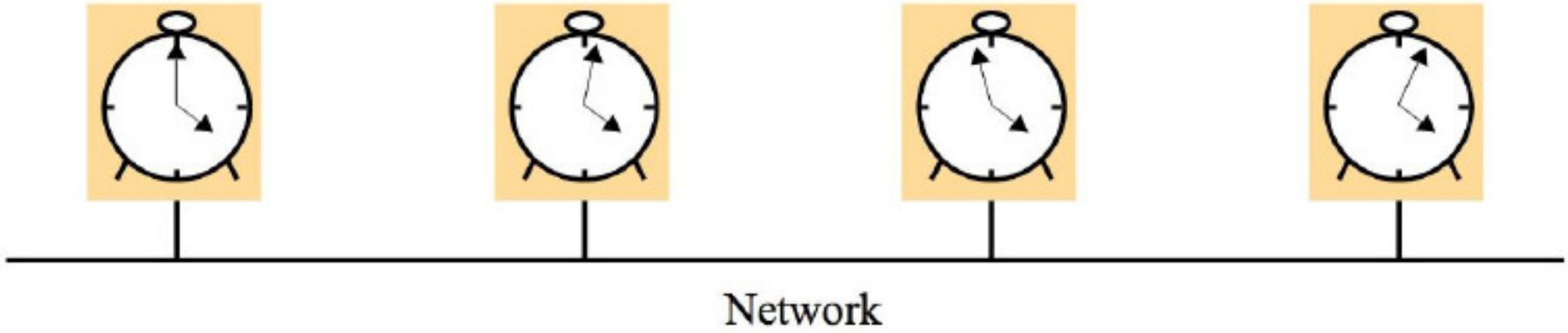
Fakultas Informatika
School of Computing
Telkom University

CSG3L3
SISTEM TERDISTRIBUSI

Topik 14 : Pewaktuan dan Kondisi Global

Ilustrasi Perbedaan Waktu Komputer pada Sebuah Sistem Terdistribusi

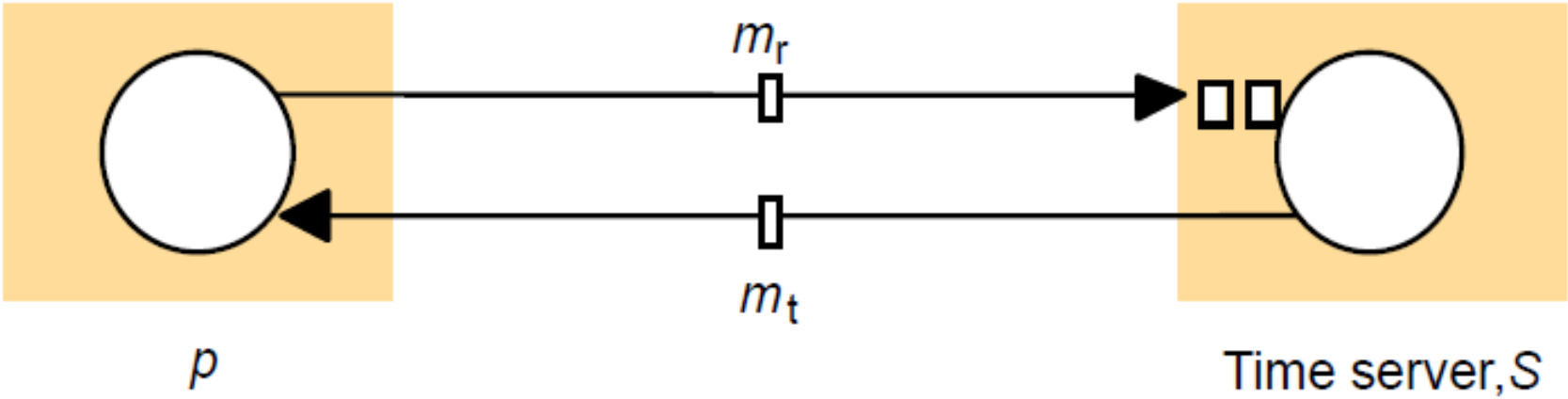


Apa pengaruh dari kondisi tersebut?

Pembahasan:

- Membahas mengenai pewaktuan terhadap *event* saat proses mengeksekusi *event* tersebut.
- Contoh kasus nyata adalah pada *e-commerce*, sistem harus mencatat waktu tiap transaksi yang melibatkan waktu pada komputer penjual, komputer bank, dan komputer pembeli.
- Karakteristik sistem terdistribusi **umumnya** – ***tidak ada sistem / notasi waktu global***
- Beberapa algoritma sinkronisasi

Solusi Perbedaan *Clock / Time*

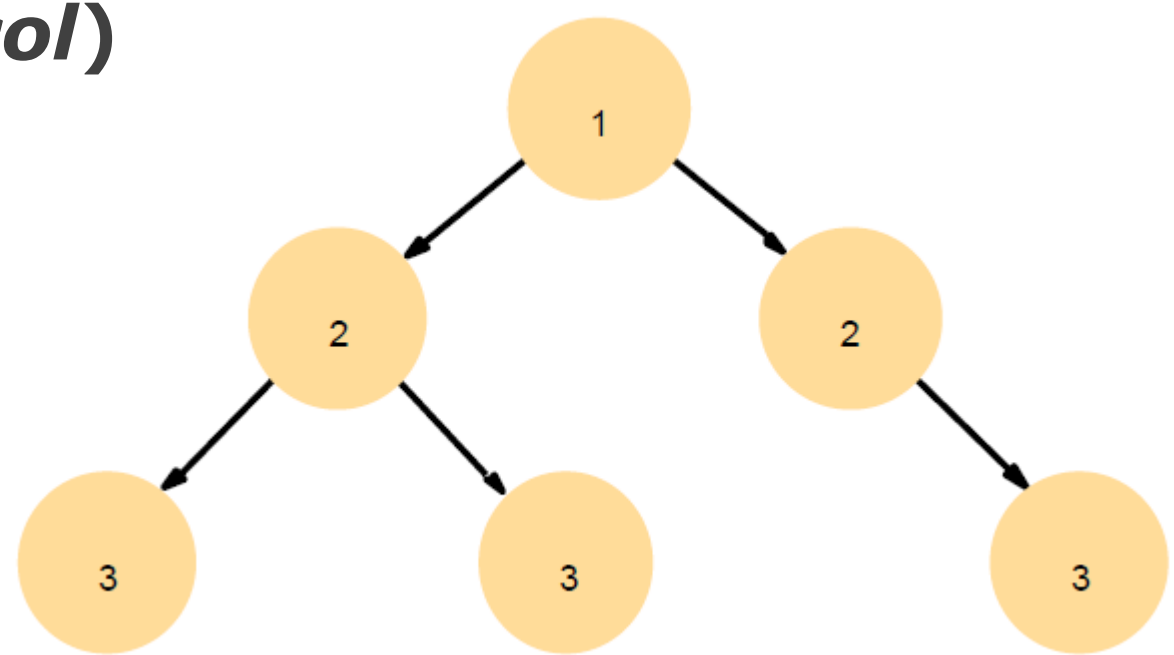


Sinkronisasi waktu menggunakan waktu *server* – *"Cristian's method"*

Penjelasan *Cristian's* method / Sinkronisasi *timeserver*

- Proses *p* meminta waktu server melalui pesan m_t dan menerima pesan balasan berupa waktu server *t* pada pesan m_t.
- Proses *p* merekam total RTT (*Round Trip Time*) atau T_{round}. Waktu yang didapat proses *p* yaitu $t + (T_{round} / 2)$

Contoh Sinkronisasi Sub-jaringan pada Implementasi NTP (*Network Time Protocol*)

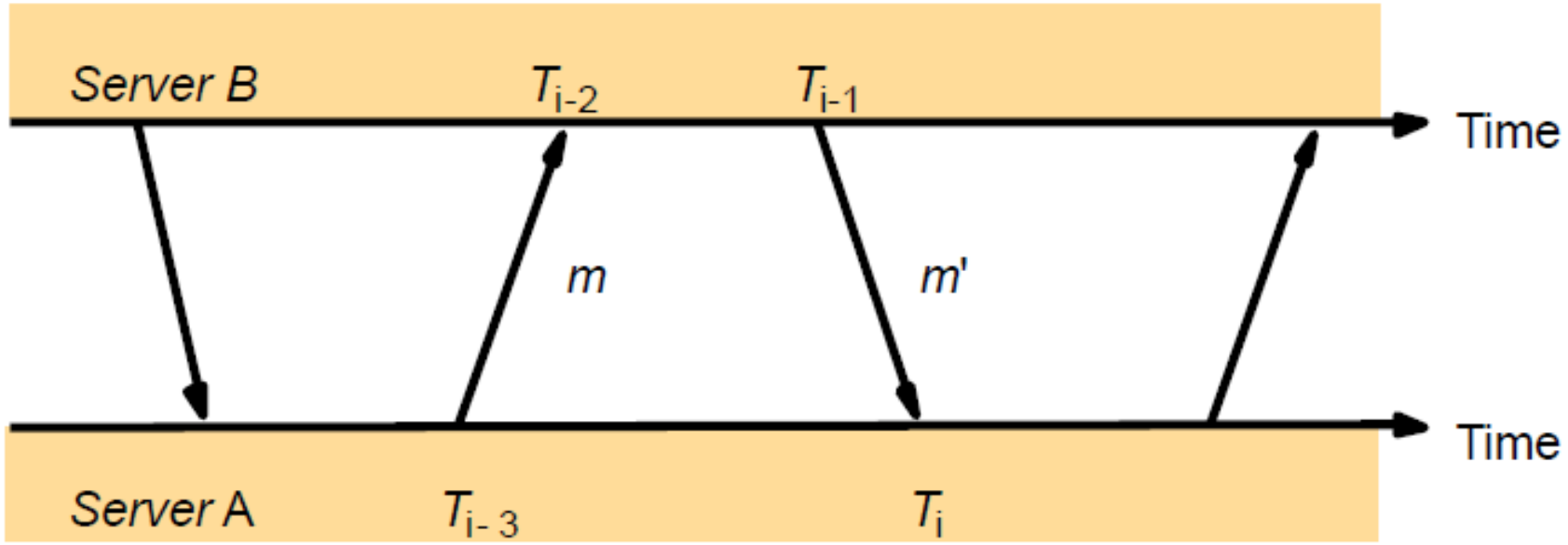


Tanda panah menunjukkan kontrol sinkronisasi, nomor menunjukkan tingkatan

Penjelasan Arsitektur NTP

- *Root* adalah *primary server* (server utama)
- Tiap server terhubung satu sama lain menggunakan hirarki *logic* yang disebut *synchronization subnet* terdiri dari beberapa level. Level 1 (utama), level 2 (sekunder), dan seterusnya hingga level paling bawah (*leaf node*) merupakan *user workstation* (host).

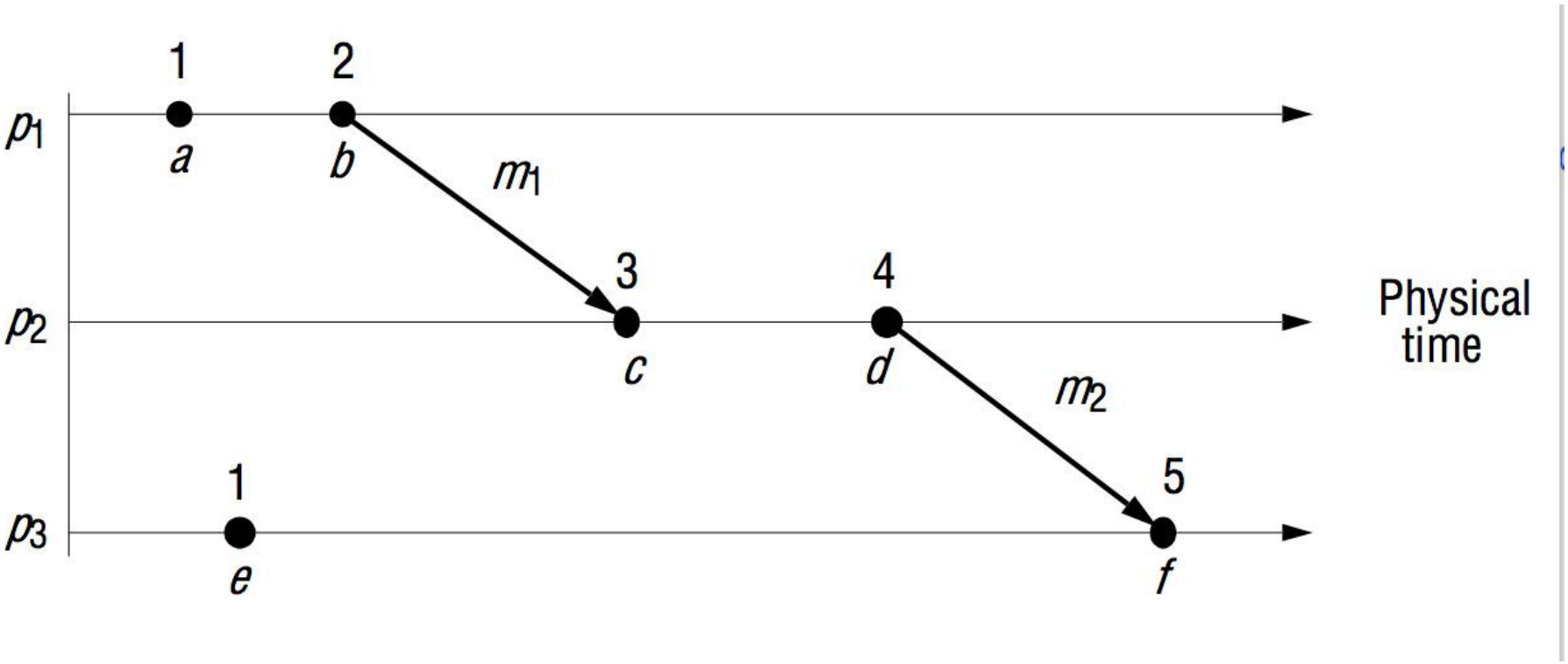
Pertukaran *message* antara dua buah *node* pada NTP



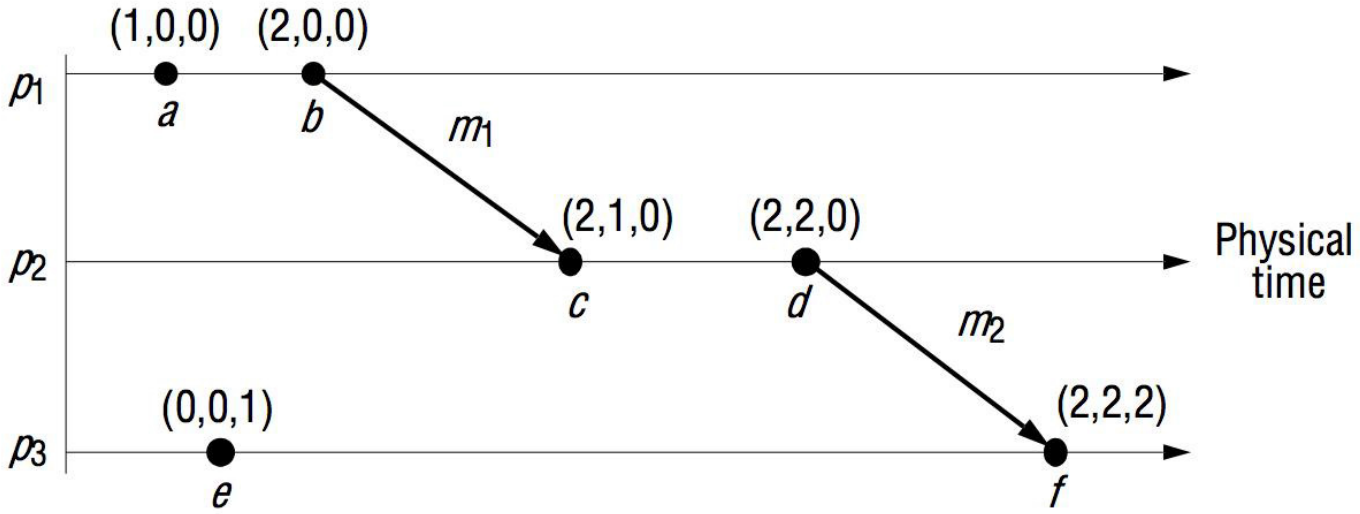
Logical Time dan *Logical Clock*

- Sebuah proses memiliki sederetan *event* yang dieksekusi secara sekuensial (urut berdasarkan waktu eksekusi lokal (*local time*)).
- Lamport menemukan sebuah mekanisme disebut dengan relasi H-B (*Happened-before*). Jika ada proses p_i : e $\rightarrow_i$ e′, maka e $\rightarrow$ e′. Untuk tiap pesan *m,* send(*m*) $\rightarrow$ receive(*m*). Jika e, e′, e″ adalah sederetan *event* dan e $\rightarrow$ e′, e′ $\rightarrow$ e″, maka e $\rightarrow$ e″.

Lamport's Timestamp



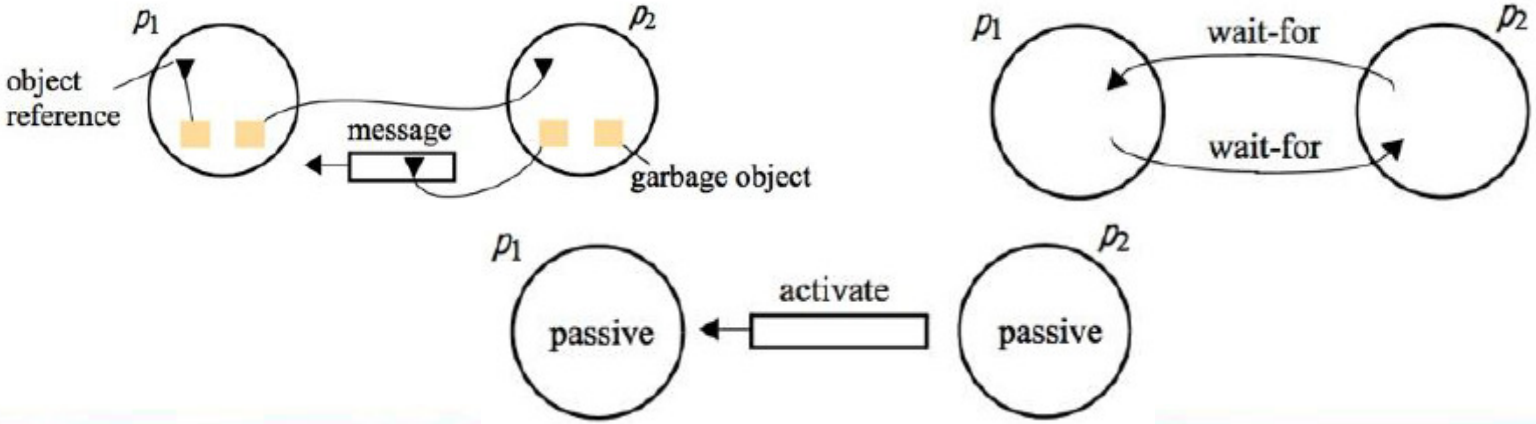
Vector Timestamp



Solusi untuk kelemahan *Lamport's timestamp*. Jika ada L(e)<L(e') belum tentu e → e'. *L* adalah *logical time* dari sebuah *event*.

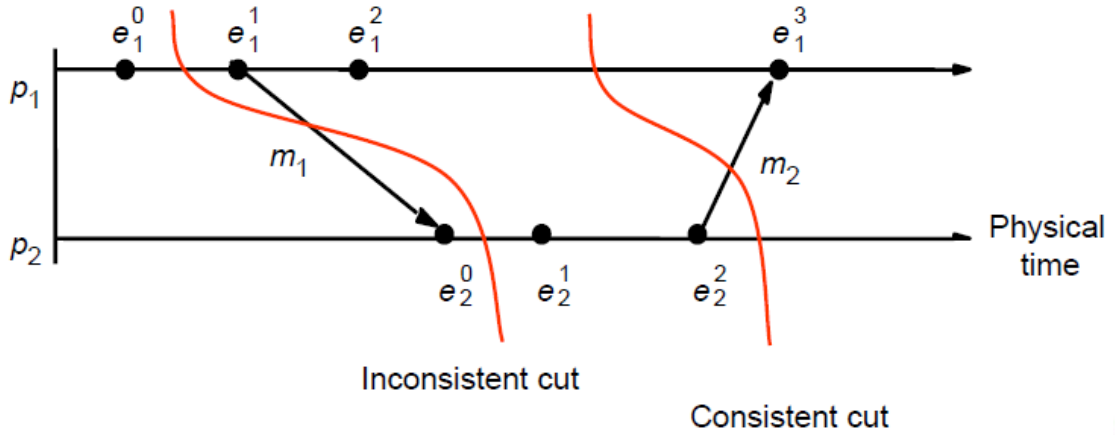
Kondisi Global (*Global States*)

- Tujuannya adalah untuk mengecek sebuah kondisi bernilai *true* atau *false* pada sebuah sistem terdistribusi saat eksekusi sebuah proses.
- Kondisi global digunakan untuk beberapa kasus seperti: *garbage collection*, *deadlock detection*, *termination detection*, dan *distributed debugging*.



Global States dan "*Cuts*"

- Kondisi global bisa didapat dari kondisi lokal (*local state*) yang telah direkam *(history*).
- Sebuah *history* proses p_i didefinisikan sebagai: $history(p_i) = h_i = <e_i^0, e_i^1, e_i^2, e_i^k>$
- *Global state* (H) = $h_0 \cup h_1 \cup \ldots \cup h_{N-1}$.



Algoritma 'Snapshot' Chandy & Lamport

Marker receiving rule for process p_i

On receipt of a *marker* message at p_i over channel c:

if (p_i has not yet recorded its state) it

records its process state now;

records the state of c as the empty set;

turns on recording of messages arriving over other incoming channels;

else

p_i records the state of c as the set of messages it has received over c since it saved its state.

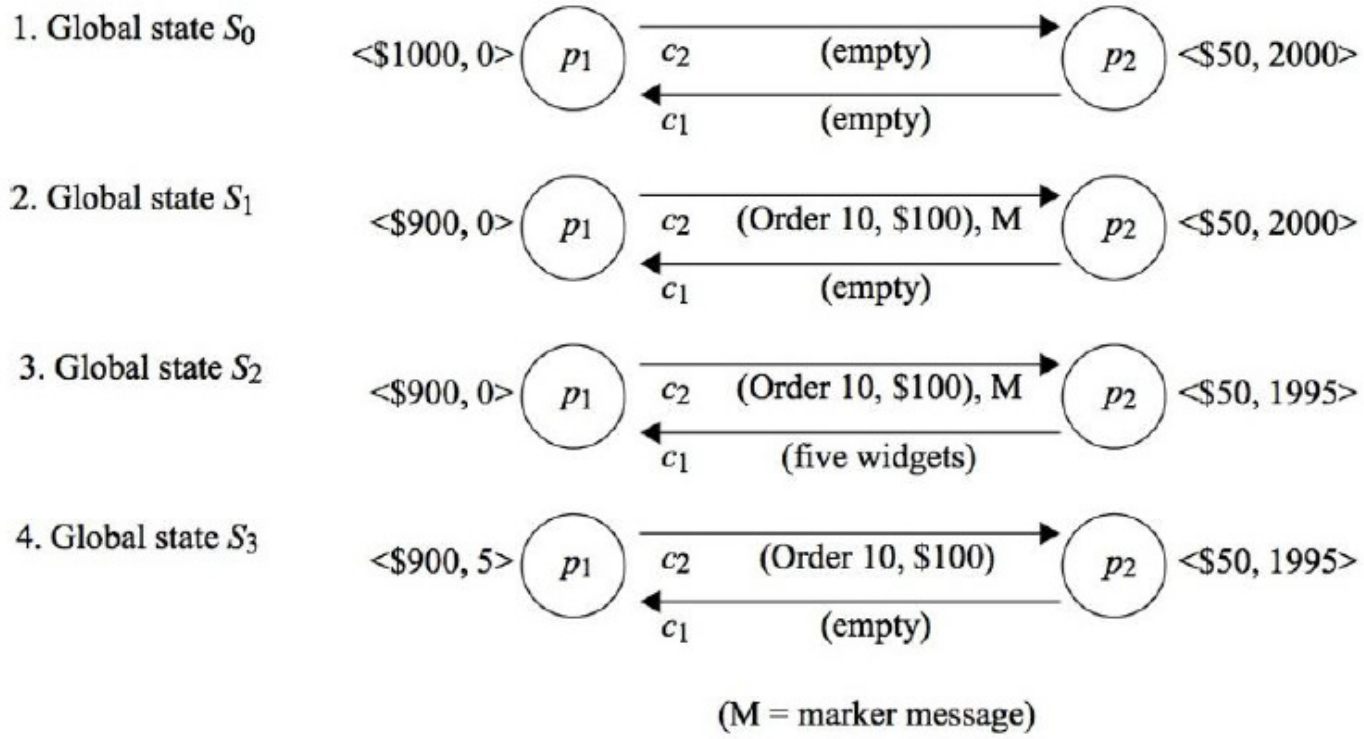
end if

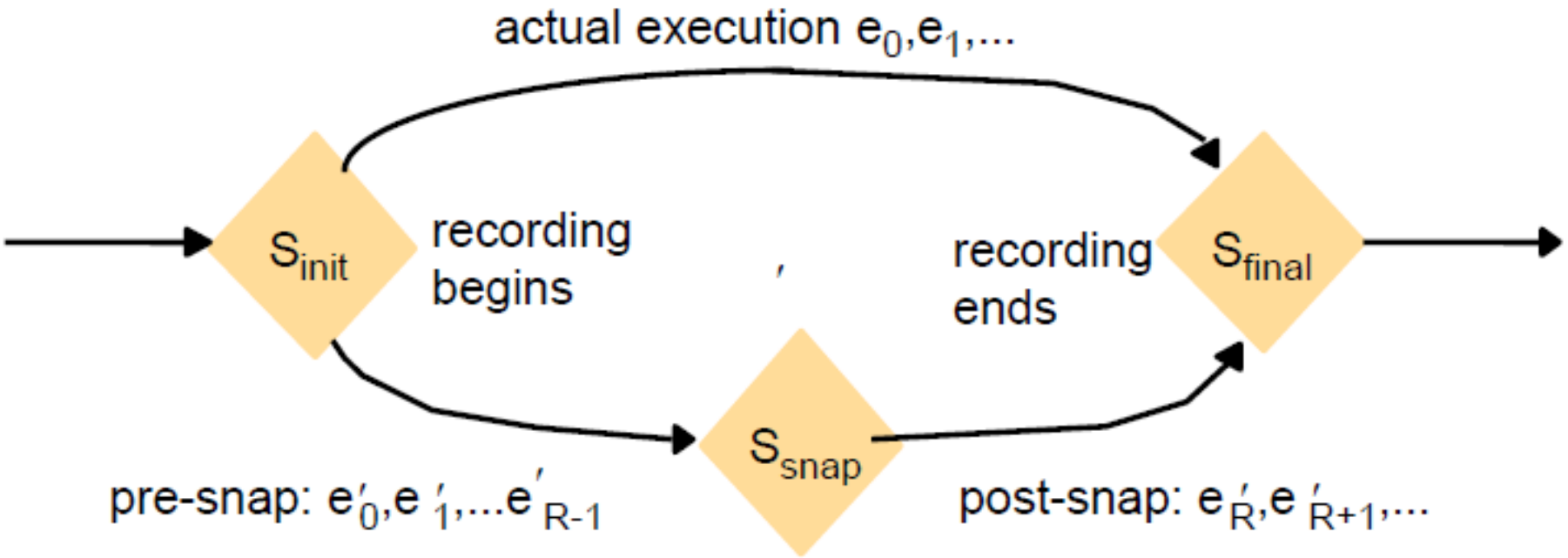
Marker sending rule for process p_i

After p_i has recorded its state, for each outgoing channel c:

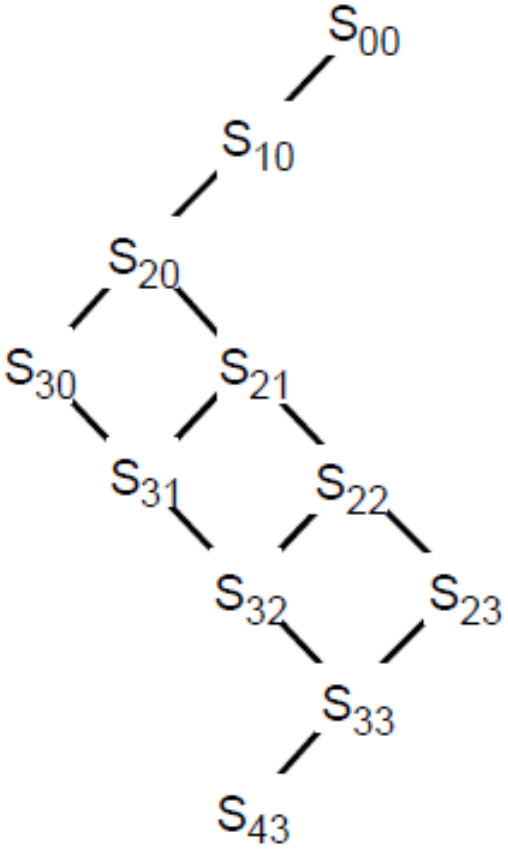
p_i sends one marker message over c (before it sends any other message over c).

Ilustrasi urutan eksekusi dari transaksi pembelian 5 buah *widget* menggunakan Algoritma Chandy & Lamport




actual execution $e_0, e_1, \ldots$
S_{init}
recording begins
recording ends
S_{final}
S_{snap}
pre-snap: $e'_0, e'_1, \ldots e'_{R-1}$
post-snap: $e'_R, e'_{R+1}, \ldots$

Status Global dari Dua Buah Proses



S_{ij} = Status global setelah *i* buah *event* terjadi pada proses 1 dan *j* buah *event* terjadi pada proses 2.

Algoritma Evaluasi Kemungkinan *(Possibly)* dan Kepastian (*definitely*)

1. *Evaluating possibly* ϕ *for global history H of N processes*

$L := 0;$
$States := \{ (s_1^0, s_2^0, \ldots, s_N^0) \};$
while $(\phi(S) = False$ for all $S \in$ States$)$
    $L := L + 1;$
    $Reachable := \{S' : S'$ reachable in H from some $S \in$ States $\wedge$ level$(S') = L\};$
    $States := Reachable$
end while
output "*possibly* ϕ";

2. *Evaluating definitely* ϕ *for global history H of N processes*

$L := 0;$
if $(\phi(s_1^0, s_2^0, \ldots, s_N^0))$ *then* $States := \{\}$ *else* $States := \{ (s_1^0, s_2^0, \ldots, s_N^0) \};$
while $(States \neq \{\})$
    $L := L + 1;$
    $Reachable := \{S' : S'$ reachable in H from some $S \in$ States $\wedge$ level$(S') = L\};$
    $States := \{S \in Reachable : \phi(S) = False\}$
end while
output "*definitely* ϕ";

Ada Pertanyaan?

Referensi

- Coulouris, G. F., Dollimore, J., & Kindberg, T. (2012). *Distributed Systems: Concepts and Design 5th Edition*. London: Pearson Education.

Fakultas Informatika
School of Computing
Telkom University

THANK YOU