EE 382V: Parallel Algorithms **Summer 2017**

Lecture 2: September 5

Lecturer: Vijay Garg *Scribe: Huy Doan*

2.1 Logical Clock

2.1.1 Definition

A map $C : E \to$ is a logical clock if $\forall e, f \in E : e \to f \Rightarrow C(e) < C(f)$.

2.1.2 Logical Clock Algorithm

Let c be the counter.
In-process: $c{+}{+}$
Send: $c{+}{+}$, piggy back c with the message
On receive: with the current number d, $c = max(c, d) + 1$

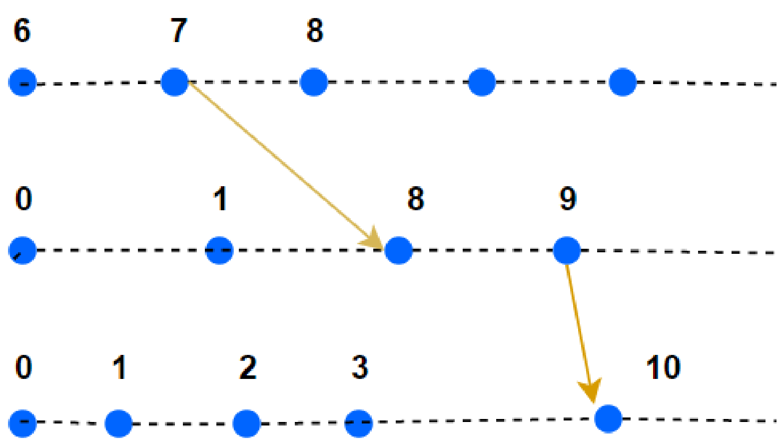

Figure 2.1: Logical Clock

Note:

- given the logical time of two events, we can only make some certain conclusion. For example, events e and f that have $C(e) = 5$ and $C(f) = 7$, there are two possibilities: $e \to f$ or $e \parallel f$.
- the combination $(logicaltime, processid)$ is usually used in comparision to resolve the conflict when two proceses have the same logical time. The comparision follows lexicongraphic order.

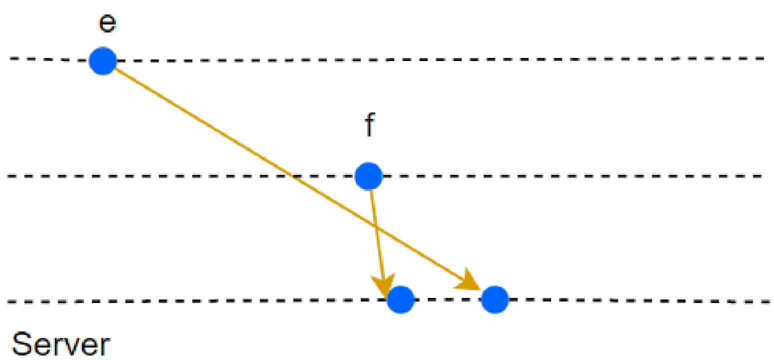

Figure 2.2: Logical Clock is needed to determine which event happens first

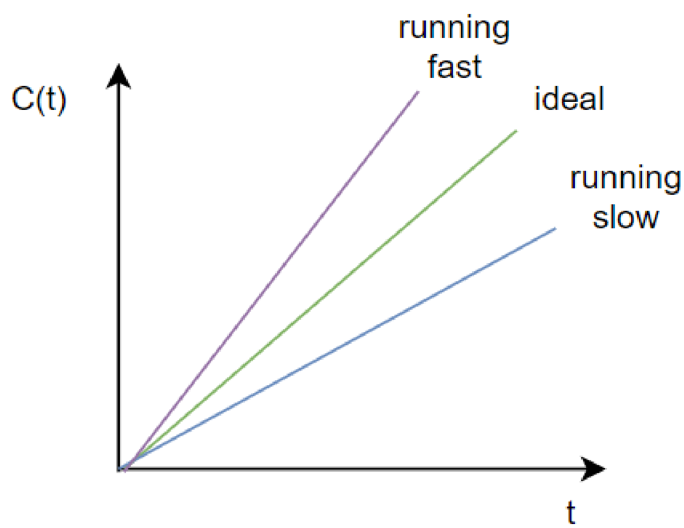

Figure 2.3: Physical clock rates

2.2 Physical Clock

$|1 - dC_i/dt| < \kappa \quad \forall i, \kappa \ll 1$: drift rate
τ: synchronization period
$|C_i(t) - C_j(t)| < \epsilon \quad \forall i, j$: synchronization condition

$G = (E, V)$ undirected graph representing the topology in 2.4.
Let d be the diameter of the graph G. $d = max_{i,j}$ length of the shrotest path from i to j

2.2.1 Lamport's Physical Clock Algorithm

Assume that every message takes time in $[\mu, \mu + \xi]$
Every τ units of time, send your clock value to neighbors.
On receving message timestamped with d, $c = max(c, d + \mu)$

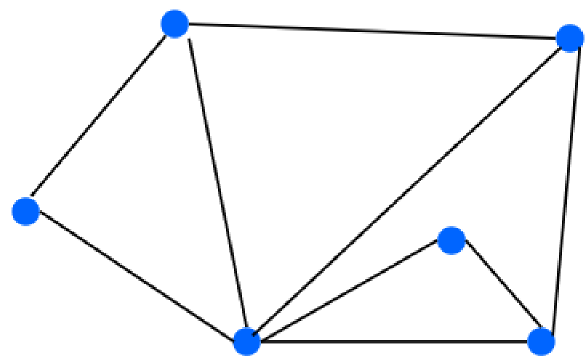

Figure 2.4: Process topology

$\epsilon = d(2\kappa\tau + \xi)$

Physical clock condition gives logical clock for free.

2.3 Down-sets

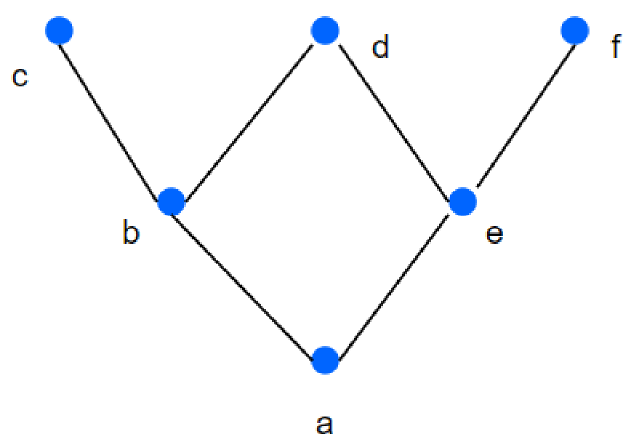

Figure 2.5: Downset example

Let $(X, \leq)$ be a poset.
$Y \subseteq X$ is a down-set of $(X, \leq)$ if $\forall x, y \in X : (y \in Y) \wedge (x \leq y) \Rightarrow x \in Y$

Example: $\{a, b\}$ and $\{c, e, b, a\}$ are down-sets while $\{b\}$ is not.

2.4 Principle Ideals

$D[x]$ is the down-set of x and $V(x)$ is the vector of x.
$D[e] = \{a, d, e\}$ can be characterized by number of events included from each process, i.e. $D[e] = (1, 2, 0)$

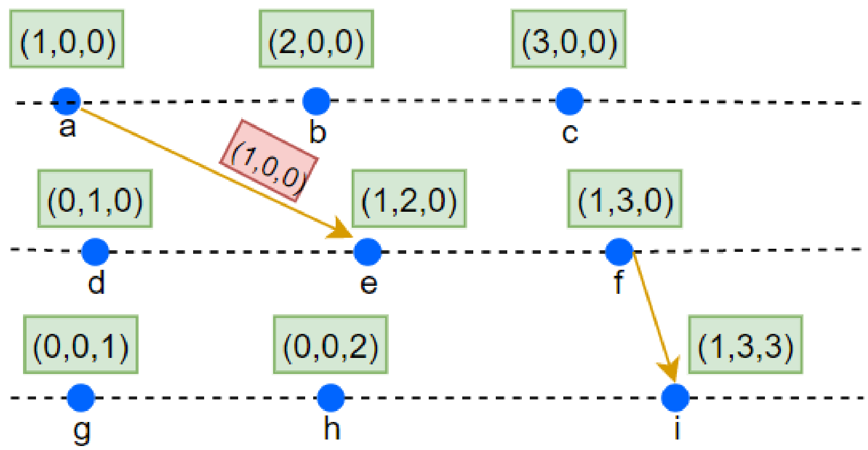

Figure 2.6: Vector clock

Theorem: $e \to f \iff D[e] \subseteq D[f] \iff V(e) \le V(f)$
Definition: Given w, x are vectors. $w \le x \equiv \forall i : w(i) = x(i)$

Proof:

- $\Rightarrow$: $e \to f \Rightarrow D[e] \subseteq D[f]$ because $z \to e \Rightarrow z \to f \forall z$
- $\Leftarrow$: $e \not\to f \Rightarrow D[e] \not\subseteq D[f]$ because $e \in D(e)$ but $e \notin D(f)$

2.5 Mattern and Fidge's Vector Clock Algorithm

Initialize such that : $\forall i \quad (V(i) = 1) \wedge (V(j) = 0) \quad \forall j \neq i$
On any event: $V(i) + +$
On send: include vector lock in the message
On receive: take max

2.6 Dilworth's Theorem

Any poset can be decomposed into w chains where w is the width of the poset.
Consider the largest antichain of size w
$A = \{x \in C_i | x$ is the largest element that belongs to some anti chain of size $w\}$

References

[AGM97] N. Alon, Z. Galil and O. Margalit, On the Exponent of the All Pairs Shortest Path Problem, *Journal of Computer and System Sciences* **54** (1997), pp. 255–262.

[F76] M. L. Fredman, New Bounds on the Complexity of the Shortest Path Problem, *SIAM Journal on Computing* **5** (1976), pp. 83-89.