Taller de Sistemas Operativos Avanzados
Práctica #6: Algoritmos de Sincronización de Relojes

Objetivo

Implementar alguno de los algoritmos para la sincronización de relojes (Cristian, Berkeley y Por Promedio) para llevar a cabo periódicamente la sincronización en tiempo entre varias computadoras de tal forma que se garantice el que dos relojes (el más lento y el más rápido) no difieran más de cierta cantidad de tiempo entre ellos, mientras se encuentren en ejecución las aplicaciones de simulación en las máquinas.

Descripción

Independientemente del algoritmo de sincronización de relojes elegido, éste se ejecuta cada determinado tiempo según el máximo deseado de distorsión de reloj (la diferencia de valores de tiempo entre las computadoras).

Considerandos:

a) La distorsión de reloj *d* máxima
b) El valor de la constante *p* para un cronómetro dentro de su especificación, acorde a:

$$1 - \rho \leq \frac{dC}{dt} \leq 1 + \rho$$

c) El cronómetro de una computadora realiza H interrupciones por segundo

Casos:

1) Si contáramos con un reloj perfecto (p=0), si H=10 entonces se determina una interrupción cada 100mseg y tras cada interrupción se añadirían N=100mseg al reloj, de tal modo que en un minuto se obtendrían M=H*60=600 marcas de reloj, o sea, un tiempo de M*N=60,000mseg (60 seg.)
2) Suponiendo un reloj lento con H=10 y p=0.2 (una exageración de p para ejemplificar) entonces se obtienen en el mismo lapso de tiempo 8*60=480 marcas, o sea, un tiempo de 48,000 mseg (según este reloj han transcurrido sólo 48 seg.)
3) Por otro lado, en un reloj rápido también con H=10 y p=0.2 se obtienen 12*60=720 marcas de reloj, o sea, un tiempo de 72,000 mseg (éste dice que han transcurrido 72 seg.)

…con lo anterior tenemos que los dos relojes (el más lento y el más rápido) difieren en 24,0000 unidades de tiempo (24 seg.) al haber transcurrido un minuto, esto se calcula con la fórmula:

$2p(At) = 2(0.2)(60,000) = 24,000$ //difieren en 24 seg.

donde At es el tiempo transcurrido después de que los relojes habían sido sincronizados

Tiempos de los relojes

Reloj lento		Reloj perfecto		Reloj rápido
0		0		0
4,800		6,000		7,200
9,600		12,000		14,400
14,400		18,000		21,600
19,200		24,000		28,800
24,000		30,000		36,000
28,800		36,000		43,200
33,600		42,000		50,400
38,400		48,000		57,600
43,200		54,000		64,800
48,000		60,000		72,000

Sincronización de relojes:

Si deseamos que los relojes de los casos anteriores (el más lento y el más rápido) no difieran más de d=10,000 unidades de tiempo (10 seg), habrá que sincronizarlos cada $d/2p$ segundos, o sea:

$d / 2p = (10,000) / 2(0.2) = 25,000$ //unidades de tiempo según el cronómetro local

…lo anterior significa que justo antes de que transcurran 25,000 unidades de tiempo después de la última sincronización según la información de la máquina local considerándose a sí misma ser la perfecta (lo sea ó no lo sea, según p) iniciará la ejecución del algoritmo de sincronización.

Tiempos de los relojes
(si el “perfecto” sincroniza)

Lento		Perfecto		Rápido
19,200		24,000		28,800
20,000		25,000		30,000

Tiempos de los relojes
(si el “rápido” sincroniza)

Lento		Perfecto		Rápido
14,400		18,000		21,600
16,000		20,000		24,000

Resincronización en tiempos de relojes (no sigue los casos previos)

Caso 1 de reloj adelantado

Reloj modelo		Reloj adelantado (asume que debería tener 24,800)
24,800		25,000
24,900		25,050
25,000		25,100
25,100		25,150
25,200		25,200
25,300		25,300

Caso de reloj atrasado

Reloj atrasado (asume que debería tener 25,000)		Reloj modelo
24,800		25,000
24,950		25,100
25,100		25,200
25,250		25,300
25,400		25,400
25,500		25,500

Requerimientos No Funcionales Generales

La simulación deberá proveer de al menos uno de los 3 tipos de algoritmos para la sincronización:
1) Algoritmo de Cristian
2) Algoritmo de Berkeley
3) Algoritmos con promedio

…y cubrir los siguientes requerimientos explícitos en código fuente y verificados en tiempo de ejecución:

1. Se lleve a cabo la sincronización de relojes entre al menos 3 computadoras.
2. La comunicación entre las computadoras sea mediante datagramas.
3. Sea implementado un hilo de control cuyo trabajo consista en la recepción de mensajes y que procure bloquearse para recibir el siguiente mensaje lo más pronto posible, delegando el procesamiento del mensaje recibido a otro hilo.
4. Sea implementado un cronómetro para cada computadora.
5. El cronómetro realice H interrupciones por segundo (suponiendo cronómetro perfecto con p=0).
6. Cada interrupción añada N milisegundos al reloj según H.
7. Se garantice que dos relojes (el más lento y el más rápido) no difieran más de cierta cantidad de milisegundos entre ellos (establecido en punto 12), durante todo el tiempo que se encuentren en ejecución las aplicaciones de simulación en las computadoras.
8. Se lleve a cabo periódica y automáticamente la sincronización de relojes mediante uno de los 3 algoritmos en el momento oportuno antes de ocurrir la distorsión de reloj máxima, considerando el tiempo estimado para la propagación de mensajes en la red y consultando el cronómetro local para determinar el momento de inicio del algoritmo.
9. Apegarse a la bibliografía de Sistemas Operativos Distribuidos pag.127 a la 131 para la especificación de los algoritmos
10. Independientemente de si el tiempo deba aumentar o disminuir, como resultado de ejecutar el algoritmo, el cambio en el reloj sea gradual y nunca establecer un tiempo menor al actual en el reloj.

Requerimientos Visuales y Funcionales

11. Interfaz gráfica por máquina en la cual se proporcione
    a) Área de texto para monitorear los sucesos relacionados a la sincronización (arrivo de mensajes, envíos, datos en el mensaje, cálculos, resultados, decisiones)
    b) Campos de texto independientes para cada uno de los siguientes parámetros: el tiempo actual de la máquina (en milisegundos), actualizado tras cada interrupción de reloj; el valor de H; el valor actualizado de N tras cada trabajo de sincronización; la distorsión de reloj d máxima; el valor de la constante p; el intervalo de tiempo $d/2p$ calculado para iniciar la sincronización; tiempo estimado para la propagación de mensajes en la red actualizado tras cada sincronización de relojes

12. Se permita definir manualmente a través de la interfaz gráfica:
    a) El tiempo de reloj inicial
    b) La distorsión de reloj *d* máxima.
    c) El valor de H (según el punto 5)
    d) Si el reloj en la máquina es rápido, lento, o perfecto
    e) El tiempo estimado inicial para la propagación de mensajes por la red en miliseg.
    f) El valor de la constante *p* para un cronómetro dentro de su especificación, de modo que dicha constante determine la frecuencia de las interrupciones acorde a:
    $1-p <= dC/dt <= 1+p$

Requerimientos No Funcionales Particulares

1) Algoritmo de Cristian

a) Se utilice la plataforma MicroNúcleo operativa a la fecha con sus primitivas send(dest,messagePtr) y receive(addr,messagePtr) para paso de mensajes
b) Implementación para los dos tipos de máquinas (cliente y servidor de tiempo)
c) Fijar manualmente en el servidor de tiempo, mediante su interfaz gráfica, el tiempo del controlador de interrupciones
d) Fijar manualmente en el servidor de tiempo la cantidad de milisegundos para simular tanto el retardo en recibir la solicitud como el envío de la respuesta al cliente
e) Solicitud del tiempo por parte del cliente acorde al punto 8
f) Iniciar el procesamiento de la solicitud por parte del servidor de tiempo después de transcurrida la cantidad de milisegundos especificada en el inciso “d” anterior
g) Procesamiento de la solicitud por parte del servidor de tiempo tardando al menos la cantidad de milisegundos especificada en el inciso c anterior
h) Envío de respuesta al cliente incluyendo el tiempo simulado UTC y el de la constante del controlador de interrupciones justo después de transcurrido el tiempo según el inciso “d”
i) Cálculo en la máquina cliente, del tiempo a asumir considerando el tiempo de propagación por la red según su propio reloj y el tiempo del controlador de interrupciones
j) Establecimiento automático del valor de tiempo estimado para la propagación de mensajes por la red a considerar para la siguiente resincronización
k) Aplicación del cambio gradual en el tiempo del reloj local

2) Algoritmo de Berkeley

a) Implementación para los dos tipos de computadoras (servidor de tiempo y servidores complementarios)
b) Alta manual o automática de la membresía de grupo en el servidor de tiempo
c) Envío del tiempo por parte del servidor de tiempo a los servidores complementarios, e incluyéndose a sí mismo, acorde al punto 8
d) Procesamiento de la solicitud recibida en todos los servidores y envío de la respuesta con la diferencia de tiempo respecto al servidor de tiempo
e) Procesamiento de la respuesta/solicitud de tiempo de ajuste acorde al cálculo del promedio de todas las diferencias recibidas por el servidor de tiempo

f) Envío del ajuste en tiempo correspondiente a cada una de las máquinas del grupo
g) Cálculo en todas las computadoras, del tiempo a asumir acorde al mensaje recibido según el inciso anterior
h) Aplicación del cambio gradual en el tiempo del reloj local

3) Algoritmos con promedio

a) Implementación para operar de forma distribuida
b) Establecimiento automático de los intervalos de resincronización, donde el $i$ésimo intervalo va desde $T_0 + iR$ y va hasta $T_0 + (i+1)R$
c) Alta manual del parámetro de sistema R, con la única restricción de que éste no provoque exceder la distorsión de reloj máxima según el punto 8
d) Establecer de forma manual el valor del intervalo S para reunir las transmisiones de las máquinas
e) Fijar manualmente el tiempo de reloj en cualquier momento (inicial y durante su ejecución) sin que esto afecte en tiempo para realizar la siguiente resincronización según el reloj de la máquina local
f) Alta manual o automática de la membresía de grupo en cada máquina
g) Transmisión del tiempo local a todas las máquinas del grupo, e incluyéndose a sí misma, acorde al punto 8
h) Inicio de cronómetro local para reunir las transmisiones de todas las máquinas del grupo, incluida sí misma, que lleguen en un intervalo S
i) Cálculo del tiempo a asumir acorde a los mensajes recibidos según el inciso anterior
j) Aplicación del cambio gradual en el tiempo del reloj local

Criterios de evaluación

? Calificación dependiente de completar la práctica #5.
? Los establecidos en las “Reglas de Operación y Evaluación” del Taller de Sistemas Operativos Avanzados y los correspondientes “Periodos de Entrega” de la práctica con especial atención al punto #10 (prácticas seriadas)
? Entregar proyecto sistemaDistribuido hasta la fecha, incluyendo como subpaquete las clases involucradas con la práctica e indicando en la parte superior de cada archivo: nombre, sección y no. de práctica; entregado vía e-mail en formato zip.
? Fecha de asignación de la práctica: 27 de Mayo de 2009
? Fecha límite para entrega de la práctica completa: 08 de Junio de 2009
? Observación: Se puede trabajar simultáneamente mientras terminan prácticas 4 y 5 dadas las 3 semanas de contingencia de las que solo se recupera 1
? Evaluación por cobertura de requerimientos y fecha de entrega.