The $\varphi$ accrual failure detector

Presenter
Dat Tien Le
dattl@ifi.uio.no

INF5360: Seminar on Dependable and Adaptive Distributed Systems

Department of Informatics
University of Oslo

April 15, 2015
Papers to present

- The $\phi$ accrual failure detector
  - By Naohiro Hayashibara, Xavier Defago, Rami Yared and Takuya Katayama
The φ accrual failure detector

- An implementation of an accrual failure detector.

- Topics covered:
  - Failure detection
  - Statistics
Definition

- Failure detector is an application or a subsystem that is responsible for detection of node failures or crashes in a distributed system.
Classification of failure detectors

- **Degree of completeness**
  - **Strong completeness**: every faulty process is eventually permanently suspected by every non-faulty process
  - **Weak completeness**: every faulty process is eventually permanently suspected by some non-faulty process.

- **Degree of accuracy**
  - **Strong accuracy**: no process is suspected before it crashes
  - **Weak accuracy**: some non-faulty process is never suspected
  - **Eventually strong accuracy**: after some initial period of confusion, no process is suspected before it crashes
  - **Eventually weak accuracy**: after some initial period of confusion, some non-faulty process is never suspected
Classification of failure detectors

- Failure detector classes
  - P (Perfect): strongly complete and strongly accurate
  - S (Strong): strongly complete and weakly accurate
  - ◊P (eventually perfect): strongly complete and eventually strongly accurate
  - ◊S (eventually strong): strongly complete and eventually weakly accurate
Conventional failure detection

- A conventional failure detector will produce boolean result (trust vs suspect)
- The precision of the result depends on the inherent trade-offs:
  - Conservative failure detection: reducing the risk of wrongly suspecting (low mistake rates), high detection time.
  - Aggressive failure detection: low detection time, high mistake rate.
Conventional failure detection

- Trivial heart-beat failure detector:
  - Sending heart-beat message every $\Delta I$
  - Process q suspect process p if it fails to receive any messages from p for a period $\Delta t_\text{to} \geq \Delta I$
  - The choice of $\Delta t_\text{to}$ will affect the trade-offs between mistake rates and detection time.
  - $\Delta t_\text{to}$ should be chosen based on $\Delta I$ and the average network transmission delay of the network $\Delta tr$
Conventional failure detection

- Adaptive failure detector: considering the network changing condition.
  - Chen-FD:
    - Sampled arrival time and compute estimation of the arrival time of next heartbeat.
    - $\Delta t$ is computed based on the estimation plus a safety margin $\alpha$
    - Has 2 versions: 1 for synchronized clock, 1 for unsynchronized clock with negligible drift (Version 2 is used in the experiment)
  - Bertier-FD:
    - Same approach as Chen
    - Combine Chen’s estimation with a dynamic estimation based on Jacobson’s estimation of the round-trip time.
Accrual failure detector

- A failure detector service outputs a value on a continuous scale.
- If the process actually crashes, the value is guaranteed to accrue over time and tend toward infinity.
Accrual failure detector properties

- Suppose two processes p and q, q monitors p
- The output of the failure detector over time is not negative: $\text{susp\_level}_p(t) \geq 0$
- Asymptotic completeness: If process p is faulty, $\text{susp\_level}_p(t) \rightarrow \text{inf}$ as time $\rightarrow \text{inf}$
- Eventual monotony: If process p is faulty, there is a time after which $\text{susp\_level}_p(t)$ monotonic increasing.
- Upper bound: Process p is correct if and only if $\text{susp\_level}_p(t)$ has an upper bound over an infinite execution
- Reset: If process p is correct, then for any time $t_0$ , $\text{susp\_level}_p(t) = 0$ for some time $t \geq t_0$
A accrual failure detector can be transformed into a class of $\Diamond P$ following this algorithm:

- Process $q$ maintains two dynamic thresholds $\text{Thigh}$ and $\text{Tlow}$ initialized to the same arbitrary value greater than 0.
- S-transition: whenever $\text{susp\_level\_p}$ crosses $\text{Thigh}$, $q$ updates the value of $\text{Thigh}$ to $\text{Thigh} + 1$ and suspect $p$.
- T-transition: whenever $\text{susp\_level\_p}$ fall below $\text{Tlow}$, $q$ updates the value of $\text{Tlow}$ to that of $\text{Thigh}$ and stops suspecting $p$. 
Implementation of $\phi$ failure detector

- Straight forward flows
  - Sampling heartbeat arrivals
  - Calculating estimation distribution of heartbeat arrival based on the samples (Assuming its follow normal distribution)
  - Calculating $\phi$ value

![Diagram of failure detector process]

- App. 1: $\phi > \Phi_1 \Rightarrow \text{suspect}$
- App. 2: $\phi > \Phi_2 \Rightarrow \text{suspect}$
- App. 3: do $\text{action}(\phi)$
The meaning of \( \varphi \) value

- The \( \varphi \) is calculating from the probability that a heartbeat will arrives more than after the previous one

\[
\varphi(t_{\text{now}}) \overset{\text{def}}{=} -\log_{10}(P_{\text{later}}(t_{\text{now}} - T_{\text{last}}))
\]

- Hence, when \( \varphi \geq \Phi=1 \), the likeliness is 10%. When \( \Phi=2 \), the likeliness is 1% statistically.
Implementation of $\phi$ failure detector

- Calculating $\phi$ value
  - Sampling heartbeat intervals:
    - The monitoring process stores heart-beat arrival times into a sampling window of fixed size $WS$.
    - Determine the mean $\mu$ and the variance $\sigma^2$
  - Estimating the distribution and computing $\phi$

\[
P_{\text{later}}(t) = \frac{1}{\sigma \sqrt{2\pi}} \int_{t}^{+\infty} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \, dx
\]

\[
= 1 - F(t)
\]

\[
\varphi(t_{\text{now}}) \overset{\text{def}}{=} -\log_{10}(P_{\text{later}}(t_{\text{now}} - T_{\text{last}}))
\]
Experiment setup

- **Experiment goal:**
  - Study the effect of several parameters to behavior of $\varphi$ failure detector.
  - Compare the results of $\varphi$ with Chen and Bertier.

- **Environment setup**
  - Computer $p$ (monitored; Switzerland): Pentium III processor at 766 MHz and 128 MB of memory. Running Red Hat Linux 7.2
  - Computer $q$ (monitoring; Japan): Pentium III processor at 1GHz and 512 MB of memory. Running Red Hat Linux 9
  - All messages transmitted using UDP/IP
Experiment setup

- Heart-beat sampling:
  - Heart-beat messages were generated every 100ms
  - In total, 5,845,712 heartbeat messages were sent among which only 5,822,521 were received (about 0.4% of message loss).
  - The round-trip time measure is average of 283.3ms with a standard deviation of 27.3ms, a minimum of 270.2ms, and a maximum of 717.8ms
Experiment setup

- The heart-beat messages are then replayed for each different failure detector implementation to compare the result.
Experiment result

- Experiment 1: average mistake rate
  - A clear improvement shown when increases $\Phi$ from 0.5 to 2
  - The second significant improvement come when $\Phi$ from 8 to 12. As the lost of burst 1 is not generated suspicious.
Experiment result

- Experiment 2: average detection time
  - The results shown a sharp increase in avg detection time when the threshold value beyonds 10 or 11
Experiment result

- Experiment 3: Effect of window size
  - The experiment result confirm that the mistake rate of $\phi$ failure detector improve while increasing windows size.
  - The curves seem flatten for large value implies that increasing it further yields only little result
Experiment result

- Experiment 4: Comparision with Chen-FD and Bertier-FD
  - The behaviour of 3 detectors was measured with different tuning parameters (Φ for φ and α for Chen) and plots to compare.
  - The result shown that φ-FD does not incur any significant performance cost
Experiment result

- Experiment 4: Comparision with Chen-FD and Bertier-FD
  - When compare with Chen’s: $\phi$-FD does a little better in the aggressive range meanwhile Chen-FD does a little better in conservative range.
  - Bertier-FD did not perform very well as it is designed for LANs.
Summary

- **Paper main goal**
  - Introducing the $\phi$-FD implementation, an accrual failure detector.
  - Study influence of different parameters to its behaviour and compares with others FDs.

- **To achieve this goal**
  - Clearly describe the algorithm for $\phi$-FD
  - Do experiment to sample heart-beat messages and use the result to compare the implementation of different FDs.
Interesting points

- A concise background in failure detection in distributed system
- Clear description of $\phi$ FD algorithm
- Intuitive meaning of $\phi$ value
- Recorded heart-beat sample in a real experiment for result comparison.
Questions and discussion

- The computation cost of $\varphi$ and the distribution is should be investigating (For different windows size)
The END