Aula 22 - Problema do Consenso/Acordo

Monday, June 6, 2016 13:45

Esta semana:
- Detectores de Falha
- Problema do Consenso/Acordo

Próxima semana:
- Algoritmo Paxos

Depois: Prova 2 dia 20/06

O consenso

Considere um sistema Π (pi) de n processos.
$\Pi = \{p_1, p_2, \dots, p_n\}$.
Modelo temporal: Assíncrono
Modelo de falhas: *crash*

No consenso, consideramos que os processos propõem valores iniciais.
- Estes valores são de um conjunto de valores possíveis. Por exemplo, $\{0, 1\}$.

Após a execução do consenso, todos os processos entram em acordo sobre um determinado valor. Os processos "decidem" sobre um único valor.

Formalmente, o consenso é definido em termos de três propriedades:
1. Acordo (*agreement*). Todos os processos corretos decidem pelo mesmo valor.
2. Validade: Se um processo correto decide por um valor, então esse valor foi proposto por um processo correto.
3. Terminação: Todo processo *eventually* (fatalmente) decide por um valor.

O consenso tem várias aplicações:
1. Replicação distribuída;
2. Outros problemas são redutíveis ao consenso; por exemplo:
    a. Broadcast atômico;
    b. 2PC, 3PC: *2-phase commit*, *3-phase commit* (NBAC - *non-blocking atomic commitment*);
    c. *Group membership*.

Em 1985, Fischer, Lynch & Patterson (FLP) provaram que, em um sistema

distribuído assíncrono sujeito a falhas *crash* de processos, o consenso é impossível**.** **Impossibilidade FLP**.

Uma forma de "questionar" as implicações práticas da impossibilidde FLP é examinar o modelo.
Em 1988, Dwork & Lynch investigaram modelos parcialmente síncronos.

- A impossibilidade FLP foi **decisiva** na comunidade de teoria de sistemas distribuídos.
- Muitos resultados para sistemas assíncronos
- Implicação prática
- O problema básico é distinguir processos falhos de processos lentos.

Em 1996, Candra & Toueg propuseram uma nova forma de "enxergar" o problema do consenso. Propuseram os detectores de falhas não-confiáveis ("*Unreliable Failure Detectors*"). Cada processo que executa o consenso tem acesso a um módulo do detector de falhas. O detector de falhas, quando consultado, retorna lista de processos **suspeitos** de terem falhados.

Tanto a falha como o estado "sem-falha" (correto) podem não refletir a realidade! Daí: ***unreliable***.

Os detectores de falha são chamados também de "oráculos".

- Informam uma lista "tentativa" de processos suspeitos de terem falhado.

A grande pergunta é:

- Quais das propriedades que o detector de falhas deve ter para **permitir** o consenso em sistemas assíncronos sujeitos a falhas *crash*?

Outra pergunta (talvez mais importante!) é:

- Qual é o "mínimo" necessário (em termos de propriedades dos detectores de falhas) que permita o consenso?

Chandra & Toueg definiram duas propriedades:

1. **Completeness** (completude): o detector de falhas suspeita de todos os processos que falharam.
    a. Falhas *crash* são permanentes.
2. **Accuracy** (precisão): o detector não suspeita de processos corretos.
    a. Inclusive "lentos".

Se um nodo tiver um diagnóstico errado, em mais algum tempo ele detecta a falha (crash permanente).

As duas propriedades são classificadas em fraca (*weak*)/forte(*strong*):

- **Strong completeness**: *Eventually*, todo processo que falha é suspeito por todo processo correto.
- **Weak completeness**: *Eventually*, todo processo que falha é suspeitado por pelo menos 1 processo correto.
- **Strong accuracy**: Nenhum processo é suspeito sem ter falhado
- **Weak accuracy**: Pelo menos um processo correto nunca é suspeitado
- **Eventual strong accuracy**: Existe um tempo após o qual nenhum processo correto é suspeitado.
- **Eventual weak accuracy**: Existe um tempo após o qual pelo menos 1 processo correto não é suspeitado.