Transactions (Partie II)

G. Raschia

dernière mise à jour : le 20 mars 2026

1 2PL

Soit le plan d'exécution P des trois transactions T_1, T_2 et T_3 :

$$P = r_1(a); r_2(c); r_2(b); w_1(a); r_3(a); r_1(b); w_2(c); w_2(b); c_2; r_3(b); r_3(c); c_3; w_1(b); c_1$$

1. Énumérer dans P les paires d'actions conflictuelles pouvant produire chacune des anomalies suivantes :
   (a) les « lectures sales » (*dirty reads*)
   (b) les « lectures uniques » (*unrepeatable reads*)
   (c) les « modifications perdues » (*lost updates*)
2. Construire le graphe de sérialisation $\mathcal{G}(P)$ du plan P.
3. P est-il sérialisable par conflit ? Justifier.
4. On s'intéresse au protocole de verrouillage à deux phases dans sa version SS2PL [1]. Les modes exclusif $x_i(.)$ et partagé $s_i(.)$ sont disponibles, ainsi que l'opération de libération $\ell_i(.)$ et la promotion de verrou s en verrou x, notée $x_i^*(.)$. P est-il SS2PL ?
5. Jouer l'histoire P selon SS2PL. Si une action ne peut accéder à une ressource, la transaction correspondante est mise en attente et l'action suivante du plan est considérée. Reporter les problèmes éventuels.

   Un verrou de mise-à-jour $u_i(.)$ est une forme spécifique de verrou partagé, qui, une fois posé, interdit toute pose de verrou supplémentaire, partagé ou exclusif. En revanche, il n'est pas impossible que la ressource soit verrouillée en lecture avant la pose du verrou de mise-à-jour. À quoi sert un verrou de mise-à-jour ? Il est utile en combinaison avec la promotion de verrou $x_i^*(.)$. En effet, chaque fois qu'une transaction lit un objet dans la perspective de l'écrire, le programmateur pose un verrou de mise-jour plutôt qu'un verrou partagé, réalise la lecture, puis il promeut ce verrou de mise-à-jour en verrou exclusif au moment de l'écriture.
6. Dessiner la matrice de compatibilité des verrous s, u et x, selon qu'une ressource est déjà verrouillée par s, u ou x.
7. Rejouer l'histoire P selon SS2PL, en considérant que l'on dispose maintenant de verrous de mise-à-jour.

1. La phase de libération est confondue avec le `commit/rollback` de la transaction.

2 Journalisation

Soient les 12 enregistrements d'un journal des images Avant/Après :

01	$\langle \text{START } T_1 \rangle$
02	$\langle T_1, a, 4, 5 \rangle$
03	$\langle \text{START } T_2 \rangle$
04	$\langle \text{COMMIT } T_1 \rangle$
05	$\langle T_2, b, 9, 10 \rangle$
06	$\langle \text{START CKPT } (T_2) \rangle$
07	$\langle \text{START } T_3 \rangle$
08	$\langle T_3, a, 5, 17 \rangle$
09	$\langle T_2, a, 17, 4 \rangle$
10	$\langle \text{END CKPT} \rangle$
11	$\langle \text{COMMIT } T_2 \rangle$
12	$\langle \text{COMMIT } T_3 \rangle$

Un point de reprise (`CKPT`) a une durée marquée par les instants de début (`START CKPT`) et de fin (`END CKPT`). Toutes les transactions actives au début du point de reprise sont indiquées dans l'enregistrement `START CKPT`. Au point de reprise, l'écriture des pages sales (opération *flush*) est garantie pour toutes les actions antérieures à l'enregistrement `START CKPT`. Cependant, tant que l'enregistrement de fin de point de reprise (`END CKPT`) n'est pas écrit dans le journal, le point de reprise en cours n'offre *aucune* garantie.

1. Décrire la procédure de reprise lorsqu'une panne survient :
   (a) après l'enregistrement 12 ;
   (b) après l'enregistrement 11 ;
   (c) après l'enregistrement 10 ;
   (d) après l'enregistrement 09.