CS 6410
09nov2010
Time

Vainstein K.

Chocolate to Motivate the Discussion

- please take 1 each during 0^{th} traversal
- may commence eating, once you participate
- milk chocolate: in red wrappers
- really seriously very bitter dark chocolate: ***small*** bars, in black wrappers
- What are nibs? Cleaned, roasted, winnowed, and lightly crushed cacao beans
- no, I'm not providing cognac-drizzled ice cream and fresh raspberries "to go with"; just eat it as is

http://www.thespicehouse.com/spices/roasted-cacao-nibs

What is Time?

- *"the player that need not cheat to win"* –Baudelaire, tr. Edna St. Vincent Millay
- *an abstraction that determines the ordering of events in a given temporal frame of reference* –Mills
- *partial ordering on events in a distributed system* – Lamport
  - why partial? Two events are "concurrent" if we have no way of knowing which happened first
  - causality: A happened before B => A may have caused B

Mills, D.L. Network Time Protocol (Version 3) specification, implementation and analysis. Network Working Group Report RFC-1305, University of Delaware, March 1992

Time, Clocks, and the Ordering of Events in a Distributed System, Lamport. CACM 21(7). July 1978.

Why is Time of Interest in Distributed Systems?

- coordination of action (e.g. snapshot)
- providing realtime guarantees
- given synchronized clocks, many distributed algorithms are simpler (no timeout quandary)
- without synchronized clocks, local clock at least affords timeout
- distributed filesystems need to detect conflicts

Clocks

- logical clock (a.k.a. virtual clock): a counter
- physical clock: periodic oscillator that increments a counter
  - approximates *real time*, which is the actual (Newtonian) time; f: real time → clock time
- offset: time difference between two clocks, Ω
- skew: change in offset wrt continuous time, dΩ/dt
  - some authors call this "drift"
- drift: $d^2\Omega/dt^2$
- accuracy: how close to real time
- precision: (of multiple readings) how close together

Gunther N. J., The Practical Performance Analyst, McGraw-Hill 2000

Clock Synchronization

- external clock synchronization: conformance of each node's clock with an RT clock external to system; e.g., NTP
- internal clock synchronization: movement of each node's clock to "majority consensus", minimizing inter-node skew
  - convergence (gradual)
  - agreement (immediate)
- instantaneous resynchronization inadvisable

Christoph Lenzen, Thomas Locher, and Roger Wattenhofer. 2010. Tight bounds for clock synchronization. *J. ACM* 57, 2, Article 8 (February 2010)

Structure of Main Talk, per Paper

- try to stay high-level
- objectives?
- claims?
- assumptions?
- results?
- context?

Optimal Clock Synchronization, Srikanth and Toueg

- Sam Toueg: Cornell CS faculty
- emphasis on fault tolerance: # of incorrect servers, and kinds of incorrectness (incl. Byz.)
- objective: accuracy of logical clocks same as (no worse than) that of physical clocks
  - claim solution is optimal wrt accuracy
- assume:
  - physical clock skew ρ, bounded, constant, $|\rho| > 0$
  - 0 drift
  - reliable fully-connected point-to-point network

Srikanth and Toueg, continued...

- basic algorithm, roughly:
  - at a boundary, server sends "I want to synchronize"
  - when receives enough of such messages, resets its logical clock by adding α (greater than propagation delay)
- $O(n^2)$ messages per resynchronization
- this is internal synchronization
- improvements claimed possible:
  - messages not authenticated
  - network not fully connected

Probabilistic Internal Clock Synchronization, Cristian and Fetzer

- Flaviu Cristian: d. 1999
- Christof Fetzer: PhD UCSD 1997, AT&T Labs 1999-2004, Pfsr at T U Dresden 2004+
- this is internal synchronization (cf. title)
- claimed improvements on older approaches:
  - O(n) messages, with "transitive" clock reading scheme
  - message exchanges scattered in time (not bursty)
  - optimal (minimal) skew of logical clocks

LinkedIn.com database
http://www.computer.org/portal/web/csdl/doi/10.1109/DCCA.1999.10003

Cristian and Fetzer, continued... (2 of 3)

- what is a probabilistic method?

"to prove existence of a combinatorial structure with certain properties, we construct an appropriate probability space, and show that a randomly chosen element of this space has the desired property, with positive probability"

"no bound on clock reading error"

- reading a remote clock is probable with some 0<P<1
- assume:
  - bounded skew, 0 drift
  - bounded and small initial accuracy of all the clocks
- do *not* assume:
  - bounded communication delay
  - reliable channels

http://www.math.ucla.edu/~bsudakov/probabilistic.html

Cristian and Fetzer, continued... (3 of 3)

- processes exchange their estimates for other clocks' error bounds
- remote clock responds to request with complete state (recv. history, etc); large!
- a separate algorithm for different failure assumptions: crash only, read only, hybrid
- use broadcast on LANs to further reduce # of messages sent

Using Time Instead of Timeout for Fault-Tolerant Distributed Systems, [1984]

Lamport

- Leslie Lamport: PhD Brandeis 1972, at MSFT Research since 2001
- assume:
  - physical clocks are already synchronized
  - time to generate and transmit message, δ, is constant, which should hold given no network congestion or CPU/disk contention
    - (can choose a constant that will always be big enough)
  - clocks' skews are within epsilon of each other
  - process can determine true source of message (guard against rogue Byzantine servers spoofing messages)

Lamport, continued... (2 of 3)

- normally ("traditional timeout"), not receiving a message could mean [a] network delay, or [b] crash
- with synchronized clocks and bounded propagation delay, not receiving a message (or: receiving a NULL message) can mean "I have failed", or another positive statement
- state machine: replicating process actions (≈SIMD)
  - apply operations to replicas atomically
    - can use to implement active replication
  - seems to assume failstop? Lamport: "can design around" ;-|
- distributed semaphore: another problem now easy

Lamport, continued... (3 of 3)

- resource allocation: "synchronize access to a shared resource by N processes so that only one process at a time can use it". Like in "Time, clocks, and the ordering of events in a distributed system", but also want fault-tolerance.
  - "before" means "occurring earlier in time", since Lamport's old "|-->" relation requires messages
- performance: "traditional timeout" only has response time advantage when SW has complete low-level control over network driver

Understanding protocols for Byzantine Clock Synchronization, Schneider

- Fred Schneider: Cornell CS faculty, distinguished researcher in distributed computing since late 1970s
- paper provides unified view of all fault-tolerant (internal) clock synchronization protocols
- unified view: servers reset their logical time in response to event emitted by "reliable time source"
  - which must be distributed, for fault-tolerance
- network delay uncertainty largely due to uncertainty in program scheduling (blame multiprogramming, interrupts)
  - put clock reading into μ-kernel, reduce this uncertainty

Schneider, continued... (2 of 2)

- properties of a convergence function:
  - monotonicity (nondecreasing successive values)
  - translation invariance (all clocks shifted by same ν)
  - precision enhancement
  - accuracy preservation
- to implement reliable time source, must solve:
  - make all processes synchronize within elapsed β
  - read clock of another processor, within error Λ
  - choose convergence function satisfying the above

Network Time Protocol (NTP) [1985], Mills

- David Mills: PhD U Mich 1971, Pfsr at U Delaware from 1986
- primary reference clocks slaved to authoritative hardware clocks, by hardware (EMR/optic) means
  - placed into gateways (major ISP switches)
- provide time to LAN-level hosts (2ry reference clocks)
  - which would redistribute time to rank-and-file hosts
- this is an external synchronization protocol
- client sends UDP request to clock, clock immediately returns it with latest update time, estimated drift, (originate|receive|transmit) timestamp
- symmetric mode also possible
- client (or peer) calculates RTT, and hence clock offset

http://www.ece.udel.edu/~mills/