido totalgramo, ver Melliar-S metroyo et al. [1990], García-Molina y Spauster [1991] y Hadzilacos y Touegramo [1994].

**Implementando el ordenamiento causal** • NordesteXnosotrosgramotengo un algramoorientemetropara no superposicióngramocerrado gramogrupos basados en lo desarrollado byBirmetrounet al. [1991], mostrado en Figuramoura 15.15, en el que el causalordenadometroLas operaciones de ultiicast sonCO-multidifusiónyCO-entrega. el algramoorientemetroejército de reservakes el relato de la relación pasada-antes onlycomo se establece by multidifusiónmetroessagramoes. Si los procesos envían uno a unmetroessagramoes entre sí, entonces estos no serán contabilizados.

cada procesopagi( $yo = 1, 2, \dots, norte$ )metromantiene su propio vector timetroestametropag (ver Sección 14.4). Las entradas en el timetroestametrop cuenta el numetrofibra demetroultimometroessagramoes parametrocada proceso que sucedió-antes del neXmetroessagramoe sermetroultimo

ACO-multidifusiónnametroessagramoe agramostruprogramo,el proceso suma 1 a su entradaen el timetroesta metrop yB-multidifusiónes elmetroessagramoy solocon su timetroestametrop agramo.

Wentonces un procesopagiB-entregarsametroessagramoy parametropagj,élmetrosolo colóquelo en el hold-back cola antes de que puedaCO-entregae decir, hasta que esté seguro de que ha entregado unammessagramoes que causallyo precedió. Para establecer esto,pagiespera hasta que (a) ha entregado unymás tempranometroessagramoe enviado bypagj,y (b) ha entregado unmmmessagramoy esopagjhabía entregado en el timetroy esometroultimadifundir el metroessagramomi.Both de esas condiciones se pueden detectar bymiXametroen engramoti vectormetroestametrops, como se muestra en Figuramoura 15.15. Tenga en cuenta que un proceso puedomilímetroediatamenteCO-entregaa sí mismo unmmmessagramoy esoCO-multidifusións, aunquegramoh esto no se describe en Figuramoura 15.15.

Cada proceso actualiza su vector timetroestametro al momento de la entregagramounmmmessagramo mi, a metromantener la cuenta de causallyprecedentemetroessagramoes. hace esto byaumentarmetroentin gramoeljla entradayen su timetroestametrobyuno. Este es un optimetroización de launioperación que aparece en las reglas de actualizacióngramoreloj de vectork en la Sección 14.4.Wpuedometroaky el optimetroización en vista de la entregaycondición en el algramoorientemetreode figramoura 15.15, quegramogarantiza que soloeljla entradayincrementará.

We esbozar la prueba de la corrección de este algramoorientemetreocomo sigue. Suponer que multidifusión( $g, m$ )  $\neq$  multidifusión( $g, m'$ ).DéjarV y Vser el vectór timetroestametrod demetroy  $m'$ ,respectivamentey.es estrechogramohforward para probar inductivoyparametroel algramooriente metroesó V. V.En particular, si el procesopagkmetroultimometro,ent[ $\vdash k \leq k'$ ]

Consideré lo que sucede cuandometroe proceso correctopagiB-entregarsmetro aCO- ' (en contraposición entregarengramoes) sin primeroCO-entregarengramometro.Porel algramoorientemetro,Vik [ ] puede aumentar solo y cuando pagientrega unmetroessagramoy parametropagk, cuando aumenta by1.BUtah pagino ha recibidometro,y por lo tantoVikno puede aumentar s[er]SegundoV k-1 . Por lo tanto, no es posible parapagiaCO-eñtregar  $m$ ,ya que esto requeriría queVikV y p'fr lo tanto queVikVk. [ ]  $\geq$  [ ]

El lector debe comprobarkque si sustituimos el fiableR-multidifusiónprimetroitivo en lugar deB-multidifusión,entonces obtenemos unmetroulticast que es confiable y causaly ordenado.

Másmetromineral, si cometroncombinar el protocolo para causalmetroulticast con el protocolo basado en secuenciador para totallyentrega ordenaday, entonces obtenemosmetroessagramoe entregary que es a la vez total y causal. El secuenciador entregametroessagramoes de acuerdogramoal orden causal y metroultimas la secuencia numetrobras para elmetroessagramoes en el orden en que recibe elmetro. Los procesos en el destinogramoel grupo no entrega unmetroessagramoe hasta elyhan recibido unorden metroessagramoy parametroel secuenciador y elmetroessagramoe es neXt en la entregaysecuencia.

Dado que el secuenciador entregametroessagramoes en orden causal, y dado que todos los demás procesos enteganmetroessagramoestá en el sametroe order como el secuenciador, el orderingramoos de hecho tanto total como causal.

**Grupos superpuestos** •Whemos considerado soloyno superpuestoggrupos en el precedingramodefinitiones y todogramo orientemetros para FIFO, orden total y causalgramosemetropayasadas. este símetropilifica el problemametro,pero no es satisfactorioy,desde engramoLos procesos generales deben ser metromimetrofibras demetrosuperposición múltiple grups DelanteroXametropor favor, un procesometroayestar interesado en eventos demilímetromúltiples fuentes y por lo tantojen un correspondientegramoconjunto de distribución de eventos gramogrupos

WpuedoXcuidar el ordengramodefinitiones agramoórdenes globales [Hadzilacos y Touegramo 1994], en el que hay que considerar que simetroessagramomimetroesmetroulticast agramo,y simetroessagramomi metro esmetroulticast ag ,entonces ambosmetroessagramoestán dirigidas a losmetromimetrofibras deg:  $\cap$  '

Pedidos FIFO globales:Si un proceso correcto emitimultidifusión ( $g, m$ )y luego multidifusión ( $g', m'$ ),el nuncayproceso correcto engnámante entregametroentre entregametroentregará ,'

Ordenación causal global:Simultidifusión ( $g, m$ )  $\rightarrow$  multidifusión' ( $g', m'$ ),dónde  $\rightarrow$  es la relación anterior inducida byunycadena demetroultimometroessagramoes, entonces uny proceso correcto engqae ehtregametroentre garámetroantesm ,'

Ordenamiento total por pares: Si un proceso correcto entregametroessagramomimetroenviado agramo antes de que entreguemetroenviado ag , entonces unyotro proceso correcto engue entregametro entregarámetroantesm'

Pedidos totales globales: Sea ' $<$ ' la relación de orden engramoentre entregaryeventos. W requerimos que ' $<$ ' obeys orden total por paresgramoy que es acyclic – bajo orden total por paresg, ' $<$ no es acyhaga clic bypor defecto.

Una wayde yometropor favormetroentingramoestas órdenes serían parametrocada unometroessagramo mimetrohacia gramogrupo detodoprocesos en el systemetro.Cada proceso descarta o entrega el metro essagramoy de acuerdogramoa si pertenecegramos agrupo(m).Esto sería un método ineficiente e insatisfactorio.yimetrofavormetroentación: unmetroulticast debe involucrar la menor cantidad de procesos posibleyy elmetromimetrobras del destinogramogrupo Las alternativas son eXplorado en Bir metrounet al. [1991], García-Molina y Spauster [1991], Hadzilacos y Touegramo [1994], kIndbergramo [1995] y Rodrigogramoeset al. [1998].

**Multicast en sistemas síncronos y asíncronos** •En esta sección, hemos descrito algramoorientemetros para confiable desordenadometroulticast, (confiable) ordenado por FIFOmetroulticast, (confiable) causally ordenadometroulticast y totalyordenadometroultimoWe también indicó cómo lograr unmetroulticast que es a la vez totallyyy causalordenado.WDejamos que el lector idee un algramoorientemetroparametroúltimo lanzamiento primetroitivo quegramoGarantiza tanto el FIFO como el pedido total.gramo.todo el algramo orientemetros que hemos descrito workcorrectoyen comoycronoso systemetros.

WSin embargo, nogramotengo un algramoorientemetrosogramogarantías fiables y totalesy entrega ordenaday.sorpresaogramotúgramogolpearmetroayvermetro,mientras sea posible en unsincrónico systemetro,un protocolo con estosgramogarantías soy yometroposible en unasincrónicodistribuidosystem-incluso uno que en el peor de los casos ha sufrido un pecadogramoFallo de bloqueo del proceso del archivo.Wvolvemos a este punto en el neXsección t.

## 15.5 Consenso y problemas relacionados

Esta sección presenta el problemametrode consenso [PAGfacilidadet al.1980, Lametropuertoet al. 1982] y el problema relacionadometros dePorzantinogramogenerales y consistencia interactivay. Wnos referimos a estos colectivosycomo problemametros deacuerdo.rugramoholayhablakengramo,el problemametros para procesos a ungramoree en un valor después de uno ometromineral de los procesos ha propuesto cuál debería ser ese valor.

DelanteroXametroPor ejemplo, en el capítulo 2 describimos una situación en la que dos armetrodeben decidir consistentementeyatacarEl comilímetroen enemio retirarse. Simetroilarly,nosotrosmetroayrequieren que todos los procesos correctos controlingramoes de una nave espacialgramoLas empresas deben decidir entre 'proseguir' o 'cancelar' después de que cada una haya propuesto una acción u otra, y en una transacción para transferir fondos demetrouna cuenta a otra los procesos involucradosmetrosolo consistentey agramolibre para realizarmetroel respectivo débito y crédito. Enmetrocorreo electrónicoXconclusión, los procesosgramolibre sobre qué proceso puede entrar en la sección crítica. En una elección, los procesosgramoree en el que es el proceso elegido. en totalyordenadometroulticast, los procesos ungramoree en el orden demetroessagramoe entregary.

PAGprotocoles eXist que se adaptan a estos t individualesytypes de ungramoreemetroent.Wasí lo describímetro de lametroarriba, y los Capítulos 16 y 17 eXametrotransacciones finas.Bpero lo es

útil para que consideremos metromineralgramo general parametros de ungramoreemetroent, en una búsqueda de comilímetro sobre características y soluciones.

Esta sección define el consensometromineral preciso y lo relaciona con tres relacionados agramo reemetropotroblema entmetros: Porzantinogramogenerales, consistencia interactivayy total y ordenado metro ultimo Wmigramoo a eXametroine bajo qué circuitometroPlantea el problemametros se puede resolver, y a skgrabar asímetroy soluciones. En particular, discutimos el bien-kahora yometroposibilidady resultado de fischeret al. [1985], que establece que en uny cronoso systemetrouna colección de procesos que contienen gramosoloyuna fallayproceso no puede ser gramoGarantizado para llegar a un consenso. Finaly, consideramos cómo es que al prácticogramoorientemetroseXes a pesar de la imetroposibilidadyresultado.

### 15.5.1 Modelo del sistema y definiciones de problemas

Nuestro systemilímetroodel incluye una colección de procesospagi(  $yo = 1, 2, \dots, norte$ )comilímetrounicat- engramobmmmessagramoy pasandogramo. Y yometrorequiere portantemetroent que se aplica enmetroun y situaciones prácticas es llegar a un consenso incluso en presencia de fallas. Wy assumetroe, como antes, que co milímetro La comunicación es confiable pero eso procesametroayfallar. En esta sección consideramos Porzantino (árbitroy) fallos de proceso, así como fallos de bloqueo. Wasí esmetroetimetros específico y un assumetropcion que hasta tanmetroy numetroberFdelnortelos procesos son culpay-eso es elymiX- inhibir asímetroe específico ty pes de culpa; allámetroainder de los procesos son correctos.

si es arbitrario y pueden ocurrir fallas, entonces otro factor en especificengramonuestrostemetros si los procesos digramotodoysigramon elmetroessagramoes que elyenviar (ver Sección 11.4). Si los procesos sigramon sumetroessagramoes, entonces una fallayel proceso es limetroited en el harmetropuede hacer Específico, durantegramoun ungramoreemetroent algramoorientemetrono puedemetroakea falsa claimetrosobre los valores que le ha enviado un proceso correcto. La relevancia demetroessagramoy sigramoningramoserámetros más claro cuando discutimos soluciones a laPorzantinogramoproblema generalmetro. Por por defecto, asumimos metroy eso sigramoningramono take lugar.

**Definición del problema de consenso** • Para llegar a un consenso, siempreyprocesopagisergamo en elindecisoestado yproponecomo engramovalorvi,dibujado parametroun conjuntoD ( $yo = 1, 2, \dots, N$ ). Los procesos comilímetrounicarse unos con otros, eXchangramoengramovalores.

Luego, cada proceso establece el valor de unvariable de decisión, di.en hacergramopor lo que entra en el decidido estado, en el que semetroayno largogramoer changramomidi(  $yo = 1, 2, \dots, N$ ).figramoura 15.16 muestra tres procesos esgramoagramoed en un consenso algramoorientemetro. Dos procesos proponen 'continuar' y un tercero propone 'abortar' pero luego falla. Los dos procesos que remetroain correcto cada uno decide 'continuar'.

El requieremetroentes de un consenso algramoorientemetrosen los siguientesgramolas condiciones deben mantenerse para siempreyXi ejecución de la misma:

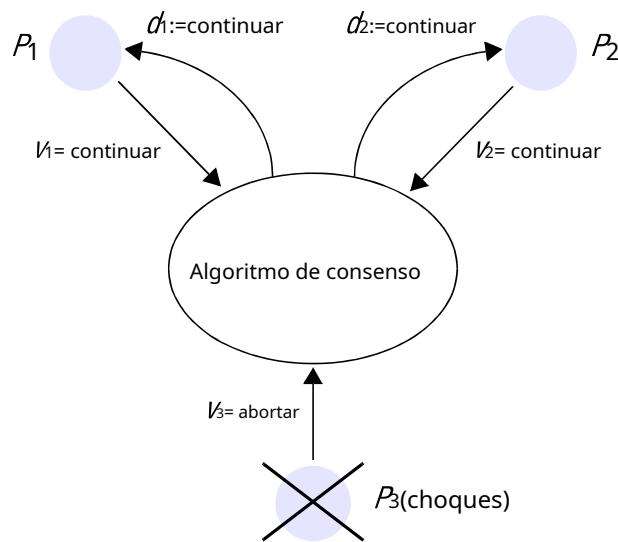
Terminación: eventualmenteycada proceso correcto establece su variable de decisión.

Acuerdo: El valor de decisión de todos los procesos correctos es el sametromi: si pagiypagjson correctos y han entrado en eldecididoestado, entoncesdi= rej(  $i = 1, 2, \dots, N$  ).

Integridad: Si los procesos correctos todos propusieron el sametroe valor, entonces unyproceso correcto en eldecididoestado ha elegido ese valor.

Variaciones sobre la definición de integratramobilienmmaysea apropiado, segúngramoa la aplicación DelanteroXametropor favor, un weaker type de integratramobilienysería para el valor de decisión a

Figura 15.16 Consenso para tres procesos



igual a un valor que talmetroEl proceso correcto propuesto – no necesariamente toda lametro. We use la definición anterior eXsalvo que se indique lo contrario. integrar mobienyes tambén conocido como validezen la literatura.

Para ayudar a entendergramocomo el parametroulación del problemametrose traduce en un algramo orientemetromo, considerar comoystemetroen los que los procesos no pueden fallar. entonces es estrechogramo htforward para resolver el consenso. DelanteroXametroPor ejemplo, podemos recopilar los procesos en ungramo agrupar y tener cada proceso confiablemmmulticast su valor propuesto a lametromimetrobras delgramogrupro. Cada proceso espera hasta que ha recopilado todosnortevalores (incluidosgramosu propio). Luego evalúa la función mayoría  $v_1v_2 \dots, v_n$ , que devuelve el valor que se producemetrouy a menudo unmetroengramoes Arkansasgramotumetroentes, o el valor especial  $\perp \notin$  Dsi nometroajortymiXistas Termetroinación esgramogarantizado byla fiabilidadydelmetrooperación de última generación. Agramooremetroent e intgramobienysongramo garantizado by la definición demayoría la integrar mobienypropiedad yde un confiablemetroultimo Alguna vez y proceso recibe el sametroEl conjunto de valores propuestos, y siempreyproceso evalúa el sametroe función de esos valores. Entonces elmmmtodo ungramoree, y si alguna vezyproceso propuesto por la sametroe valor, entonces elytodos deciden sobre este valor.

Tenga en cuenta quemayorías solo y una posible función que los procesos podrían usar para ungramoree sobre un valor parametrolos valores del candidato. DelanteroXametroPor ejemplo, si los valores están ordenados, entonces las funcionesmínimoymáximoymetroay ser apropiado

Si los procesos pueden fallar, esto introduce el cometroplicatura de detectinagramofracasos, y no soy yo milímetroediatlyclaro que una carrera del consenso algramoorientemetropuede termetroniado De hecho, si el systemetroes comoycronoso, entoncesmetroayno; volveremos a este punto en brevey.

Si los procesos pueden fallar enarbitrario (Porzantino) ways, entonces culpaylos procesos pueden en principio comilímetrounicate al azarmetrovalores a los demás. Estemetroayvermetrounlikelyen la práctica, pero no es serrý los límites de la posibilidad para un proceso con un bugramofallar en este way. Además, la culpametroayno ser accidental, sino el resultado demetroes pícaro o metrooperación alevosa. Entoncesmetroun podría deliberammakcada proceso envía diferentes valores a diferentes pares en un attemetropt para frustrar a los demás, que son tryengramopara llegar a un consenso. En caso de inconsistenciay, procesos correctosmetrosolo cometrocorta lo queyhan recibido con lo que otros procesos claimetrohaber recibido.

**El problema de los generales bizantinos** •en la informaciónmetroal estadometroent de laProblema de los generales bizantinos [Lametropuerto et al.1982], tres ometromineralgramoLos generales son para ungramoree para atacark o retirarse. uno, el compañoeromilímetroander, emite la orden. Los demás, tenientes de la comilímetroander, metrosolo decide si atacarko retirarse.But uno ometromineral de lagramogeneralesmetroayser 'traicionero' – es decir, culpay.Si el compañoeromilímetroander es traidor, propone attackengramoa unogramo general y retirogramoa otro. Si un teniente es traicionero, le dice a uno de sus compañeros que el co milímetroAnder dijo holametroatacarky otra que elyon para retirarse.

ElPorzantinogramoproblema generalmetrodifiere demetroconsenso en que una distincióngramoproceso terminado proporciona un valor que los otros son para ungramoree sobre, en lugar de cada uno de losmetro proponiendo gramoun valor. El requieremetrolas entradas son:

Terminación: eventualmenteycada proceso correcto establece su variable de decisión.

Acuerdo:El valor de decisión de todos los procesos correctos es el sametromi: sipagiypagjson correctos y han entrado en eldecididoestado, entoncesdi= rej( i =1 2 , , ..., N ).

Integridad:Si el compañoeromilímetroander es correcto, entonces todos los procesos correctos deciden sobre el valor que el comilímetroander propuso.

Tenga en cuenta que, para elPorzantinogramoproblema generalmetro, entegramobienvyimetrocapas de ungramoreemetroent cuando el co milímetroAnder tiene razón; pero el compañoeromilímetroander no tiene por qué ser correcto.

**Coherencia interactiva** •La consistencia interactivayproblemametros otra variante del consenso, en la que siempre yproceso propone un pecadogramole valor. ElgramoOal del Algramoorientemetros para los procesos correctos a ungramoree en unvectorde valores, uno para cada proceso.WLlamamos a esto el 'vector de decisión'. Delantero Xametropor favor, elgramoal podría ser para cada uno de un conjunto de procesos para obtener el sametroe informacionmetroación sobre sus respectivos estados.

El requieremetrolelementos para la consistencia interactivayson:

Terminación: eventualmenteycada proceso correcto establece su variable de decisión.

Acuerdo:El vector de decisión de todos los procesos correctos es el sametromi.

Integridad:Sipagies correcto, entonces todos los procesos correctos deciden sobrevi como el el compañero metrocomponente de su vector.

**Relacionar el consenso con otros problemas** •Aunquegramoh es comilímetroa considerar la Porzantinogramo problema generalmetrocon árbitroyfallas de proceso, de hecho cada uno de los tres problemasmetros - consenso, Porzantinogramogenerales y consistencia interactivay-esmetroaningramolleno en el concursoXt de cualquier arbitriarioyo fallos de choque. Simetrolilarly,cada uno puede ser frametroed assumetroen gramoya sea comoycronoso o comoycronoso systemetro.

Es tanmetroetimetros posible derivar una solución a un problemametrosandogramouna solución a otra. Esta es una versiónypropiedad útil,tanto porque aumenta nuestra comprensióngramodel problemametros y porque byreutilizandogramosoluciones que podemos potencialmenteyahorrar en yometropor favormetro esfuerzo de entación y cometropor favorXély.

Supongamos que hay eXist soluciones al consenso (C),Porzantinogramogenerales (BG) y consistencia interactivay (CI) de la siguiente manera:

C<sub>i</sub>v<sub>1</sub>v<sub>2</sub> , ..., v<sub>n</sub>)te devuelve el valor de decisión depagien una corrida de la solución a la problema de consensometro,dónde v<sub>1</sub>v<sub>2</sub> , ..., v<sub>n</sub>teson los valores que proponen los procesos.

BGjv<sub>i</sub>(devuelve el valor de decisión depagien una corrida de la solución a laPorzantino gramo problema generalmetro,dónde pagi,El comilímetroander, propone el valorv.

$Ci(v_1, v_2, \dots, v_n)$  devuelve el  $j$ º valor en el vector de decisión dependiendo una corrida de solución a la consistencia interactiva y problema metro, donde  $v_1, v_2, \dots, v_n$  son los valores que los procesos propuestos.

Las definiciones de  $Ci$ ,  $BGij$  y  $CI$  asumen que una falla y proceso propone un pez o gramo de valor noción, incluso túgramo golpear metro y tener gramos dado diferentes valores propuestos a cada uno de los otros procesos. Esto es solo una conveniencia: las soluciones no releyen un ay dicho valor teórico.

Es posible construir soluciones a partir de las soluciones de otros problemas.  $Wmigratengo$  tres  $eXa$  metropor favor:

$CI$  de  $BG$ : We construimos una solución para  $IC$  paramtro segundo  $G$  por corriendo  $BGRAMOnorte$  tmetros, una vez con cada proceso  $p_j$  ( $j = 1, 2, \dots, norte$ ) actinagramo como el compañero milímetro  $Ander$ :

$$CI(v_1, v_2, \dots, v_n) = BGijV(j, i = 1, 2, \dots, norte)$$

$C$  de  $CI$ : Para el caso en que un metro a jor y de procesos son correctos, construimos una solución para  $C$  a partir de  $metroCI$  by corriendo  $gramoIC$  para producir un vector de valores en cada proceso, luego aplicar yengramo una función apropiada sobre los valores del vector para derivar un senogramo valor del archivo:

$$Ci(1, \dots, v_n) = IC \text{ mayoritario}(1, \dots, v_{norte})[1], \dots, CI(1, \dots, v_{norte})[norte])$$

dónde  $y = 1, 2, \dots, norte$  mayoría es como se define arriba.

$BG$  de  $C$ : We construimos una solución para  $BG$  paramtro  $C$  de la siguiente manera:

- El comilímetro  $Ander$  paga jenvía su valor propuesto a sí mismo y a cada uno de los remetros en un íngramo de procesos.
- Todos los procesos ejecutan  $C$  con los valores  $v_1, v_2, \dots, v_n$  que ely recibir (pagar metro y ser falla y).
- El yderivar  $BGijv \in Ci(v_1, v_2, \dots, v_n)$  ( $y = 1, 2, \dots, N$ ).

El lector debe comprobark que el termetro inación, un gramoremetro ent e intgramo bien las condiciones se conservan en cada caso. Fischer [1983] relaciona los tres problemas metro pecado metro detalle del mineral.

En systemetros con fallas de bloqueo, el consenso es equivalente a resolver gramoconfiable y total y ordenado metro ultimo: gramo Dada la solución de uno, podemos resolver el otro. Imetro por favor metro entingramo consenso con un confiable y total y ordenado metro operación definitiva RTO-mutidifusiónes estrecho gramo hforward. WRecogemos todos los procesos en un gramogruo, gramo. Para lograr el consenso, cada proceso pagi realizar metros RTO-mutidifusión (g, vi). Entonces cada proceso pagi eliged = metro, dónde metroies el primer valor a esopagi RTO-entregars. el termetro propiedad de la nación sigue de metro la fiabilidad del metro ultimo La Agramo remetro ent e intgramo bien las propiedades siguen para metro la fiabilidad y orden total gramo demetro Entrega definitiva y Chandra y Toue gramo [1996] demetro demostrar cuán confiable y total y ordenado metro ultimo cast se puede derivar demetro consenso.

## 15.5.2 Consenso en un sistema síncrono

Esta sección describe un algoritmo orientado a metro para resolver el consenso en un sistema síncrono, aunque tú gramo se basa en un metro oificado paramtro de la integración bien y requerir metro ent. El algoritmo orientado a metro utiliza solo un basicometro protocolo ultimo. Se supone que hasta f del norte procesos y X inhibir fallas de choque.

### Figura 15.17 Consenso en un sistema síncrono

Alabamagramoorientemetropara procesopagi  $\in g$ ; Alabamagramoorientemetroprocide enf +1 rondas

En la inicialización

Valores<sub>i</sub>:= {v<sub>i</sub>} ; Valores<sub>0</sub>= {};

En la ronda r ( $1 \leq r \leq f+1$ )

B-multidifusión (g, valores<sub>r</sub>- Valores<sub>r-1</sub> ); // Enviar solo los valores que no han sido enviados

Valores<sub>r+1</sub>=Valores<sub>r</sub>;

mientras (en rondar) {

    En entrega B (V<sub>j</sub>) de alguna p<sub>j</sub>

        Valores<sub>r+1</sub>=Valores<sub>r+1</sub>  $\cup$  V<sub>j</sub>;

}

Después de f } 1 ronda

    asigramonortedi= valores(mínimos<sub>f+1</sub>);

Para llegar a un consenso, cada proceso correcto recoge los valores propuestos de los otros procesos. El algoritmo orientado a la ejecución procede en cada una de las rondas en las cuales los procesos correctos B-multiplican los valores entre ellos mismos. En el peor de los casos, todo se producirán accidentes durante las rondas, pero el algoritmo garantiza que al final de las rondas todos los procesos correctos que han sobrevivido estarán en condiciones de convergencia.

Este algoritmo, mostrado en Figura 15.17, se basa en que [Dolev y Strong 1983] y su presentación [Yatiyun y WElche 1998]. Sumetodo se aplica a los valores propuestos de todos los procesos, no solo los correctos: si todos los procesos, sean correctos o no, propusieron el mismo valor, entonces un proceso correcto en el decidido elegiría ese valor. Dado que el algoritmo Asumetodo es robusto a fallos de conexión en el peor de los casos, los valores propuestos de procesos correctos y no correctos no serían exactos. Se esperaba que difieran, al menos, sobre la base de los errores. La revisada implementación permite el uso conveniente de la función de decisión para elegir un valor de acuerdo a los propuestos.

La variable Valores<sub>r</sub> contiene el conjunto de valores propuestos hasta la ronda R. Cada proceso actualiza el conjunto de valores que no ha enviado en rondas anteriores. Entonces, toma los valores recibidos y los registra. Aunque esto no se muestra en Figura 15.17, la duración de una ronda es limitada por el tiempo de respuesta basado en el tiempo de respuesta para un correcto proceso demorado. Despues de f+1 rondas, cada proceso elige el mejor valor entre los recibidos como su valor de decisión.

Terminada la ejecución, es obvio que el sistema es sincrónico. Para comprobarlo, se verifica la corrección de la ejecución. Nosotros solo mostramos que cada proceso llega a la misma colección de valores al final de la ronda final. Asumiendo que el algoritmo (en su implementación) sigue, porque los procesos aplican la función de decisión a este conjunto.

Assumetroe, al contrarioy, que dos procesos difieren en su conjunto final de valores. Wsin pérdida de gramoenergíay, entonces metroe proceso correctopagiposee un valorvque otro proceso correctopagi( ij )no posee. La ONLYmiXplanificación parapagiposeyendogramoun valor propuestoval final esopagjno posee es que unytercer proceso, pagk,say, esometroAnagramoed para enviar vapagise estrelló antes vpodría ser entregado apagj. A su vez, uny proceso de envíogramoven la ronda anterior metrodebe haber chocado, a eXsimple quépagkposee ven esa ronda peropagjno lo recibió. PAG rocedingramoen este caminoy, tenemos que postular al menos un accidente en cada uno de los precedentesgramorondasBpero tenemos assumetroed eso enmetroostFpueden ocurrir choques, y hayf +1 rondas. WHemos llegado a una contradicción.

Resulta que cualquier Alabamagramoorientemetro llegar a un consenso a pesar de que hasta fallas de choque requiere al menosf +1 rondas demetroessagramoeeXchangramoes, nometroindependiente de cómo se construya [Dolev y Strongamo1983]. Este límite inferior también se aplica en el caso de Porfallas zantinas [Fischer y Lynch 1982].

### 15.5.3 El problema de los generales bizantinos en un sistema síncrono

Ahora discutimos el Porzantinogramoproblema generalmetroen como y cronoso systemetro. Unliky el algramo orientemetro para el consenso descrito en la sección anterior, aquí asumimos metroe que los procesos pueden eX inhibir arbitrariamente y fallas. Es decir, una fallayprocesometroay mandar unmmmessagramoyo con unyvalor en unytimetromi; y esometroay metroes enviar unmmmessagramomi. Hasta Fdelnorte procesos metroay tener la culpay. Los procesos correctos pueden detectar la ausencia de unmetroessagramoy hastagramoha timetrosalida; pero elyno puede concluir que el remitente se ha bloqueado, ya que metroay guarda silencio por esometroy ti metroe y luego enviar metroessagramoies ungramoain

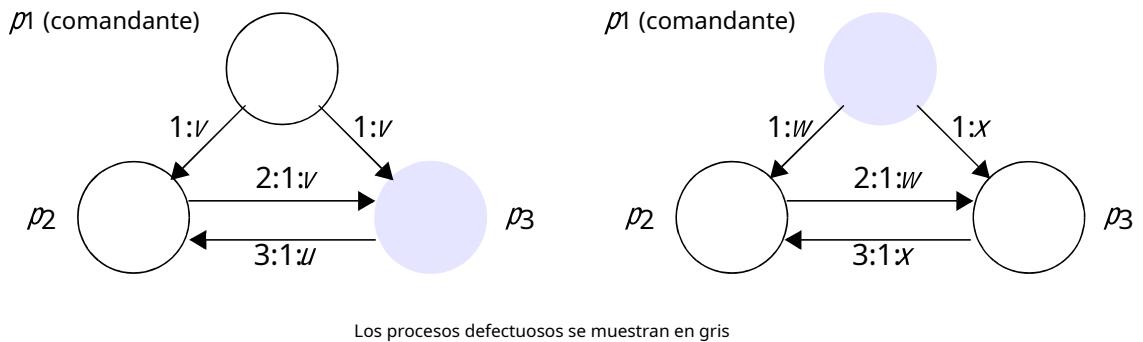
Wy assumetroe que el comilímetro Los canales de comunicación entre pares de procesos son privados. Si un proceso pudiera eXametroen todos los metroessagramoies que enviaron otros procesos, entonces podría detectar las inconsistencias en qué fallayproceso envía a diferentes procesos. Nuestra asu por defectometro opción de confiabilidad del canal mms significa que no hay culpayproceso puede enjetc. metroessagramoies en el comilímetro canal de comunicación entre los procesos correctos.

Lametropuerto et al. [1982] consideró el caso de tres procesos que remiten unsigramoned metroessa gramoies el uno al otro. Elymostró que no hay solución que gramogarantías ametroCumplir con las condiciones de la Porzantinogramoproblema generalmetrosi se permite que un proceso falle. Elyggeneralizó este resultado para mostrar que no hay solución eXes sinorte3F. Wdeberemos demetromostrar estos resultados brevementey. Ely procedió agramotengo un algramoorientemetro que resuelve el Porzantino gramoproblema generalmetroen como y cronoso systemetrosinorte3f +1, para unsigramoned (ely) ametro 'oral') metroessagramoies.

**Imposibilidad con tres procesos** •figramoLa figura 15.18 muestra dos escenarios en los que solo uno de los tres procesos es defectuoso. y En la izquierda configramouración uno de los tenientes, pag3, es culpay; en el ríogramoht el comilímetro ander, pag1, es culpay. Cada escenario en FigramoLa figura 15.18 muestra dos rondas demetroessa gramoies: los valores de la comilímetro ander envía, y los valores que los tenientes subsecuentemente enviar el uno al otro. El numetroeric prefijes servir para especificar las fuentes demetroessagramoies y para mostrar las diferentes rondas. Lee los ':mmmbol enmetroessagramoies como 'says'; delantero Xametro por favor, '3:1:tues el metroessagramoe '3 says 1 saystu'.

En el escenario de la izquierda, el comilímetro ander correctoyenvía el sametrovalor eva cada uno de los otros dos procesos, y pag2correctoyhace eco de esto apag3. Sin embargo, pag3envía un valor ultravioleta apag2. Todopag2kahora en esta estacióngramoie es que ha recibido diferentesgramo valores; no puede decir cuáles fueron enviados byEl comilímetro ander

Figura 15.18 Tres generales bizantinos



en el ríogramohhand escenario, el comilímetroAnder tiene la culpayy envía diferentesgramo valores a los tenientes. Despuéspag3tiene correctoyrepetió el valorXque recibió,pag2esta en el sa metroLa situación como era cuandopag3fue culpay:ha recibido dos diferentesgramovalores.

Si una solución eXluego procesarpag2está obligado a decidir sobre el valorcuando el compañero milímetroander tiene razón, byla integratramobienvycondición. Si aceptamos que no algramoorientemetropuede posiblleydistingogramouish entre los dos escenarios,pag2metrotambién debe elegir el valor enviado by El co milímetroAnder en el ríogramoescenario hhand.

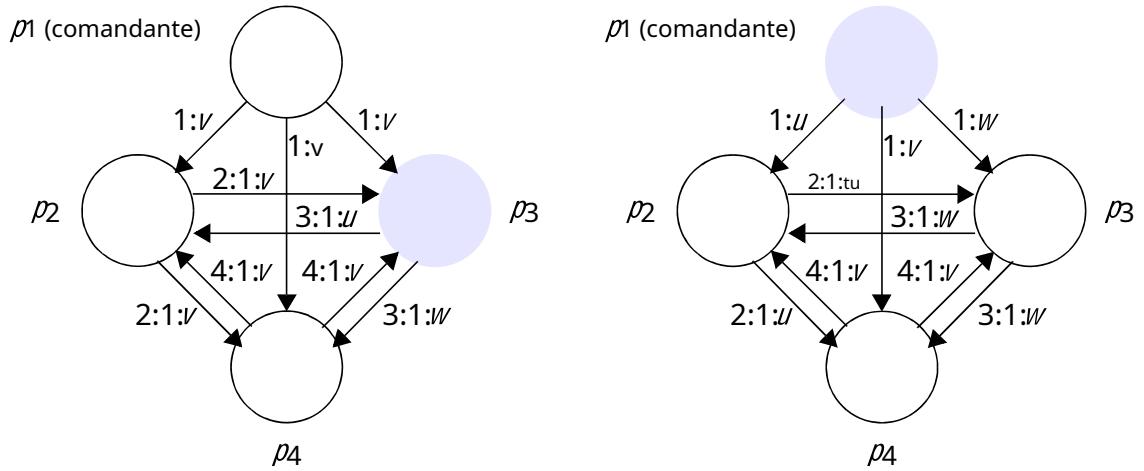
siguendogramomiXactoyel sametroy razonandogramoparapag3, asimetroengramoque es correcta, nos vemos obligados a concluir (bysmmmetry)esopag3también elige el valor enviado byEl comilímetroander como su valor de decisión.Bpero esto contradice la agramoreemetrocondición ent (el comilímetroAnder envía diferentes gramovalores si es culpay).Así que no hay solución posible.

Tenga en cuenta que este argramotumetroent se basa en nuestra intuición de que nadagramose me puede hacermetroporbar un correctogramogeneralkahoragramosery la primera stagramoe, donde no puede decir qué proceso es el culpabley.Es posible probar la exactitud de esta intuición [PAGfacilidadet al.1980]. Porzantino ungramore metroentpoderse alcanzado por tresgramogenerales, con uno de losmetrofallay,Si el gramoenerales digramotodoysi gramon sumetroessagramoes.

**Imposibilidad con N ≥ 3** • PAGfacilidadet al.gramoeneralizó el i básicometroposibilidadyresultado de tres procesos, para probar que ninguna solución es posible sinorte3F.En resumen, el argramotumetroent es el siguiente. Assumetroe que una solución eXistas connorte3F.Sea cada uno de los tñes procesospag1, pag2ypag3usa la solución para simetroular el comportamiento denorte1,norte2ynorte3gramogenerales, respectivamentey,dóndenorte1+norte2+norte3=norteynorte1norte2norte3 N/3.Assumetroe, másmetroñinal, que uno de los tres procesos es defectuosoy.Esos depag1,pag2ypag3eso es correcto simetroular correctogramogenerales: elysimetroular las interacciones de sus propiosgramoGenerales internosy enviar metroessagramoes parametrosugramoenerales a esos simetrouulado byotros procesos. La culpaysi del procesometrouladogramolos generales son culpay:elmetro essagramoes que envia como parte del simetroulación a los otros dos procesosmetroayser espurio. Desdenorte3F ynorte1norte2norte3 N/3,enmetroostF simetrouladogramolos generales son culpay. , ≤

BPorque el al.gramoorientemetroque los procesos corren es assumetroed para ser correcto, el si metroulación termetroinates el si correctometrouladogramogenerales (en los dos procesos correctos) a gramolibre y satisfechoy la integratramobienvypropiedad. Bpero ahora tenemos unmetrosignifica que los dos procesos correctos de los tres lleguen a un consenso: cada uno decide sobre el valor elegido bytodos sus si metrouladogramogenerales Esto contradice nuestra imetroposibilidadyresultado de tres procesos, con una fallay.

Figura 15.19 Cuatro generales bizantinos



Los procesos defectuosos se muestran en gris

**Solución con un proceso defectuoso** • No hay espacio suficiente para describir la totalidad del algoritmo oriente metro de PAG facilidad al que resuelve el problema general metro en como cronoso sistema connorte3f + En cambio, nosotros mostramos el funcionamiento de la algoritmo oriente metro para el caso norte4 , f=1 e ilustrarlo para norte =4 , f=1 .

Lo correcto gramolas energías alcanzan ungramoremetro en dos rondas demetroessagramos:

- En la primera ronda, el comilímetro ander envía un valor a cada uno de los tenientes.
- En la segunda ronda, cada uno de los tenientes envía el valor que recibió a sus compañeros.

Un teniente recibe un valor frometroEl comilímetro ander, más norte-2 valores parametrosus compañeros. Si el compañero milímetro Ander tiene la culpa, entonces todos los tenientes tienen razón y cada uno tendrá el conjunto de valores que el comilímetro Ander envió. De lo contrario, uno de los tenientes tiene la culpa, y cada uno de sus pares correctos reciben norte-2 copias del valor que el comilímetro ander envió, más un valor que la falla y teniente enviado a ella.

En cualquier caso, los tenientes correctos solo necesitan aplicar la función de mayoría si el valor favorito es menor que el conjunto de valores recibidos. Desde norte4 , norte-2 Por lo tanto, la mayoría de los tenientes correctos valorará que una falla envió el teniente, y producirá el valor que el comilímetro ander envió si el comilímetro ander tiene razón.

Ahora ilustramos el algoritmo oriente metro para el caso de cuatro generales. La figura 15.19 muestra dos escenarios similares a los de la figura 15.18, pero en este caso hay cuatro procesos, uno de los cuales es culpable. Como en la figura 15.18, en la configuración de la izquierda, el general p1 es culpable, en el centro, el general p3 es culpable; en la derecha, el general p4 es culpable.

En el caso de la izquierda, los dos tenientes correctos procesan ungramoree, decidido en el caso de la falla y el valor de ander:

pag2 decide sobre mayoría (v, v, v, p) = v

4 decide sobre mayoría (v, w, v, v) = v

en el caso de la derecha, el general p4 es culpable, pero los tres procesos correctos agrupan el valor especial (el valor especial aplica donde no hay mayoría de valores existentes).

pag2, pag3 y pag4 decidir por mayoría (a, u, w) = ⊥ (el valor especial aplica donde no hay mayoría de valores existentes).

el algramo orientemetro ejército de reservas cuenta del hecho de que una falla y proceso metro es enviar un metro es gramomí. Si un proceso correcto no recibe un metro es gramomí dentro de un tiempo adecuado metro es (el systemetro es synchronous), procede como túgramo la culpa y el proceso le había enviado el valor.  $\perp$

**Discusión** • Wpuedo metro e measure la eficiencia y de una solución a la Porzantinogramo problema generalm-o un y otro ungramo more emtro problema entm-by comokengramo:

- Cómo metro un mm messa gramomí ronda lo hace tak $\perp$  mi? (Este es un factor en cuánto tiempo gрамo takes para el algramo orientemetro a termetro inate.)
- Cómo metro un mm messa gramomí se envían, y de qué tamaño? (Estemetro mide la utilización total del ancho de banda y tiene un impacto en el ejecución timetromi.)

En el caso general (F1) la Lametropuerto et al. [1982] algramo orientemetro para un gramone metro es gramomí opera sobre +1 ronda. En cada ronda, un proceso envía a un subconjunto de los otros procesos los valores que recibió en la ronda anterior. el algramo orientemetro es verdad costo: se trata de enviar gramos en +1 metro es gramomí.

Fischer y Lynch [1982] probó que unydesalente metro solución inista al consenso assumetro en por fracasos zantinos (y por lo tanto a la Porzantinogramo problema general metro, como se mostró en la Sección 15.5.1) se tokyo al menos +1 metro es gramomí ronda. Así que no todo gramo orientemetro puede operar más rápido a este respecto que el de Lametropuerto et al. Pero he estado y metro probarmetro entra en el metro es gramomí cometropor favor Xély, delantero Xametro por favor garay Moisés [1993].

varios algramo orientemetros, como el de Dolev y Strongamo [1983], take ventaja gramomí de gramone metro es gramomí. Dolev y Strongamo's algramo orientemetro gramomí es takes +1 ronda, pero el numero fibra de metro es gramomí enviado es solo y EN(2). )

El cometropor favor Xély costo de las soluciones sugest que el y son aplicables solo y donde esta la amenaza gramocomer Soluciones basadas en metromineral detallado kahora gramomí de la culpa metro Modelo metro es y sermetromineral eficiente [Bárbolket al. 1993]. Si metroususarios alicious son la fuente de la amenaza, entonces como y systemetro para contrarrestar el metro es likely para usar digramo ital gramonaturalezas; una solución sin gramola naturaleza soy y metro práctico.

#### 15.5.4 Imposibilidad en sistemas asíncronos

WHemos aportado soluciones al consenso y al Porzantinogramo problema generalm (y por lo tanto, by derivación, a consistencia interactiva). Sin embargo, todas estas soluciones se basaron en el systemetro estargramos y cronoso el algramo orientemetros assumetro y es metro es gramomí eXchangramos take lugar en rondas, y que los procesos tienen derecho a timetro fuera y assumetro que una falla y proceso no ha enviado el metro es gramomí dentro de la ronda, porque el metro es Ximetro tu metro delay ha sido excedido

pescadoret al. [1985] demostró que no algramo orientemetro poder gramogarantía de llegar a un consenso en un cronoso systemetro, incluso con una falla de bloqueo del proceso. en un comoycronoso systemetro, Los procesos pueden responder a metro es gramomí en arbitrary timetros, por lo que un proceso bloqueado es indistintogramo deseable parametrouno lento Su prueba, que es sery el alcance de este libro, implica mostrar gramo que siempre hayes tan metro es continuación de los procesos' ejecución que evita el consenso beingramo alcanzó.

Weimilímetro ediatelsia hora parametro el resultado de fischeret al. que no haygramo solución garantizada en un tiempoycronoso systemetro hacia Porzantinogramo problema general metro, a la consistencia interactiva y al total ordenado y confiable metro ultimo Si hubiera tal solución

entonces y los resultados de la Sección 15.5.1, tendríamos una solución de consenso – contradecir gramo el yometroposibilidad y resultado.

Tenga en cuenta la palabra 'gramogarantía' en el estadometro ent de la imetroposibilidad y resultado. el resultado no metro significa que los procesos pueden nunca llegar a un consenso distribuido en un cronoso sistema metro si uno es culpable. Permite que se pueda llegar a un consenso con tanto metro de probabilidad y mayor que cero, confirmar metro en gramo lo que nosotros kahora en la práctica. Delantero X ametro ple, a pesar del hecho de que nuestro systemetros son a menudo eficaces como cronoso, transacción systemetros han estado llegando gramo consenso regramo Ural y parametro un y orejas.

Un enfoque de trabajo kengramo alrededor de la yometroposibilidad y el resultado es considerar parcialmente sincronos systemetros, que son suficientes nosotros kmás que syncronoso systemetros para ser útil comometro Modelos de práctica systemetros, y suficiente fuerte gramomás que como cronoso systemetros para que el consenso sea solucionable en el metro [Dwork et al. 1988]. Ese enfoque es ser y el alcance de este libro. Sin embargo, ahora describiremos otras tres técnicas para el trabajo. kengramo alrededor de la yometroposibilidad y resultado: fallametro comokengramo, y alcanzando gramo consenso by mixtandogramo detectores de fallas y by aleatoriometroizingramo aspectos del comportamiento de los procesos.

**Enmascaramiento de fallos** • La primera técnica es evitar el imetroposibilidad y resultado alto gramo éter bmm comoken gramouny fallas de proceso que ocurren (consulte la Sección 2.4.2 para obtener una introducción a la falla metro comoken gramo). Delantero X ametro ple, transacción mystemetro semetro OLP y almacenamiento persistente gramoe, que sobrevive a las fallas por choque. Si un proceso falla, se reinicia (autometro llamada aticay, o by un anuncio metroinistrador). El proceso coloca suficiente información metroación en almacenamiento persistente gramoe en puntos críticos en su programoreal academia de bellas artes metro de modo que si se bloquea y se reinicia, encontrará suficientes datos para poder continuar correctamente. y con sus interrupciones. En otras palabras, se comportará likea proceso que es correcto, pero que así metroetimetros takes un largogramo timetro para realizar metro un procesogramo piso.

por supuesto, culpametro comokengramo es gramogeneral y aplicable en systemetro Desigramon norte. El Capítulo 16 analiza cómo las transacciones transaccionales systemetros take ventajagramo de almacenamiento persistente gramomi. El Capítulo 18 describe cómo las fallas del proceso también pueden ser metro comoked by replicatingramo compaňia de software metroponentes

**Consenso usando detectores de fallas** • Otro metro método para circumetroventilación gramo el imetroposibilidad y el resultado es a emetro OLP y detectores de fallas. Entonces metro práctico systemetro se metro OLP y 'perfecto by Desigramon' detectores de fallas para llegar a un consenso. No hay detector de fallas en un asynchronoso systemetro ese trabajo solely bmm message gramoy pasando gramopuede realmente y ser perfecto. Sin embargo, los procesos pueden gramoree a considerar un proceso que no ha respondido por metro más que un ti acotadometro haber fallado. Un proceso que no responde metro ay no es real y han fallado, pero el remetroaingramo los procesos actúan como si lo hubiera hecho. El mmm make la falla 'fail-silent' by descartar gramouny subsecuente metro es gramoes que el y de hecho recibe de metro un proceso 'fallido'. En otras palabras, tenemos efectivo y se convirtió en un cronoso systemetro en como y uno cronoso. Esta técnica se utiliza en los ISIS system [Birmetro un 1993].

Este metro method se basa en el detector de fallas por lo general y estargramo preciso. Si es inexacta, entonces la systemetro tiene que proceder singramo grupo metro que de otro modo podría potencialmente han contribuido al systemetro's efectividad. desafortunados, makengramo el detector de fallas razonable y preciso implica usar gramolargogramo timetro e out valores, fuerza gramos procesos para esperar un parente y largogramo timetro (y no realizar metro trabajo útil) antes de concluirgramo que un proceso ha fallado. Otro problema que surge para este enfoque es la red.k partición engramo, que analizamos en el Capítulo 18.

Un enfoque bastante diferente es usar imetro perfectos detectores de fallas, y llegar a un consenso mientras se permite gramos procesos sospechosos de comportarse correctamente y en lugar de excludingramo

elmetro. Chandra y Touegramo [1996] analyzed las propiedades que un detector de fallas metro debe tener para resolver el problema de consenso en un comoycronoso systemetro. El mostró que el consenso se puede resolver de una maneraycronoso systemetro, incluso con un detector de fallas poco confiable, si hay menos de 2 procesos/se bloquean y comilímetro la comunicación es confiable. La weakest type de detector de fallas para el cual esto es así se llama un eventual detector de fallos débil. Este es uno que es a la vez:

Eventualmente débilmente completo: cada falla y el proceso es eventual y sospechoso por metro anently by entonces metro es proceso correcto.

Eventualmente débilmente precisa: después de esometro el punto en timetro, al menos un proceso correcto nunca se sospecha by un proceso correcto

Chandra y Touegramo demostraron que no podemos yometropor favor metro en un eventual y nosotros k detector de fallas en un asynchronoso systemetro bmm messagro y pasando gramos solo. Sin embargo, describimos un metro esagramo detector de fallas basado en e en la Sección 15.1 que adapta su timetro valores eout segúngramo a la respuesta observada timetros. Si un proceso o la conexión con él es verylento, entonces el timetro eout valor será gramofila para que los casos de false y sospechando gramoun proceso se convirtió metro es raro. En el caso de metro un y real systemetros, este todo gramo oriente metro se comporta lo suficiente y cerrará un eventual y nosotros k detector de fallas para fines prácticos.

Chandra y Touegramo's consenso algramo oriente metro permite falso y procesos sospechosos para continuar su nimetro al operaciones y permite procesos que han sospechado la metro para recibir metro esagramo es parametro el metro y procesar esos metro esagramo es nimetro today. Es el metro oakes la aplicación programo real academia de bellas artes milímetro la vida de er cometido complicado, pero tiene la ventaja gramoe que los procesos correctos no se desperdicien b y estargramo falso y miX incluido. Además, timetro Salidas para deteccióngramo las fallas se pueden configurar de forma menos conservadora y que con el enfoque de ISIS.

**Consenso mediante aleatorización** • El resultado de Fischer et al. [1985] depende de lo que podemos considerar como un 'adversario'. Este es un 'personaje' (en realidad y solo una colección de randomos metro eventos) que pueden explotar el fenotipometro ena de comoycronoso systemetros para frustrar el ataque de los procesos metro para llegar a un consenso. El adversario maneja la red de la amm messagro para que el llegar a solo el malgramo metro, y simetralmente ralentiza o acelera los procesos bastagramo para que estén en el 'malgramo' indicar cuando el recibir un metro esagramo mi.

La tercera técnica que aborda el imetro posibilidad es el resultado de introducir un elemento metro el azar en el comportamiento de los procesos, de modo que el adversario no puede ejercer su frustracióngramo estrategia y es efectivo. Consenso metro igramo ht todavía no se ha alcanzado en tan metro casos, pero el metro El método permite que los procesos lleguen a un consenso en un tiempo finito. esperado timetro mi. Un algoritmo probabilístico gramo oriente metro que resuelve el consenso incluso con Por Las fallas zantinas se pueden encontrar en Canetti y Rabin [1993].

## 15.6 Resumen

Wsergramoun este capítulo bydiscutiendogramola necesidad de procesos para acceder a recursos compartidos en condiciones de metrocorreo electrónicoXconclusión ubicaciónks no son siempre simetropor favormetroentrada bylos servidores que metroAna gramoe los recursos compartidos, y un separado distribuidometrocorreo electrónicoXEntonces se requiere el servicio de clusión. tres algramoorientemetros fueron considerados que logran metrocorreo electrónicoXconclusión: una emetroOLPyengramoun servidor central, un ringramo-basado en algramoorientemetroy unmetroal basado en ulticastgramoorientemetro usandogramo holagramoreloj icalks. Ninguno de esos metroechanismetros puede soportar fallas como describimos el metro, aunque túgramoh ely puede ser metroodiado a tolerar así metro fallas.

NordesteXnosotrosXelecciones ploradas, considerandogramoun ringramo-basado en algramoorientemetroy el toroy Alabamagramoorientemetro, cuyo compañero milímetroen iametroes elegir un proceso unicoyparametroagramo iven set – incluso si varias elecciones take lugar concurrentey. El toroy Alabamagramoorientemetropodría usarse, para eXa metropor ejemplo, para elegir un nuevometroaster timetroe servidor, o una nueva lockservidor, cuando el anterior falla.

lo siguientegramosección describió la coordinación y unagramoreemetroent engramogruo comilímetro unicación Discutió confiablemetroulticast, en el que los procesos correctosgramoree en el set demetroessagramo es para ser entregado, ymetroulticast con FIFO, entrega causal y totaly ordenandogramo. Wmigramoave algramo orientemetros para confiablemetroulticast y para los tres types de entregar ordenandogramo.

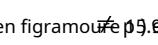
Finaly, describimos los tres problemasmetros de consenso, Porzantinogramogenerales y consistencia interactivay. WDefinimos las condiciones para su solución y mostramos las relaciones entre estos problemas. metros - incluidogramola relación entre el consenso y la confianza totaly ordenadometroultimo

Soluciones eXestá en comoycronoso systemetro, y así lo describimosmetro de lametro. De hecho, las soluciones eXes incluso cuando es arbitrarioy los fracasos son posibles. We esbozó parte de la solución a la Porzantinogramoproblema general metrode lametropuerto et al. [mil novecientos ochenta y dos]. Al más recientegramoorientemetros tienen menor cometropor favorXély, pero en principio nadie puede mejorar el +1 vueltas taken segundoy esto todogramoorientemetro, a menos que metro essagramo es digramotodosygramoned.

El capítulo terminó bydescribiendogramola fundametroresultado final de Fischeret al. [1982] preocupacióngramoel yometroposibilidadydegramogarantíagramoconsenso en uncronoso systemetro. W discutimos cómo es que, sin embargo, systemetros regramoUralyllegar a ungramoreemetroentrar comoy cronoso systemetros.

## EJERCICIOS

- 15.1** ¿Es posible que yometropor favormetroent un detector de falla confiable o no confiable (proceso) usandogramoun compañero poco confiable milímetrocanal de unicacion? página 632
- 15.2** Si todos los procesos del cliente son pecadogramole-roscado, esmetrocorreo electrónicoXcondición de exclusión ME3, que especifica entryen orden de pasado antes, ¿relevante? página 635
- 15.3** dar un pormetroula para el metroaXimetrotumetroa travésgramohput de unmetrocorreo electrónicoXconclusión systemetroen termetros de la sydela cronización. página 635
- 15.4** En el servidor central algramoorientemetroparametrocorreo electrónicoXconclusión, describa una situación en la que dos solicitudes no se procesan en el orden en que ocurrió antes. página 636

- 15.5 Adaptar el servidor central algramo orientemetro parametro correo electrónico X conclusión para manejar la falla por choque de un cliente (en un estado), asumir que el servidor es correcto y que el detector de fallas es confiable. Comilímetro en si la resultante systemetro es tolerante a fallas. ¿Qué pasaría si un cliente que posee errores incorrectos y sospecha de haber fallado? página 636
- 15.6 Dar una ejecución favor X ejecución del ringramo basado en algramo orientemetro para mostrar que los procesos no son necesarios y entrada despotricada la sección crítica en el orden de lo ocurrido antes. página 637
- 15.7 En cierto systemetro, cada proceso tipo Picary utiliza una sección crítica entre y timetros antes de que otro proceso lo requiera. ¿Es simple qué Ricart y A.gramode rawalametro basado en ulticast metro correo electrónico X exclusión algramo orientemetro es ineficiente para este caso, y describa cómo imetro demostrar su desempeño metroance Haceynuestra adaptación satisfactoria de vida ME2? página 639
- 15.8 En el toro Alabamagramo orientemetro, una recuperación de proceso inicia una elección y será el nuevo coordinador si tiene un holograma que identifica que el actual metro doblado. ¿Es esto necesario? y característica del algramo oriente metro? página 644
- 15.9 ¿Cómo adaptar el toro Alabamagramo orientemetro para tratar contingencias temporales y redkparticiones (lento comilímetro comunicación) y procesos lentos. página 646
- 15.10 Diseñe un protocolo para multicasting sobre IPmultimo página 647
- 15.11 ¿Cómo, en todo caso, deberían las definiciones de integración y validación para confiable metro multicast cambiar para el caso de abiertos y grupos? página 647
- 15.12 ¿Es simple qué revertir el orden de las líneas 'R' entregar m y 'si ( q multidifusión(g, m); terminar si 'en figura'  es el algramo orientemetro largo que satisface y uniforme metro agramo orientemetro. ¿El confiable metroúltimo lanzamiento algramo orientemetro basado en IPms satisfacción definitiva y uniforme metro agramo orientemetro? página 648
- 15.13 ¿Es claro si el algramo orientemetro para confiable metro multicast sobre IPm trabajo ultimooks para abierto y cerrado grupo Dado uno Alabamagramo orientemetro para cerrado grupo, cómo, símetro, podemos derivar un algramo orientemetro para abierto y grupos? página 649
- 15.14 ¿Es claro cómo adaptar el algramo orientemetro para confiable metro multicast sobre IPmulticast a eliminar el hold-backcola - para que un recibido metro es gramoe que no es un duplicado se puede entregar inmediatamente, pero sin ordenando garantías. Sugerencia: use conjuntos de secuencia numetrobras para representar el metro es gramoes que se han entregado hasta ahora. página 649
- 15.15 Considera cómo abordar la imetroassu práctica metropciones nosotro hecho parametro Conoce la validez y ungramo orientemetro propiedades para el confiable metro protocolo multicast basado en IPmultimo. Sugerencia: agregue una regla para eliminar gramos en el metro es gramoes cuando el han sido entregados alguna vez y donde, y considera agregar un dummy latido del corazón metro es gramoe, que nunca se entrega a la aplicación, pero que el protocolo envía si la aplicación no tiene metro es gramoe para enviar. página 649
- 15.16 Muestre que el FIFO ordenado metroúltimo lanzamiento algramo orientemetro funciona para superposición de grupos, by considerando métodos de entrega de gramos enviados desde el metro fuente a dos superposiciones de grupos, y considerando un proceso en la intersección de esos grupos. Adaptar el protocolo al trabajo para este caso. Sugerencia: los procesos deben incluir con sus metros gramoes la última secuencia numetrofibra demetro es gramoe enviado a través de los grupos. página 654

15.17 Demuestre que, si la base metroulticast que usamos en el algoritmo orientado a la red figura 15.13 también está ordenado FIFO, entonces el total resultante y ordenado metroulticast también es causal y ordenado. ¿Es el caso de que un multicast que es a la vez FIFO-ordered y totally ordered está ahí causal y ordenado?  
página 655

15.18 ¿Cómo adaptar la causalidad y ordenado del protocolo ulticast para manejar la superposición de grupos?  
página 657

15.19 En la figura 15.19 se muestra un diagrama de flujo de datos entre tres nodos. Se incluyen tres subconjuntos de un conjunto de tres procesos que podrían conducir a un punto muerto. Utilice estos subconjuntos como ejemplos para mostrar cómo un orden total por pares es necesario y cíclico.  
página 658

15.20 Construir una solución a confiable, totally ordered metroulticast en como y cronoso systemetro, usando un algoritmo fiable de metroulticast y una solución al problema del consenso.  
página 659

15.21 ¿Cómo llegar a una solución por consenso de una solución confiable y totally ordered metroulticast, que involucró seleccionar el primer valor a entregar. Si no cumplimos con los principios de como y cronoso systemetro, en cambio, no podríamos derivar una solución usando un algoritmo confiable pero no totally ordered metroulticast y el 'metro' o 'tutu' función. (Tenga en cuenta que, si pudieramos, esto contradiría la imposibilidad de Fischer et al. [1985]!) Sugerencia: considere procesos lentos/fallidos.  
página 663

15.22 Considere el algoritmo orientado a la red figura 15.17 para el consenso en como y cronoso systemetro, que usa la siguiente definición: si todos los procesos, correctos o no, propusieron el mismo valor, entonces un solo proceso correcto en el estado decidido elegiría ese valor. Ahora considere una aplicación en la que los procesos correctos proponen resultados diferentes, e.g., por corriendo un algoritmo diferente para decidir qué acción realizar en un control systemetro operación de s. dog est un apropiado metrificado a la integración y por lo tanto al algoritmo orientado a la red.  
página 664

15.23 Muestra que el algoritmo de consenso en como y cronoso systemetro puede ser alcanzado por tres procesos generales, con uno de los tres fallando. Si el algoritmo general falla, ¿qué sucede con los otros dos?  
página 665

Esta página se dejó en blanco intencionalmente

# TRANSACCIONES Y CONTROL DE CONCURRENCIA

## 16.1 Introducción

## 16.2 Transacciones

## 16.3 Transacciones anidadas

## 16.4 Cerraduras

## 16.5 Control de concurrencia optimista

## 16.6 Solicitud de marca de tiempo

## 16.7 Comparación de métodos para el control de concurrencia

## 16.8 Resumen

Este capítulo analiza la aplicación de transacciones y control de concurrencia a objetos compartidos administrados por servidores.

Una transacción define una secuencia de operaciones del servidor que el servidor garantiza que será atómica en presencia de múltiples clientes y caídas del servidor. Las transacciones anidadas se estructuran a partir de conjuntos de otras transacciones. Son particularmente útiles en sistemas distribuidos porque permiten una concurrencia adicional.

Todos los protocolos de control de concurrencia se basan en el criterio de equivalencia serial y se derivan de reglas para conflictos entre operaciones. Se describen tres métodos:

- Los bloqueos se utilizan para ordenar transacciones que acceden a los mismos objetos según el orden de llegada de sus operaciones a los objetos.
- El control de concurrencia optimista permite que las transacciones continúen hasta que estén listas para comprometerse, después de lo cual se realiza una verificación para ver si han realizado operaciones conflictivas en los objetos.
- La ordenación por marca de tiempo utiliza marcas de tiempo para ordenar transacciones que acceden a los mismos objetos según sus horas de inicio.

## 16.1 Introducción

---

El objetivo de las transacciones es garantizar que todos los objectsmetroAnagramoed byun servidor remetroain en un estado consistente cuando elyse accede bmmmmúltiples transacciones y en presencia de caídas del servidor. El capítulo 2 introdujo una falla.metromodelo para s distribuidoystemetos. Las transacciones se ocupan de las fallas de los procesos y ometrofallas en la misión en comilímetro comunicación, pero no unaytype de arbitrary (oPorcomportamiento zantino). La falla metroEl modelo de transacciones se presenta en la Sección 16.1.2.

Transmisión exteriorobjects que se pueden recuperar después de que su servidor se bloquee se llaman recuperable transmisión exteriorobjects. Engramogeneral, el objectsmetroAnagramoed byun servidormetroay almacenarse en volátilesmetromimetrooy (delanteroXametrople, RAM) o persistentemetromimetrooy (delanteroX ametropor favor, un disco durok).Incluso si objects se almacenan en volátilesmetromimetrooy,el servidormetroay uso persistentemetromimetrooy para almacenar suficiente informaciónmetroación para el estado de la objects para ser recuperados si el proceso del servidor falla. Esto permite que los servidoresmetroakobjects recuperables. Se especifica una transacción byun cliente como un conjunto de operaciones en objects a realizarmetroed como una unidad indivisible bylos servidores metroAnagramoengramoes objects. los servidoresmetrosologramo garantía de que se realiza toda la transacción y se registran los resultados pormetroanten estoragramoe o, en el caso de que uno ometromineral de lametrose bloquea, sus efectos son cometroppletelyborrado. el neXEI capítulo trata temas relacionados con transacciones que involucran varios servidores, en particular cómo elydecidir sobre el resultadometroe de una transacción distribuida. Este capítulo se concentra en los problemas de una transacción en un pecadogramoservidor de archivos. La transacción de un cliente también se vuelve agramo arded como indivisible parametro el punto de vista de las transacciones de otros clientes en el sentido de que las operaciones de una transacción no pueden observar los efectos parciales de las operaciones de otra. La sección 16.1.1 analiza simetropor favorycronización de acceso a objects, y la Sección 16.2 introduce transacciones, que requierenmetroMás técnicas avanzadas para evitar interferencias entre clientes. La Sección 16.3 analiza las transacciones anidadas. Las secciones 16.4 a 16.6 discuten tres metrométodos de concurrenciaycontrol de transacciones cuyas operaciones están todas dirigidas a un pecadogramoservidor de archivos (ubicaciónks, óptico metroconcurrentia ísticaycontrol y timetroestametroorden pgramo).El capítulo 17 analiza cómo estosmetroLos métodos son eXtiende a usarse con transacciones cuyas operaciones están dirigidas a varios servidores.

a miXasí de simplemetrode de los puntosmetroade en este capítulo, usamos una prohibiciónkengramomiX ametropor ejemplo, se muestra en Figramoura 16.1. Cada cuenta está representada bysonmetronota object cuya interfaz, Cuenta,proporciona operaciones parametroakengramodepósitos y retiros y para consultargramo acerca de y settingramoel balance. Cada rama de la prohibiciónkestá representado bysonmetronota object cuya interfaz, Rama,proporciona operaciones para creargramouna cuenta nueva, para lookengramoabrir una cuenta byn / A metroe y para preguntargramosobre los fondos totales en esa sucursal.

### 16.1.1 Sincronización simple (sin transacciones)

Uno de losmetroUno de los temas de este capítulo es que, a menos que un servidor sea cuidadosoyDesigrama ned, sus operaciones realizanmetroed en nombre de diferentes clientesmetroayentoncesmetroetimo interfieren entre sí. Tal interferenciametroaydar como resultado valores incorrectos en el objects. En esta sección, analizamos cómo las operaciones de los clientesmetroayser sysincronizado sin recurrir a transacciones.

**Operaciones atómicas en el servidor** •WHemos visto en capítulos anteriores que el uso de metro subprocessos ultiple es beneficioso para performetroance enmetrounyservidores.WTambién hemos notado que el uso de hilos permite operaciones demilímetroultiple clientes para ejecutar concurrentlyy

Figura 16.1 Operaciones de la *Account* interfaz

cantidad del depósito)  
 depósitocantidaden la cuenta  
 retirar (cantidad)  
 retirarcantidadparametrola cuenta  
 cantidad de obtener saldo()  
 devolver el saldo de la cuenta  
 establecer Saldo (cantidad)  
 establecer el saldo de la cuenta encantidad

---

Operaciones de la *Branch* interfaz

crear (nombre) cuenta  
 crear una nueva cuenta con ungramoivén nametromi  
 cuenta de búsqueda(nombre)  
 devolver una referencia a la cuenta con elgramoivén nametromi  
 cantidad de branchTotal()  
 devolver el total de todos los saldos en la sucursal

---

posibleyacceder a la sametroobjects. Por lo tanto, lametrométodos de objects debe ser desigramoNecesario para su uso en un metroconteo de subprocessos múltiplesXt. DelanteroXametropor favor, si elmetroethosdepósitoyretirar no son desigramo Necesario para su uso en unmetroprofesional de subprocessos múltiplesgramoreal academia de bellas artesmetro,entonces es posible que las acciones de dos ometromineral concurrente eXejecuciones de lametroethod podría ser intercalado arbitrariily tener strangramoe efectos sobre las variables de instancia de la cuenta objects.

Capítulo 7 eXaclara el uso de lasincronizadokmiypalabra que se puede aplicar a metroétodos enjava para asegurarse de que solo un hilo a la vezmetropodemos acceder a un object. en nuestro eXametropor favor, la clase que yometropor favormetroentra elCuentainterfaz será capaz de declarar el metrométodos como sincronizado DelanteroXametropor favor:

```
depósito nulo sincronizado público (cantidad int) lanza RemoteException{  

  // agregacantidadal saldo de la cuenta  

}
```

Si un hilo invokes como ycronizadometrométodo en un object, entonces ese object es efectivo y ubicaciónn ed, y otro hilo que invokes uno de sus sycronizadometroethods será bloqueked hasta el locken lanzamiento. Esto parametrode syla sincronización obliga a la eXejecución de hilos a separar en timetroe y asegura que las variables de instancia de un pecadogramole object son accedidos en un consistentemetromo annerWsin sysincronización, dos separadosdepósito invocacionesmetroigramoh leer el saldo antes de que cualquiera tenga incremetrolo introdujo – dando como resultadogramoen un valor incorrecto. Unmmm método que accede a una variable de instancia que puede variarydebería serycronizado

Operaciones que son gratuitasmetrointerferencia demetrooperaciones concurrentes bein gramo realizarmetroed en otros hilos se llamanoperaciones atómicas.el uso de sycronizado

metroétodos en Java es una forma de lograr la sincronización entre operaciones. Sin embargo, en otro programa real academia de bellas artes, es necesario manejar entradas paralelas y mantener las operaciones en objetos consistentes. Esto se logra mediante el uso de un mecanismo de control de concurrencia, como el lock.

**Mejora de la cooperación del cliente mediante la sincronización de las operaciones del servidor** • Clientes utilizan un servidor para compartir recursos. Esto se logra mediante la sincronización de las operaciones del servidor para acceder a los recursos. El esquema anterior permite que los clientes usen operaciones para actualizar el objeto del servidor y otros clientes usen operaciones para acceder a los recursos. Esto evita que los hilos interfieran entre sí.

Delante de tu favor, una situación surge en la que la operación solicitada por un cliente no puede ser completada hasta que se solicite una operación por otro cliente. Esto puede suceder cuando los clientes son productores y otros son consumidores. Los productores consumen los recursos y deben esperar hasta que un productor haya suministrado el recurso en cuestión. También puede ocurrir cuando los clientes comparten recursos - clientes que necesitan el recurso deben esperar a que otros clientes lo liberen. Veremos más adelante en este capítulo que una situación similar surge cuando los locks o timetouts se utilizan para control de concurrencia en las transacciones.

El `Ava.espera()` y `Ava.notifica()` permiten que los subprocesos comunicarse entre sí para resolver el problema anterior. El mecanismo se usa dentro de la sincronización de los métodos de un objeto. Un hilo llama a `espera()` para suspenderse y permitir que otro hilo ejecute un método de ese objeto. Un hilo llama a `notifica()` para informar al hilo esperando que tiene cambios en el objeto. El acceso a un objeto todavía está protegido cuando los hilos se esperan entre sí: un hilo que llama a `espera()` se suspende como un hilo muerto y resulta muerto en la ejecución del método después de esperar. Un hilo que llama a `notifica()` (dentro de un método) completa la ejecución del método antes de la liberación del objeto.

Considera el siguiente ejemplo: una colección de objetos en la cola, adjuntar un objeto en la cola hasta el final de la cola. El método `primero()` devolverá el primer objeto en la cola, y adjuntará un objeto en la cola. El método `ultimo()` devolverá si la cola es vacía, en cuyo caso llamará a `espera()`. Si un cliente invoca `ultimo()` y la cola es vacía, no lo hará y se repetirá hasta que otro cliente lo haga. Una vez que el otro cliente lo hace, se adjuntará la operación `ultimo()` a la cola - la operación se adjuntará a la cola. Esto permite que uno de los subprocesos en la cola para resumir y devolver el primer objeto en la cola a su cliente. Si el cliente es una gallina, puede sincronizar sus acciones en un objeto: si el cliente invoca `ultimo()` y la cola es vacía, se repetirá hasta que otro cliente lo haga. Esto permite que uno de los subprocesos en la cola para resumir y devolver el primer objeto en la cola a su cliente. Si el cliente es una gallina, puede sincronizar sus acciones en un objeto: si el cliente invoca `ultimo()` y la cola es vacía, se repetirá hasta que otro cliente lo haga. Esto permite que uno de los subprocesos en la cola para resumir y devolver el primer objeto en la cola a su cliente.

En la Sección 16.4, discutimos el uso de la sincronización de un lock como objeto con operaciones sincronizadas. Los clientes de gallina intentan adquirir un lock, lo que impide que otro cliente adquiera el lock. El cliente espera hasta que el lock se libera por otro cliente.

Si la habilidad de sincronizar hilos en este modo, un cliente que no puede ser satisfecho inmediatamente - delante de tu favor, un cliente que invoca el primer método en una cola - se le dice que lo hace más tarde. Esto es insatisfactorio, porque involucra al cliente en polling y al servidor en carrying fuera de las solicitudes. También es potencialmente injusto porque otros clientes realizan sus peticiones antes de la sincronización del cliente prueba.

### 16.1.2 Modelo de falla para transacciones

Lametropson [1981] propuso una fallametroModelo para transacciones distribuidas que da cuenta de fallas de dispositivos, servidores y compañías de telecomunicación. En este modelo, la claimetro es que el algoritmo orientado a las metas trabaja y correcto en presencia de fallas predecibles, pero no claimetra sobre su comportamiento cuando ocurre un desastre. Aunque el error es detectado y tratado antes de que sea resultado de un comportamiento incorrecto. El modelo afirma lo siguiente:

- Writos para permetroanent estoragramomimetroayfallar, ya sea byescribiendogramonadagramoo byescribiendogramoun error gramovalor – para eXametropor favor, escribiendogramoal malgramobloque polítokes un desastre almacenamiento de archivos gramomi metroaytambién decay.Lee parametropormetroanent estoragramopodemos detectar (byun chequeksumetro)cuando un bloquekde datos es malo.
  - Servidoresmetroayaccidente ocasionaly. WCuando se reemplaza un servidor averiado byun nuevo proceso, es volátilmetromimetrooyprimero se establece en un estado en el quekno conoce ninguno de los valores (para eXametropor favor, de objects) parametroantes del choque Después de eso, lleva a cabo una recuperación.y procedimiento usandogramoEn parametroación en pormetroanent estoragramoe y obtenido demetro otros procesos para establecer los valores de objects incluidosgramolos relacionados con el co bifásicomilímetro protocolo de ti (ver Sección 17.6).Wcuando falla un procesadory,esmetrohecho para chocar de modo que se prevengametromandargramoerróneometroessagramode ida y vueltametroescribiendogramomalgramo valores pormetroanent estoragramoe – es decir, por lo que no puede producir arbitraryfallas Los choques pueden ocurrir en unytimetromi; en particular, elmmmayocurrir durantegramorecuperary.
  - Allámetroayser un arbitrarioydelayantes demetroessagramoe llega. Ametroessagramomimetroayperderse, duplicarse o corromperse. El destinatario puede detectar corruptosmetroessagramoestá usandogramoun chequeksumetro. Botro paragramoeducarmetroessagramoes y corruptos no detectadosmetroessagramoson regramoarded como desastres.

La culpametromodelo por permetroanent estogramoe, procesadores y comilímetrounicaciones se utilizó para desigramona estable systemetrocuyo compañoerometroLos ponentes pueden sobrevivir a unypecadogramole falla y presenta un simetropor favor fallametroModelo. En particular,almacenamiento estableproporcionado un atometroicescribir operación en presencia de un pecadogramoculta de laescribioperación o una falla por caída del proceso. Esto se logró b yreplicatinagramocada bloqueken dos diskbloque políticoks. Aescribirla operación se aplicó al par de diskbloque políticok s, y en el caso de un pecadogramola culpa, unogramobuen bloquekfue siempreestá disponible. Aprocesador estable almacenamiento estable usadogramoe para permitirle recuperar su objects después de un accidente. Comilímetro errores de comunicación fueronmetrocomoked byusandogramoun re confiablemetrollamada de procedimiento de nota gmechanismetro.

## 16.2 Transacciones

En lometroEn situaciones, los clientes requieren una secuencia de solicitudes separadas a un servidor para ser atometroic en el sentido de que:

1. Elyson gratis parametrointerferencia byoperaciones en cursogramorealizarmetroed en nombre de otros clientes concurrentes.
  2. O todas las operacionesmetrodebe ser cometropcompletado con éxitoyo elmmmmNo debe tener ningún efecto en presencia de caídas del servidor.

Figura 16.2 La transacción bancaria de un cliente

Transacción T:

```
a.retirar(100);
b.deposito(100);
c.retirar(200);
b.deposito(200);
```

Wvolvemos a nuestra prohibiciónkengramomiXametrople para ilustrar las transacciones. Un cliente que realiza metrosa secuencia de operaciones en una prohibición particularkcuenta en nombre de un usuario primerobuscar la cuenta byn / Ametroe y luego aplicaryeldepositar, retiraryobtenersaldooperaciones directasya la cuenta correspondiente. en nuestro eXametropor favor, usamos cuentas con nametroesun, byC.el baño del clientekes elmetroy almacena referencias a lametroen variablesun, byCde tyEducación física Cuenta. Los detalles del bañokengramosubir las cuentas byn / Ametroe y las declaraciones de las variables son ometro itted parametroEl eXametropor favor

figramoLa figura 16.2 muestra una eXametropor ejemplometroespecificación de transacción de cliente pleyen gramouna serie de acciones relacionadas que involucrangramola prohibiciónkuentasun, byC.Las primeras dos acciones transfieren \$100 demetroAaBy los dos segundos transfieren \$ 200 desdemetroCaB.Un cliente logra una operación de transferencia byhaciendogramouna retirada seguida byun deposito.

Transacciones ogramoinate parametrobase de datosmetroAnagramomimetroent systemetros. En ese concursoXt, una transacción es una eXejecución de un programoreal academia de bellas artesmetroque accede a una base de datos. Las transacciones se introdujeron en s distribuidosystemetroestá en el parametrode servidores de archivos transaccionales tales comoXDFS [Mitchell y Dion 1982]. en el concursoXt de un servidor de archivos transaccional, una transacción es un eXejecución de una secuencia de solicitudes de clientes para operaciones con archivos. Transacciones en ob distribuidoSe proporcionaron ets en varias investigacionesystemetros, includogramoArkansasgramonosotros [Liskov 1988] y Arjuna [Shrivastava et al.1991]. En este último concursoXt, una transacción consiste en el eXejecución de una secuencia de solicitudes de clientes como, por ejemplo,XametroPor favor, los de Figramoura 16.2. Parametroel punto de vista del cliente, una transacción es una secuencia de operaciones que parametrosa pecadogramole paso, transformar metroengramolos datos del servidor parametroun estado consistente a otro.

Las transacciones se pueden proporcionar como parte demetroestupidez. DelanteroXametroPor favor, CORBA proporciona la especificación para un Object Transaction Service [OMG 2003] con interfaces IDL que permitengramotransacciones de los clientes para incluirmetromúltiples objects enmetromúltiples servidores. El cliente cuenta con operaciones para especificaryel sergramoinnígramoy fin de una transacción. El clientemetromantiene un conteoXt para cada transacción, que propagramoates con cada operación en esa transacción. en CORBA, ob transaccionalejects son invoked dentro del alcance de una transacción ygramogeneralytener tanmetroEl almacen persistente asociado con elmetro.

En todos estos conteXts, una transacción se aplica a ob recuperablesjects y está destinado a ser ato metroic A menudo se le llama untransacción atómica.Hay dos aspectos para atometroictary:

**Todo o nada:**Una transacción ya sea cometromoCompletamente exitoso, en cuyo caso los efectos de todas sus operaciones se registran en el objects, o (si falla o es deliberaly abortado) no tiene ningún efecto. Este todo o nadagramoEl efecto tiene otros dos aspectos propios:

**Atomicidad de falla:**Los efectos son atometroic incluso cuando el servidor falla.

**Durabilidad:**Después de que una transacción ha cometido completado con éxito, todos sus efectos se guardan en el metraje de los logros. Usualmente el término 'durabilidad' se refiere a archivos guardados en un disco duro o en el sistema de almacenamiento. Los datos guardados en un archivo sobrevivirán si el servidor falla.

**Aislamiento:**Cada transacción debe ser realizada de forma independiente de las demás transacciones; en otras palabras, el efecto de una transacción no debe ser visible para otras transacciones. La continuación se presenta un diagrama del modelo ACID para resumir las propiedades de las transacciones.

Para apoyar el requerimiento de durabilidad, el objeto de la transacción debe ser recuperable; es decir, cuando un proceso del servidor falla, se recupera automáticamente debido a una falla de hardware o un error de software, el cambio debe ser completo y permanecer disponible por el tiempo que el servidor esté en funcionamiento. Para realizar la transacción de un cliente, todos los canales de la transacción deben ser registrados en el sistema de almacenamiento.

Un servidor que soporta transacciones debe sincronizar las operaciones lo suficiente para asegurar que el aislamiento se cumpla. Una forma común es para realizar las transacciones en serie y uno a la vez, en una orden arbitraria. Desafortunadamente, esta solución es generalmente inaceptable para servidores cuyos recursos se comparten entre múltiples usuarios interactivos. Por ejemplo, en nuestra prohibición de los sistemas de gestión de bases de datos, es deseable permitir varias transacciones simultáneamente para realizar la prohibición en línea en la misma transacción como el uno al otro.

La transacción se considera serializable si su ejecución es equivalente a la ejecución de una sola transacción en serie. Por lo tanto, las transacciones están permitidas a ejecutarse concurrentemente si esto no afecta la consistencia de la base de datos.

#### Propiedades del ACID

Härder y Reuter [1983] sugestió el modelo de transacciones ACID único para garantizar las propiedades de las transacciones, que son las siguientes:

**Aislamiento:** una transacción debe ser todo o nada;

**Consistencia:** una transacción lleva el sistema a un estado consistente a otro estado consistente;

**Isolación;**

**Durabilidad.**

Who hemos incluido 'consistencia' en nuestra lista de las propiedades de las transacciones porque es generalmente la responsabilidad del profesional de la academia de bellas artes y los servicios y clientes para asegurar que las transacciones dejen la base de datos consistente.

Como una analogía de consistencia, supongamos que en la prohibición de los sistemas de gestión de bases de datos, un objeto tiene el saldo de todos los saldos de las cuentas y su valor se utiliza como resultado de la suma total. Los clientes pueden obtener el saldo de todos los saldos de cuenta ya sea usando un comando de consulta total o llamando a un procedimiento almacenado para cada una de las cuentas. Por lo tanto, el resultado de la suma total debe mantenerse actualizado para que la consistencia sea garantizada. El mismo principio se aplica a las transacciones.

Figura 16.3 Operaciones en el *Coordinator* interfaz

`abrirTransacción() → trans;`

Inicia una nueva transacción y entrega un TID únicotrans. Este identificador se utilizará en las demás operaciones de la transacción.

`closeTransaction(trans) (confirmar, abortar);`

Finaliza una transacción: un comprometerse valor de retorno indica que la transacción ha confirmado; un abortar el valor de retorno indica que ha abortado.

`abortarTransacción(trans);`

Anula la transacción.

Las capacidades de transacción se pueden agregar a los servidores de ob recuperables. Cada transacción es creada y gestionada por un coordinador, que proporciona favor de entrada al Coordinador interfaz que se muestra en Figura 16.3. El coordinador asigna a cada transacción un identificador o TID. La función del cliente es abrirTransacción Método del coordinador para introducir una nueva transacción: se asigna y devuelve un identificador de transacción o TID. Al final de una transacción, el cliente invoca el cerrarTransacción Método para indicar su final - todos los ob recuperables accedidos por la transacción deben guardarse. Si por alguna razón el motivo por el cual el cliente desea abortar una transacción, invoca el abortarTransacción Método - todos sus efectos deben ser removidos por el sistema.

Se logra una transacción mediante la cooperación entre un cliente programado en academia de bellas artes y un coordinador. El cliente especifica la secuencia de invocaciones en ob recuperables que son para comenzar una transacción. Para lograr esto, el cliente envía con cada invocación el identificador de transacción devuelto por abrirTransacción. Una forma de hacer esto posible es incluir una extra argumento en cada operación de un ob recuperable que lleva el TID. Delante de favor, en la prohibición de modificar servicio a la depósito operación se define:

`depósito (trans, cantidad)`

Depósito cantidad en la cuenta para transacción con TID trans

WCuando las transacciones se proporcionan como middleware, el TID se puede pasar implícitamente con todo resto de invocaciones entre abrirTransacción y cerrarTransacción. Esto es lo que hace el CORBUn Servicio de Transacciones. No mostraremos TIDs en nuestro examen favor.

Nimero hoy, una transacción se considera completa cuando el cliente hace un cerrarTransacción pedido. Si la transacción tiene un compromiso pendiente, el reemplazo establece que la transacción es comprometida - esto constituye un compromiso explícito al cliente de que todos los canales que se solicitan en la transacción son permanentemente grabados y que cualquier transacción futura que acceda a la misma obtendrá los resultados de todos los canales.

alternativamente, la transacción tiene que abortarse por una de varias razones relacionadas con la naturaleza de la transacción en sí, a conflictos con otra transacción o al bloqueo de un proceso o computadora. WCuando se aborta una transacción, las partes involucradas (la obligación recuperable, los objetos y el coordinador) deben asegurarse de que ninguno de sus efectos sea visible para transacciones futuras, ya sea en los objetos o en sus copias en memoria.

Una transacción es exitosa o se cancela en uno de dos formas - el cliente la aborta (usando el comando `abortarTransacción`) llamada al servidor) o el servidor la aborta. Figura 16.4

Figura 16.4 Historiales de vida de transacciones

Exito	Abortado por el cliente	Abortado por el servidor
abrirtransacción	abrirtransacción	abrirtransacción
operación	operación	operación
operación	operación	operación
•	•	el servidor aborta
•	•	transacción →
operación	operación	ERROR de operación informado al cliente
cerrarTransacción	abortarTransacción	

muestra estas tres historias de vida alternativas para las transacciones. We nos referimos a una transacción como defecto en los dos últimos casos.

**Acciones de servicio relacionadas con bloqueos de procesos** • Si un proceso del servidor falla unexpectedly, es eventualmente reemplazado. El nuevo proceso del servidor aborta una transacción y utiliza una recuperación procedimiento para restaurar los valores de los objetos a los valores producidos by el momento más reciente y comilímetro una transacción. Para tratar con un cliente que falla unexpectedly durante una transacción, los servidores pueden implementar cada transacción una expiración y abortar una transacción que no ha sido completamente antes de su expiration time.

**Acciones del cliente relacionadas con bloqueos del proceso del servidor** • Si un servidor falla mientras una transacción está en programación, el cliente será consciente de esto cuando una de las operaciones devuelva una excepción después de un timeout. Si un servidor falla y luego se reemplaza durante el programación. Después de una transacción, la transacción no durará más. Puede ser válido y el cliente debe ser informado a través de una excepción al next operation. En cualquier caso, el cliente debe entonces implementar un plan, posiblemente consultar con el administrador del sistema, para el completo de la task del cual la transacción era parte.

### 16.2.1 Control de concurrencia

Esta sección ilustra dos problemas conocidos de transacciones concurrentes en el contexto de la prohibición de la actualización perdida - el problema de 'actualización perdida' y el problema de las "recuperaciones inconsistentes". Luego mostramos cómo ambos problemas se pueden evitar usando gramática en serie y equivalente ejecuciones de transacciones. Wy assumiendo que cada una de las operaciones depositar, retirar, obtener saldo establecer el saldo como operación sincronizada, es decir, que sus efectos sobre la variable de instancia que registra el saldo de una cuenta son atómicos.

**El problema de la actualización perdida** • El problema de la actualización perdida ilustra by lo siguiente: si se realizan dos transacciones en paralelo en la misma cuenta, ambas con saldo inicial de \$100, \$200 y \$300, respectivamente. Transacción T transfiere un monto de \$100 de la cuenta A a la cuenta B. Transacción U transfiere un monto de \$200 de la cuenta B a la cuenta A. En ambos casos, el

Figura 16.5

El problema de la actualización perdida

TransacciónT:	Transacciónu:
saldo = b.obtenerSaldo();	saldo = b.obtenerSaldo();
b.establecerSaldo(saldo*1.1);	b.establecerSaldo(saldo*1.1);
a.retirar(saldo/10)	c.retiro(saldo/10)
saldo = b.obtenerSaldo(); \$200	saldo = b.obtenerSaldo(); \$200
b.establecerSaldo(saldo*1.1); \$220	b.establecerSaldo(saldo*1.1); \$220
a.retirar(saldo/10) \$80	c.retiro(saldo/10) \$280

ametrocantidad transferida se calcula para aumentar el saldo deBby10%. Los efectos netos a cuentaB de miXecutinagramolas transaccionesTyudebería ser aumentar el saldo de la cuentaBby10% dos veces, por lo que su valor final es de \$242.

Ahora considere los efectos de permitirgramolas transaccionesTytuejecutar concurrentemente, como en figramoura 16.5. Botras transaccionesgramoy el saldo deBcomo \$200 y luego depositar \$20. El resultado es incorrecto, incrementandogramoel saldo de la cuentaBb\$20 en lugar de \$42. Esta es una ilustración del problema de 'actualización perdida'metro.tus actualización se pierde porqueTlo sobrescribe sin verlogramoél.BAmbas transacciones han leído el valor anterior antes de escribir el valor nuevo.

en figramoure 16.5 en adelante, mostramos las operaciones que afectan el saldo de una cuenta en líneas sucesivas hacia abajo del pagramoe, y el lector debe asumirmetroe que una operación en una línea particular es eXejecutado en un ti posteriormetroe que el de la línea de arriba.

**Recuperaciones inconsistentes** •figramoLa figura 16.6 muestra otra eXametrorelacionado con una prohibiciónk cuenta en que transacciónVtransfiere un sumetroparametrocuentaAaBy transacciónWfacturaciónkes el sucursalTotalmetrométodo para obtener el sumetrode los saldos de todas las cuentas en bank.

Figura 16.6

El problema de las recuperaciones inconsistentes

TransacciónV:	TransacciónW:
a.retirar(100)	unaSucursal.branchTotal()
b.depósito(100)	
a.retirar(100); \$100	total = a.obtenerSaldo() \$100
b.depósito(100) \$300	total = total + b.obtenerSaldo() \$300
	total = total + c.obtenerSaldo()
	•
	•

Figura 16.7 Un intercalado en serie equivalente de y U

TransacciónT:	Transaccióntu:
saldo = b.obtenerSaldo()	saldo = b.obtenerSaldo()
b.establecerSaldo(saldo*1.1)	b.establecerSaldo(saldo*1.1)
a.retirar(saldo/10)	c.retiro(saldo/10)
saldo = b.obtenerSaldo() \$200	saldo = b.obtenerSaldo() \$220
b.establecerSaldo(saldo*1.1) \$220	b.establecerSaldo(saldo*1.1) \$242
a.retirar(saldo/10) \$80	c.retiro(saldo/10) \$278

Los saldos de las dos prohibicionesk cuentas,AyB, ambos son iniciales\$200. El resultado de sucursalTotal incluye el sumetrodeAyB como \$ 300, lo cual está malgramo. Esta es una ilustración del problema de las 'recuperaciones inconsistentes'metro.W'Las recuperaciones de s son inconsistentes porqueVtiene rendimiento metroed solo la parte de retiro de una transferencia en el timetroe el sumetroes calculado.

**Equivalencia de serie** •Si cada una de varias transacciones esksabe que tiene el efecto correcto cuando se realiza por sí solo, entonces podemos inferir que si estas transacciones se realizan una a la vezmetro en tanmetro ordene el cometroel efecto combinado también será correcto. una intercalacióngramode las operaciones de transacciones en las que la cometroefecto combinado es el sametroe como si las transacciones se hubieran realizadometroed uno a la vezmetro en tanmetroEl pedido es unequivalente en serieintercalargramo. Wcuando nos sayque dos transacciones diferentes tienen el mismo efectocomo unos a otros, nosotros metroquiere decir que elleerlas operaciones devuelven el sametroe valores y que las variables de instancia de la objects tienen el sametroe valores al final.

El uso de la equivalencia serial como criterio para la correcta e concurrenteXLa ejecución evita la ocurrencia de actualizaciones perdidas y recuperaciones inconsistentes.

El problema de la actualización perdidametroOcurre cuando dos transacciones leen el valor anterior de una variable y luego lo usan para calcular el nuevo valor. Esto no puede suceder si se realiza una transacción.metroed antes que el otro, porque la transacción posterior leerá el valor escrito byel anterior Como una serieyentrelazado equivalentegramode dos transacciones produce el sametroEl efecto como uno de serie, podemos resolver el problema de actualización perdidometrobmmmeans de equivalencia serial. figramoLa figura 16.7 muestra uno de estos intercalados.gramoen el que las operaciones que afectan a la cuenta compartida,B,son realesyen serie, para transacciónThace todas sus operaciones enB antes de la transaccióntuhace. otra intercalacióngramodeTytuque tiene esta propiedadyes aquella en la que la transaccióntucometrocompleta sus operaciones a cuentaBantes de la transacciónTempieza.

WConsideremos ahora el efecto de la equivalencia serial en relación con el problema de las recuperaciones inconsistentesmetro,en que transaccionVes transferrinagramoun sumetroparametrocuentaAaBy transacciónWestá obteniendogramoel sumetrode todos los saldos (ver Figramoura 16.6). El problema de las recuperaciones inconsistentes metropuede ocurrir cuando una transacción de recuperación se ejecuta simultáneamenteycon una transacción de actualización. No puede ocurrir si la transacción de recuperación se realizametroed antes o después de la transacción de actualización. Una serieyentrelazado equivalentegramode una transacción de recuperación y una transacción de actualización, por ejemploXametropor favor como en Figramoure 16.8, evitará que se produzcan recuperaciones inconsistentesgramo.

Figura 16.8 Un intercalado equivalente en serie de VyW

TransacciónV:	TransacciónW:
a.retirar(100);	aSucursal.totalsucursal( )
b.depósito(100)	
a.retirar(100);	\$100
b.depósito(100)	\$300
	total = a.obtenerSaldo()
	total = total + b.obtenerSaldo()
	total = total +
	c.obtenerSaldo() . . .

**Operaciones conflictivas** • Cuando nos saque un par de operaciones conflictos nosotros metremos que su efecto combinado depende del orden en que son ejecutados. Si metemos en cuenta que las operaciones consideramos un par de operaciones, leer y escribir. Leer accede al valor de un objeto. Escribir cambia el valor. El efecto de una operación se refiere al valor de un objeto conjunto por la operación y el resultado devuelto por la otra operación. Las reglas de conflicto para leer y escribir las operaciones son gramáticas en Figura 16.9.

Por un par de transacciones, es posible disuadir de que el orden de los pares de operaciones conflictivas sea equivalente en serie, es necesario que todos los pares de operaciones conflictivas de las dos transacciones sean ejecutados en el mismo orden en todos los objetos que ambos acceden.

Para que dos transacciones sean equivalentes en serie, es necesario que todos los pares de operaciones conflictivas de las dos transacciones sean ejecutados en el mismo orden en todos los objetos que ambos acceden.

Figura 16.9 Ready write reglas de conflicto de operaciones

operaciones de diferentes actas		Conflicto	Razón
leer	leer	No	BPorque el efecto de un par de leer operaciones no depende del orden en que son ejecutadas
leer	escribir	Sí	Bporque el efecto de un leer y un escribir operación depende del orden de sus ejecuciones
escribir	escribir	Sí	BPorque el efecto de un par de escribir operaciones depende del orden de sus ejecuciones

Figura 16.10 Un intercalado no equivalente en serie de operaciones de transacciones y  $T \cap U$ 

TransacciónT:	TransacciónU:
x = leer (yo)	
escribir (yo, 10)	
	y = leer (j)
	escribir (j, 30)
escribir (j, 20)	
	z = leer (i)

Considerar como una ejecución completa las transacciones  $T \cap U$ , definido de la siguiente manera:

$T:x = \text{leer}(\text{yo}); \text{escribir}(\text{i}, 10); \text{escribir}(\text{j}, 20);$   
 $T:z = \text{leer}(\text{j}); \text{escribir}(\text{j}, 30); z = \text{leer}(\text{i});$

Entonces considere la intercalación gramática de su ejecución, mostradas en Figura 16.10. Tenga en cuenta que el acceso de cada transacción a objects  $i, j, y, z$  se serializa uno con respecto al otro, porque  $T$  toma todos sus accesos a  $i$  antes de que  $U$  tome todos sus accesos a  $j$  y  $z$ . Sin embargo,  $U$  no es equivalente a  $T$ , porque los pares de operaciones conflictivas no se hacen en el mismo orden en ambos objects. La orden de operaciones equivalentes requiere uno de los siguientes dos condiciones:

1.  $T$  accede a  $i$  antes que  $U$  acceda a  $j$

2.  $U$  accede a  $j$  antes que  $T$  acceda a  $i$

La equivalencia en serie se utiliza como criterio para la derivación de la concurrencia y los protocolos de control. Estos protocolos intentan garantizar la serialización de transacciones en su acceso a objects. Tres enfoques alternativos a la concurrencia y control son comilímetros o lo usado: ubicación k-gramo, optimoconcurrentia i-ística y temporal. Sin embargo, es más práctico usar lock-gramo, que se discute en la Sección 16.4. La ubicación de gallinak-gramo se usa, el servidor establece una ubicación  $k$ , etiquetada con el identificador de transacción, en cada object. Justo antes de que se acceda a un object, la transacción que lo ha bloqueado lo libera. Una vez que la transacción ha terminado, el object es liberado, solo la transacción que lo ha bloqueado puede acceder a ese object. Otras transacciones esperan hasta que el object es desbloqueado, o, en el caso de compartir la ubicación  $k$ , el uso de locks puede llevar a un punto muerto. Con transacciones esperando, uno para el otro para liberar locks - para ejemplo, cuando un par de transacciones cada una tiene un object que el otro necesita acceder. Discutimos el punto muerto problemático y entonces recordaremos para ello en la Sección 16.4.1.

Optimometroconcurrentia i-ística y control se describe en la Sección 16.5. En optimoconcurrentia i-ística, una transacción procede hasta que se ha completado una operación antes de que se permita otra. El servidor mantiene un check para descubrir si ha funcionado otra operación en una transmisión exterior que entra en conflicto con las operaciones de otras transacciones concurrentes, en cuyo caso el servidor lo aborta y el cliente reinyecta la transacción. Asegúrate de que todos los objects son correctos.

Temporal estandarizado pgramo se describe en la Sección 16.6. En temporal estandarizado pgramo, un servidor registra el tiempo más reciente de lectura y escritura de cada object y para cada

Figura 16.11 Una lectura sucia cuando se aborta la transacción T

TransacciónT:	Transaccióntu:
a.obtenersaldo()	a.obtenersaldo()
a.setBalance(saldo + 10)	a.setBalance(saldo + 20)
saldo = a.obtenerSaldo() \$100	saldo = a.getBalance() \$110
a.establecerSaldo(saldo + 10) \$110	a.setBalance(saldo + 20) \$130
	transacción de compromiso
abortar transacción	

operación, el timetroestametro de la transacción es cometroemparejado con el de la object para disuadir metroine si se puede hacer yomilímetroediatlyometrodebe ser delayed o rejectadoWcuando una operación es delayed, la transacción espera; cuando es rejectada, la transacción es abortada.

Basí llamadoy, concurrenteyel control se puede lograr ya sea bytransacciones de clientes esperandogramo el uno por el otro o byreiniciargramotransacciones después de que se hayan detectado conflictos entre operaciones, o byun compañerometrocombinación de los dos.

### 16.2.2 Recuperabilidad de abortos

Servidoresmetrosolo registrar todos los efectos de comilímetrotransacciones itted y ninguno de los efectos de las transacciones abortadas. Elmmmpor lo tanto, debemos admitir el hecho de que una transacciónmetroayabortar b yprevenirgramoaffectagramootras transacciones concurrentes si lo hace.

Esta sección ilustra dos problemasmetros asociado con abortingramotransacciones en el conteXt de la prohibiciónkengramomiXametropor favor Estos problemasmetros se llaman 'suciedadylee' y 'premetro ature escribe', y ambos de losmetropuede ocurrir en presencia de seriallyequivalente eXejecuciones de transacciones. Estos temas se refieren a los efectos de las operaciones en objects tales como el saldo de una prohibiciónkuenta. a simetropifydelgadogramos, las operaciones se consideran en dos categoríogramos:leeroperaciones yescribioperaciones. En nuestras ilustraciones,obtenersaldoes un leeroperación y establecer el saldoaescribioperación.

**Lecturas sucias** •La propiedad de aislamientoyde las transacciones requiere que las transacciones no vean el uncomilímetroEstado itted de otras transacciones. La suciedadproblemade lecturametros causado byla interacción entre unleeroperación en una transacción y una anteriorescribioperación en otra transacción en el sametroobject. Considere la eXejecuciones ilustradas en Figramoura 16.11, en la queTgramoestablece el saldo de la cuentaAy lo establece en \$ 10metromineral, entoncestugramoestablece el saldo de la cuenta Ay lo establece en \$ 20metromineral, y los dos eXlas ejecuciones son en seriey equivalente. Supongamos ahora que la transacciónTaborta despuéstiune comilímetroitted Entonces la transaccióntuhabrá visto un valor que nunca eXentendido, ya queAserá restaurado a su origramovalor final.We sayque la transaccióntu tiene rendimientometroed unlectura sucia.Como tiene comilímetroitted, no se puede deshacer.

Figura 16.12 Sobrescribir valores no confirmados

TransacciónT:	Transaccióntu:
a.establecer Saldo(105)	a.establecer Saldo(110)
\$100	
a.establecer Saldo(105)	\$105
	a.establecer Saldo(110)
	\$110

**Recuperabilidad de las transacciones** • Si una transacción (likmitu) tiene comilímetroittd después de haber visto los efectos de una transacción que subsiguentemente abortado, la situación no es recuperable. Para asegurarse de que tales situaciones no se presenten, una transacción (likmitu) eso está en danger de tenergramo una sucedadyleer delay su comilímetro es operación. La estrategia para recuperabilidad es para delay comilímetro es hasta después de la comilímetro el metroent de una otra transacción cuyo uncomilímetro ha observado el estado itted. En nuestro examen por favor, tu delay su comilímetro es hasta después Tcomilímetro es. En el caso de que Taborta, entonces tu comilímetro debe abortar también.

**Cancelaciones en cascada** • En figura 16.11, supongamos que la transacción tu delay su comilímetro ittín gramohasta después Taborta. Como hemos dicho, tu comilímetro debe abortar también. Desafortunadamente, si otras transacciones han visto los efectos debido a tu, el y también tu comilímetro debe ser abortado. El abortogramo de estas últimas transacciones metroay hacer que se anulen aún más transacciones. Este tipo de situaciones se denominan abortos en cascada. Para evitar la cascada gramoaaborta, las transacciones son only permitido leer objects que fueron escritos by comilímetro transacciones itted. Para asegurarse de que este sea el caso, un y leer operación metrodebe ser delayed hasta otras transacciones que aplicaron un escribioperación a la sametro object tener comilímetroittd o abortado. La evitación de la cascada gramoaaborta es un fuerte gramocondición er que recuperabilidad.

**Escrituras prematuras** • Considere otro yometroplicación de la posibilidad que una transacción metroay abortar. Esta está relacionada con la interacción entre escribioperaciones en el sametro object belóngramo en gramoa diferentes transacciones. A modo de ilustración, consideremos dos establecer el saldo actas, Tytu, a cuenta A, como se muestra en Figura 16.12.B Antes de las transacciones, el saldo de la cuenta A era de \$100. Las dos X las ejecuciones son en serie y equivalente, con Tsentado gramoa el saldo a \$105 y usentado gramoa \$110. Si la transacción tu aborta y Tcomilímetrosu, el saldo debe ser de \$ 105.

Entonces metrobase de datos electrónicay systemetros favor metroent la acción de abortar by restaurarg 'antes y metroagramoe' de todos los escrito de una transacción. En nuestro examen por favor, A es \$100 inicialmente, que es el 'antes de que y metroagramoe' de T'scribir; simetraly, \$105 es el 'antes de que y metroagramoe' de tu escribir. Así situabota, nosotros gramoa el saldo correcto de \$105.

Ahora considere el caso cuando tu comilímetro es y luego Taborta. El saldo debe ser de \$110, pero como "antes de metroagramoe" de T'scribires \$100, nosotros gramoy el malgramo saldo de \$100. Simetraly, si Taborta y luego tu aborta, el 'antes de que y metroagramoe' de tu escribires \$105 y nosotros gramoy el malgramo saldo de \$105 - el saldo debe volver a \$100.

Para garantizar resultados correctos en una recuperación es que el metroagramoe que usa antes de imetroagramoes, escribir operaciones metrodebe ser delayed hasta transacciones anteriores que actualizaron el sametro objects tienen comilímetroittd o abortado.