Tarea 5 - Cómputo distribuido

Dalia Camacho, Gabriela Vargas

1 Dibujar dos rondas del complejo $\mathbf{S}_1$

Consideramos el complejo $S_1 = \{A \leftrightarrow B, A \leftarrow B\}^w \cup \{A \leftrightarrow B, A \rightarrow B\}^w$ en el que la falla de comunicación solo puede darse en uno de los canales, alternando con comunicación perfecta.

En la primera ronda los procesos únicamente conocen su preferencia.

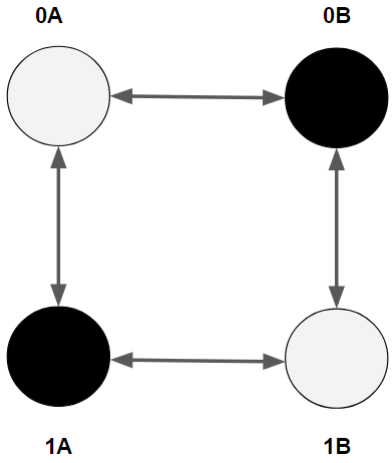


Figure 1: Gráfica que representa la primera ronda del complejo S_1.

En la segunda ronda, se consideran los escenarios de fallo en alguno de los canales. Una vez que falla un canal, no puede presentarse falla en el otro. También se considera el escenario en que los nodos mantienen comunicación perfecta.

En las etiquetas de estado de cada vértice el primer término significa la preferencia del proceso y el segundo término indica la decisión tomada de acuerdo al algoritmo explicado en la sección 2.

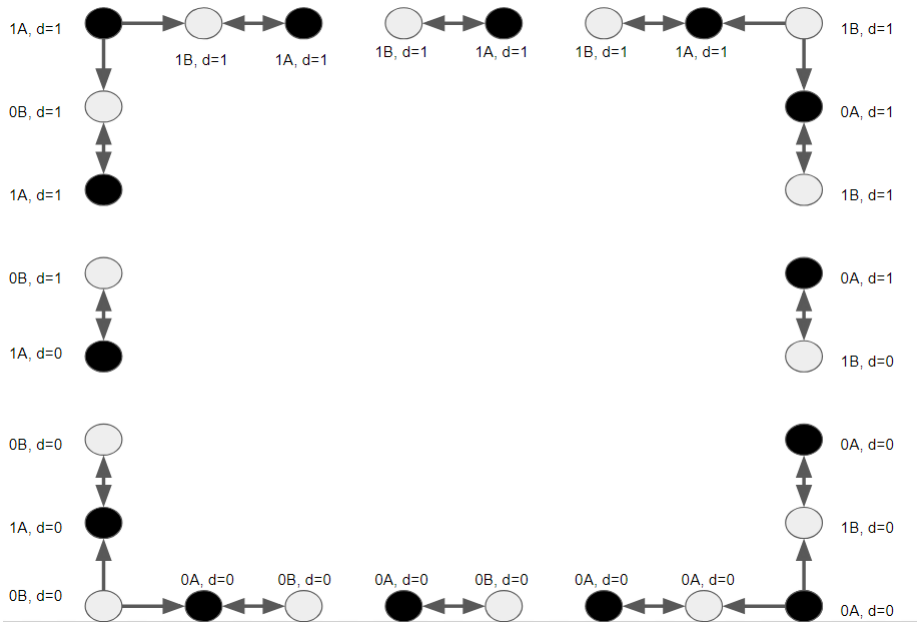


Figure 2: Gráfica que representa la segunda ronda del complejo S_1.

En la siguiente gráfica se observa que para los escenarios que presentan falla en comunicación, se cumplen las condiciones de *agreement*, *validity* y *termination* y, por lo tanto, existe solución al problema de consenso. Esto también se cumple en el caso de comunicación perfecta cuando ambos procesos comparten preferencia (regiones marcadas en verde). Sin embargo, en los casos de comunicación perfecta, donde los procesos no tienen la misma preferencia, no se cumple la condición de *agreement* y el problema de consenso no tiene solución (región marcada en gris).

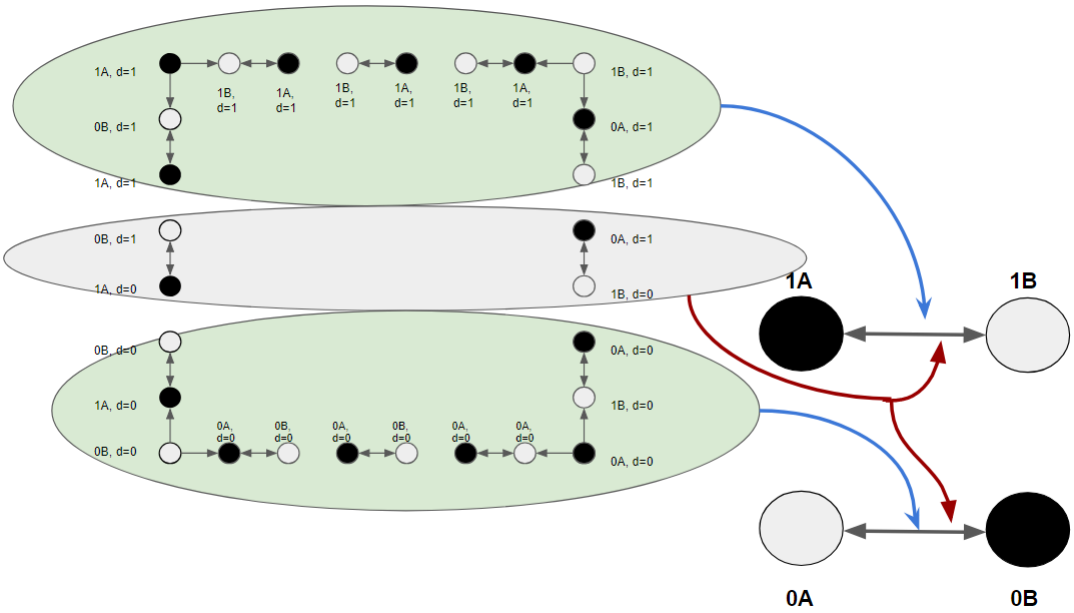


Figure 3: Gráfica que representa las primeras dos rondas del complejo S_1.

1.1 Algoritmo de Consenso para el complejo S_1

1. El proceso P_i envía el mensaje X_i a P_j en el tiempo 0 y en el tiempo 1.
2. **Decisión**
   - Si P_i no recibe ningún mensaje en R_2 entonces decide por el valor $d_i = X_i$.
   - Si P_i recibe el mensaje de P_j en una ocasión decide por el valor X_i.
   - Si P_i recibe el el mensaje X_j de P_j en dos ocasiones, decide por el valor X_j

Si $X_i = X_j$ o bien uno de los canales falla en al menos una ocasión los procesos tomarán la misma decisión. En cambio si la comunicación es perfecta y $X_i \neq X_j$ entonces $d_i = X_j$ y $d_j = X_i$ con lo que $d_i \neq d_j$ y no se logra la propiedad de *agreement*. El algoritmo anterior corresponde a tomar una decisión en la segunda ronda. Si la comunicación tomará n rondas podemos tener el algoritmo siguiente:

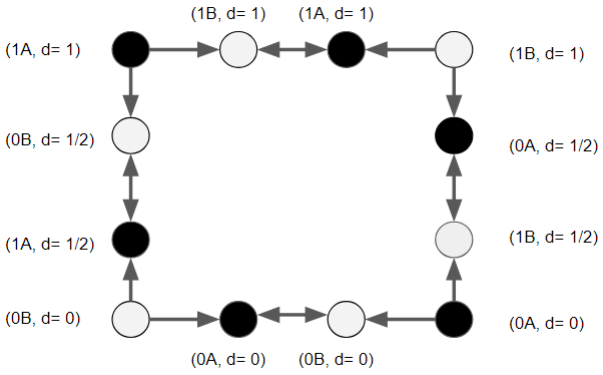
1. El proceso P_i envía el mensaje X_i a P_j los tiempos $\{0, 1, 2, \ldots n-1\}$.
2. *Decisión*
   - Si P_i no recibe ningún mensaje en R_n entonces decide por el valor $d_i = X_i$.

- Si P_i recibe el mensaje de P_j en al menos una ocasión y menos de n ocasiones X_i.
- Si P_i recibe el mensaje X_j de P_j en n ocasiones, decide por el valor X_j.

Al aumentar el número de rondas es menos probable que la comunicación se mantenga de forma perfecta, entonces al aumentar el número de rondas la probabilidad de llegar a un consenso aumenta.

2 Acuerdo de Agreement $\frac{1}{2^k}$

En el complejo de entradas se indican las preferencias de cada proceso, así como las decisiones tomadas según el acuerdo de agreement con $k = 1$, el cual indica que si dos procesos tienen preferencias incompatibles (0 y 1 respectivamente) ambos decidirán 1/2.



k.png

Figure 4: Complejo de entradas.

El complejo de salida contiene los pares de decisiones válidas compatibles tal que cumplan la condición $|DecA - DecB| <= 1/2$

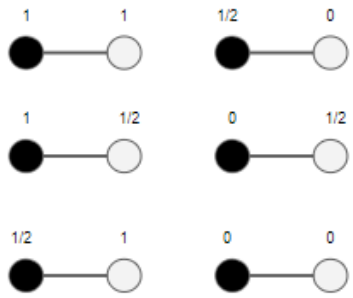


Figure 5: Complejo de Salida.

3 Resumen de la clase

La clase pasada revisamos el problema de consenso bajo algunos modelos de comunicación. El problema de consenso tiene las siguientes características:

- *Validity:* Si $d_i \neq \perp \Rightarrow d_i \in input$.
- Agreement: Si $d_i \neq \perp$ y $d_j \neq \perp \Rightarrow d_i = d_j$
- *Termination:* Eventualmente $d_i \neq \perp$ para todo i.

Los modelos de comunicación vistos en clase fueron

- $R_1 = \{A \leftrightarrow B, A \leftarrow B, A \rightarrow B\}^w$: Al menos un mensaje llega en cada ronda.
- $C_1 = \{A \leftrightarrow B\}^w \cup \{A \leftrightarrow B\}^* \cup \left(\{A \leftarrow B^w \ \cup A \rightarrow B^w\}\right)$: Uno de los canales muere i.e. *crash failure*.
- $S_1 = \{A \leftrightarrow B, A \leftarrow B\}^w \cup \{A \leftrightarrow B, A \rightarrow B\}^w$: El canal falla en a lo más un sentido, pero la comunicación puede recuperarse.

Estos esquemas de comunicación se relacionan como $C_1 \subset S_1 \subset R_1$.

Dada una ronda en que se debe tomar una decisión ambos procesos deciden por algún valor, por lo que *termination* siempre se cumple. Sin embargo es imposible resolver el problema de consenso para el caso general R_1.

Vimos también el acuerdo de agreement para $\frac{1}{2^k}, k \geq 0$.

- *Validez:* Si $d_i \neq \perp \Rightarrow d_i \in input$.
- *Acuerdo:* Se llega a un acuerdo si $|d_i - d_j| \leq \frac{1}{2^k}$ para un valor de k definido previamente.
- *Terminación* Ambos procesos toman una decisión $d_i, d_j \neq \perp$.

Vimos también el algoritmo de *flood set consensus protocol* enfocado a cuando ocurren fallas de tipo *crash failure* en un sistema síncrono, donde los procesos que presentan la falla detienen su ejecución de forma prematura.

En este algoritmo, cada proceso p_i invoca una función Consensus(v_i), donde v_i corresponde al valor propuesto por el proceso y f corresponde al número de fallas toleradas.

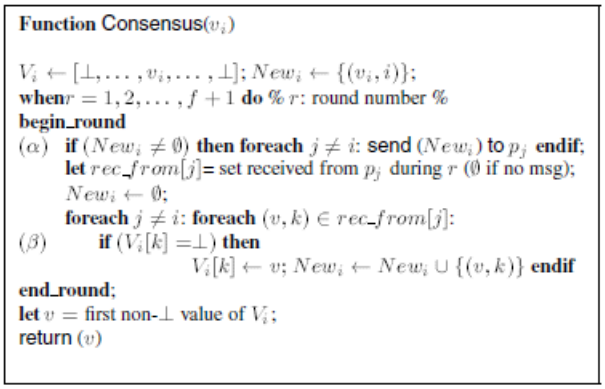


Figure 6: Flood set consensus protocol.

Este modelo tiene solución, pero el tiempo que tarda depende del número máximo de fallas.