TP 1 (2019) Systèmes concurrents et distribués

Exercice 2 :

On veut accéder des ressources numérotées 1,2,3,… dans un programme concurrent en exclusion mutuelle. Avant chaque accès on appelle la routine *hold(i)* avec *i* est le numéro de la ressource qu'on veut accéder. Cette routine est bloquante, c'est-à-dire que si la ressource n'est pas disponible le thread passe dans l'état *bloqué* et repasse dans l'état *exécutable* lorsqu'elle est libérée. Pour rendre une ressource accessible à un autre thread on utilise la routine *release(i)*.

Montrez que si un thread réserve toujours les ressources dans l'ordre croissant alors il n'y a jamais d'interblocage.

Preuve. On considère parmi l'ensemble des processus celui qui a accès à la ressource avec le numéro le plus grand. On note Max le numéro de cette ressource. Pour que ce processus soit interbloqué il faut qu'il satifasse la propriété *hold and wait*. Mais ce processus ne peut pas être en attente sur une ressource car si c'est le cas, alors cette ressource a un numéro plus grand que Max et est accédée par un autre processus. C'est impossible par la définition de Max. Ce processus va donc rendre les ressources acquises et le système va progresser, il n'y a donc pas interblocage.