Aula 23 - Detectores de falhas não confiáveis

Wednesday, June 8, 2016 13:44

Unreliable failure detectors

Podem ser usados na solução de problemas de acordo:

- Consenso;
- Atomic broadcast;
- Group membership;
- Etc.

Sistema assíncrono
$\Pi = \{p_1, p_2, \dots, p_n\}$
Processos falham por crash.
Canais de comunicação confiáveis.

Para o observador do sistema, há um relógio global ϕ (phi) que marca instantes discretos de tempo.

Se o processo p_i falha no instante de tempo t, então p_i não executa nenhuma ação após t.
Ou seja, as falhas são **permanentes**!

Estados dos processos: correto ou falho.

Uma execução do sistema é chamada de σ (sigma).
Definimos que:
$crashed(t, \sigma)$: conjunto de processos que falharam até o instante t.
$up(t, \sigma)$: conjunto de processos que não falharam na execução σ até o instante t (inclusive).

$$up(t, \sigma) = \Pi - crashed(t, \sigma)$$

Só consideramos execuções em que pelo menos 1 processo não falha:
$\forall t;\ up(t, \sigma) \neq \phi$.

Um detector de falhas D é um vetor $[D_1, D_2, \ldots, D_n]$.
Cada D_i consiste da lista de processos suspeitos por p_i.

- Cada processo p_i mantém um módulo de detecção de falhas D_i cuja saída é a lista de processos suspeitos de terem falhados.
- As suspeitas são baseados em timeouts.

Modelos para "usuários" dos detectores de falhas: assíncrono extendido com detector; os detectores de falha "encapsulam" o tempo.
$D_i(t, \sigma)$: conjunto de processos suspeitos por p_i na execução σ até o instante de tempo t (inclusive).

Se p_i falhou, então seu detector D_i é ignorado se: $p_i \in crashed(t, \sigma)$
Então $D_i(t, \sigma) = \phi$

Um detector de falhas pode cometer erros (por isso é dito "*unreliable*").
Em particular, um processo sem-falha pode ser erroneamente colocado na lista de suspeitos e depois retirado.

E o contrário? Um processo ser erroneamente considerado sem-falha.
Pode por um breve momento de tempo.

Processos distintos podem suspeitar de listas de processos diferentes.

Propriedades

Completude

O detector suspeita de todos os processos que falharam.

- 2 classes

Precisão

O detector **não** suspeita de processos corretos.

- 4 classes

Total: 8 classes.

Define a redutibilidade de detectores de falhas.
D é redutível em D': $D \rightarrow D'$ se existe um problema tal que se este problema pode ser resolvido com o detector D, então pode também ser resolvido com o detector D'.

Se $D \rightarrow D'$ e $D' \rightarrow D$, então D e D' são equivalentes.
Definição formal da completude:

Definição formal da completude:

Completude forte

Eventually, every process that crashes is permanently suspected by every correct process.

$\forall \sigma, \forall p \in crashed(\sigma), \forall q \in up(\sigma), \exists t \mid \forall t' \geq t\colon\ p \in D_q(t', \sigma)$

Completude fraca

Eventually, every process that crashes is permanently suspected by some correct process.

$\forall \sigma, \forall p \in crashed(\sigma), \exists q \in up(\sigma), \exists t \mid \forall t' \geq t\colon\ p \in D_q(t', \sigma)$

- Reduz completude fraca em forte.
- Apresenta um algoritmo: processo que suspeita de outro informa a todos.

É trivial obedecer à completude forte:
$\forall p, q \in \Pi, p \neq q\colon\ q \in D_p$.
O detector suspeita de todos os processos.
Assim, é fundamental definir a precisão.
Definição formal da precisão:

Precisão forte

A correct process is never suspected by any correct process.
$\forall \sigma, \forall t, \forall p, q \in up(t, \sigma)\colon\ p \notin D_q(t', \sigma)$.

Precisão fraca

Some correct process is never suspected by any correct process.
$\forall \sigma, \exists p \in up(\sigma), \forall t, \forall q \in up(t, \sigma)\colon\ p \notin D_q(t', \sigma)$.

Estas propriedades são chamadas de perpétuas: devem ser satisfeitas todo o tempo.
Foram "enfraquecidas" para permitir que sejam satisfeitas "*eventually*".

Eventual strong accuracy

Existe um instante de tempo após o qual processos corretos não são suspeitados por outros processos corretos.
$\forall \sigma, \forall p, q \in up(t', \sigma), \exists t | \forall t' \geq t\colon\ p \notin D_q(t', \sigma)$.

Eventual weak accuracy

Existe um instante de tempo após o qual algum (pelo menos um) processo correto não é suspeitado por outros processos corretos.

$\forall\sigma, \exists p \in up(t',\sigma), \forall q \in up(t',\sigma), \exists t | \forall t' \geq t\colon\ p \notin D_q(t',\sigma).$

Classes

1) Perfeito(P): completude forte e precisão forte
2) *Eventually perfect* ($\diamond P$): completude forte e precisão *eventually* forte
3) Forte (S): completude forte e precisão fraca
4) *Eventually strong* ($\diamond S$): completude forte e precisão *eventually* fraca
5) Fraco (W): completude fraca e precisão fraca
6) *Eventually weak* ($\diamond W$): completude fraca, precisão *eventually* fraca

Nunca se assume precisão mais forte que completude.

Com a classe $\diamond W$, já é possível resolver o problema de consenso.

Algoritmo

/* executado pelo processo $\mathrm{p} \in \Pi$ */

<u>Inicialização</u>

$\mathrm{D_p} = \Pi - \{\mathrm{p}\}$

$\forall \mathrm{q} \in \Pi - \{\mathrm{p}\}$: $\mathrm{timeout_q} = (1 - \mathrm{p})\Delta\mathrm{heartbeat_q}$

Envie $\mathrm{heartbeat_p}$ $\forall \mathrm{q} \in \Pi - \{\mathrm{p}\}$

<u>Tarefa 1</u>

A cada intervalo $\Delta heartbeat_p$:

Envie $heartbeat_p$ $\forall q \in \Pi - p$

<u>Tarefa 2</u>

Ao receber $heartbeat_q$:

$D_p = D_p - q$

$timeout_q = (1 - q)\Delta heartbeat_q$

<u>Tarefa 3</u>

$timeout_q$ expira!

$D_p = D_p \cup \{q\}$