On considère le problème de l'accès de n processus à une unique ressource non partageable. La ressource ne peut être accédée que par un processus à la fois : elle est dite *critique*, et son utilisation se fait au sein de ce que l'on appelle une *section critique*. On souhaite donc définir des *protocoles d'exclusion mutuelle*, garantissant cette propriété d'accès unique.

I Définitions

Processus. On considère un ensemble fini $\mathcal{P}$ de $n > 0$ processus : $\mathcal{P} = \{p_0, p_1, \ldots, p_{n-1}\}$ s'exécutant simultanément.

États. L'ensemble X_i des états possibles du processus p_i est partitionné en

$$X_i = R_i \cup E_i \cup C_i \cup S_i$$

Ces ensembles deux à deux disjoints seront appelés respectivement section *restante*, section *d'essai*, section *critique* et section *de sortie*.

Variables. $\mathcal{V}$ est un ensemble fini de $m > 0$ variables, partagées entre les processus. La j^{e} variable de $\mathcal{V}$ a ses valeurs dans l'ensemble V_j.

Configurations. Une *configuration* est un $(n+m)$-uplet

$$q = (x_0, \ldots, x_{n-1}, v_0, \ldots, v_{m-1})$$

où $x_i \in X_i$ pour $0 \leq i < n$ et $v_j \in V_j$ pour $0 \leq j < m$. Une configuration contient donc les états respectifs de tous les processus et les valeurs de toutes les variables à un instant donné. On notera X_i, V_j et V les projections définies par

$$X_i(q) = x_i, \quad V_j(q) = v_j \quad \text{et} \quad V(q) = (v_0, \ldots, v_{m-1})$$

Système. Un *système* est un quadruplet $S = (\mathcal{P}, \mathcal{V}, q_0, \varphi)$ où
- $\mathcal{P}$ est l'ensemble des processus,
- $\mathcal{V}$ est l'ensemble des variables,
- q_0 est la configuration initiale de S,
- φ est la fonction de transition de S, définie ci-dessous.

Transitions. Soit Q l'ensemble de toutes les configurations de S. Alors la fonction de transition φ est une fonction totale de $\{0, \ldots, n-1\} \times Q$ dans Q telle que, si $i \in \{0, \ldots, n-1\}$ et $q \in Q$, alors $q' = \varphi(i, q)$ vérifie

1. $X_j(q') = X_j(q)$ pour $j \in \{0, \ldots, n-1\} \backslash \{i\}$, et,
2. si $X_i(q) \in R_i \cup E_i$ alors $X_i(q') \in E_i \cup C_i$, et,
3. si $X_i(q) \in C_i \cup S_i$ alors $X_i(q') \in S_i \cup R_i$.

On notera $q \xrightarrow{i} q'$ une telle transition, et l'on parlera d'une transition du processus p_i.

Lectures/écritures. On restreint la fonction de transition avec les conditions suivantes : pour tout $i \in \{0, \ldots, n-1\}$, X_i peut être partitionné en $2m$ ensembles disjoints $Read_i^k$ et $Write_i^k$ pour $k \in \{0, \ldots, m-1\}$ tels que, pour toute configuration q :

- Si $X_i(q) \in Read_i^k$, on dit que p_i est *prêt à lire la variable* k, c'est-à-dire que la prochaine transition *du processus* p_i sera une lecture de la variable k. La fonction de transition φ doit donc vérifier les conditions suivantes :
  1. $V(\varphi(i, q)) = V(q)$, et,
  2. pour toute configuration q' avec $X_i(q') = X_i(q)$, si $V_k(q') = V_k(q)$ alors $X_i(\varphi(i, q')) = X_i(\varphi(i, q))$.

  On dit alors que la transition $q \xrightarrow{i} \varphi(i, q)$ est une *lecture* de la k^{e}variable par p_i.

- Si $X_i(q) \in Write_i^k$, on dit que p_i est *prêt à écrire la variable* k, c'est-à-dire que la prochaine transition *du processus* p_i sera une lecture de la variable k. La fonction de transition φ doit donc vérifier les conditions suivantes :
  1. $V_j(\varphi(i, q)) = V_j(q)$ pour $j \neq k$, et,
  2. pour toute configuration q' avec $X_i(q') = X_i(q)$, on a $V_k(\varphi(i, q')) = V_k(\varphi(i, q))$ et $X_i(\varphi(i, q')) = X_i(\varphi(i, q))$.

  On dit alors que la transition $q \xrightarrow{i} \varphi(i, q)$ est une *écriture* de la k^{e} variable par p_i.

Histoire, calcul, atteignabilité, résultat. Une *histoire* de S est une suite (finie ou infinie) d'indices de processus (éléments de $\{0, \ldots, n-1\}$). Une occurrence de l'indice i dans une histoire est appelée *pas* du processus p_i dans cette histoire.

Si q_1 est une configuration et $h = i_1 i_2 \ldots$ une histoire, alors $q_1 q_2 \ldots$ est un *calcul* par h à partir de q_1, où $q_{j+1} = \varphi(i_j, q_j)$ pour $j = 1, 2, \ldots$. Si h est finie, alors $res(q_1, h)$ est la configuration finale dans le calcul par h à partir de q_1. On dit que la configuration q' est *atteignable* à partir de la configuration q s'il existe une histoire finie h telle que $q' = res(q, h)$. Une configuration est dite *atteignable* si elle est atteignable à partir de q_0.

Un point important est de voir que, puisqu'un calcul est une suite de transitions opérées chacune par un seul processus, un processus peut être prêt à lire (ou écrire) une variable pendant plusieurs transitions au cours d'un calcul avant que cette lecture (ou écriture) ne devienne effective.

Équité, arrêt. Une histoire h est *équitable* à partir de la configuration q si pour tout préfixe fini h_1 de h et pour tout $i \in \{0, \ldots, n-1\}$, $X_i(res(q, h_1)) \notin R_i$ implique que i apparaît dans h_2, où h_2 est défini par $h = h_1 h_2$.

Un processus *s'arrête* dans une histoire si et seulement si il y apparaît un nombre fini de fois. La notion d'arrêt est donc reliée à celle d'histoire, plutôt qu'à l'état d'un processus. Si une histoire h est équitable à partir de q, tout processus qui s'arrête dans h est dans sa section restante après son dernier pas dans le calcul par h à partir de q.

Exclusion mutuelle. Une configuration q *ne satisfait pas la propriété d'exclusion mutuelle* s'il existe deux valeurs distinctes i et j de $\{0, \ldots, n-1\}$ telles que $X_i(q) \in C_i$ et $X_j(q) \in C_i$. Le système S *satisfait la propriété d'exclusion mutuelle* si toute configuration atteignable q de S satisfait la propriété d'exclusion mutuelle.

Progrès global. Le processus p_i *change de section* par h en partant de q s'il existe des préfixes finis h_1 et h_2 de h tels que $X_i(res(q, h_1))$ est dans une section différente de celle de $X_i(res(q, h_2))$. Un système S a la propriété de *progrès global* si pour toute configuration atteignable q et toute histoire non nulle h équitable à partir de q, un processus au moins change de section par h en partant de q. On notera qu'un système ayant la propriété de progrès global peut laisser un processus en situation de famine, c'est-à-dire dans une situation où il tente d'entrer en section critique mais n'y parvient jamais.

II Une borne supérieure

L'algorithme de la figure 1 est exprimé dans un langage impératif de type Pascal. Il opère sur un tableau partagé *drapeau*[] de taille n, à valeurs dans $\{haut, bas\}$. À l'initialisation du système, toutes les valeurs de *drapeau*[] valent *bas*, et chaque processus p_i exécute le code du programme proc_i. En plus des sections restante, d'essai, critique et de sortie, on définit une sous-section d'essai et une section de passage à l'intérieur de la section d'essai, comme indiqué en commentaire dans l'algorithme. La section de passage correspond donc à la fin de la section d'essai.

On veut montrer que le système défini par cet algorithme vérifie les propriétés de progrès global et d'exclusion mutuelle.

```
program proc_i
    local var j : 0..n-1
begin
    while true do begin
        section restante                                   (* section restante *)
        repeat                                             (* début de la section d'essai *)
            drapeau[i] ← bas
            repeat until ∀j ∈ {0,...,i−1} : drapeau[j] = bas   (*sous-section d'essai*)
            drapeau[i] ← haut
        until ∀j ∈ {0,...,i−1} : drapeau[j] = bas
        repeat until ∀j ∈ {i+1,...,n−1} : drapeau[j] = bas    (*section de passage*)
        section critique                                   (* section critique *)
        drapeau[i] ← bas                                   (* section de sortie *)
    end
end.
```

FIG. 1 – Un algorithme d'exclusion mutuelle

Question 1. *Soit q une configuration telle que au moins un processus n'est pas dans sa section restante, et h une histoire équitable telle que aucun processus ne change de section par h en partant de q. Montrer que pour $i \in \{0,\ldots,n-1\}$, soit $X_i(q) \in R_i$ et i n'apparaît pas dans h, soit $X_i(q) \in E_i$ et i apparaît infiniment souvent dans h.*

Réponse. Soit $i \in \{0,\ldots,n-1\}$. Si $X_i(q) \in R_i$, comme i ne change pas de section, i n'apparaît pas dans h. D'autre part, si $X_i(q) \notin R_i$, comme h est équitable et que i ne change pas de section, i apparaît infiniment souvent dans h. Comme de plus les seules boucles dans le protocole sont dans la section d'essai, on a alors $X_i(q) \in E_i$.

□

On dit qu'un processus qui n'est pas dans R_i est *actif*.

Question 2. *Avec les mêmes hypothèses, montrer qu'il existe un préfixe de h tel que dans la configuration q' atteinte par ce préfixe en partant de q, tout processus actif est soit dans sa section de passage, ou dans un état à partir duquel il bouclera infiniment dans la sous-section d'essai.*

Réponse. Si p_i atteint la section de passage, il y restera pour le reste de l'histoire, et son drapeau vaudra toujours *haut*. Soit $k = \min\{i \in \{0,\ldots,n-1\} \mid p_i \text{ est actif en } q\}$. Comme p_k détecte que $\forall j \in \{0,\ldots,k-1\}, drapeau[j] = bas$, p_k atteint la section de passage après un préfixe fini de h. Après une extension appropriée de ce préfixe, chaque p_i actif sera soit dans sa section de passage, soit en train de boucler indéfiniment dans sa sous-section d'essai, avec $drapeau[i] = bas$, puisque tous les processus qui n'atteignent pas la section de passage auront détecté que $drapeau[k] = haut$. On appelle alors q' la configuration correspondant à ce préfixe.

$\square$

Question 3. *Toujours avec les mêmes hypothèses, montrer qu'il existe un processus qui, en partant de la configuration q', va changer de section par h.*

Réponse. Soit $l = \max\{i \in \{0, \ldots, n-1\} \mid X_i(q') \text{ est dans la section de passage }\}$. Le processus p_l détecte que $drapeau[i] = bas$ pour tout $i \in \{l+1, \ldots, n-1\}$, donc p_l va changer de section par h.

$\square$

Question 4. *Conclure sur la propriété de progrès global.*

Réponse. Les hypothèses de la question 1 correspondent à la négation de la propriété de progrès global, et la question 3 mène à une contradiction, d'où le résultat par l'absurde.

$\square$

Question 5. *Montrer que l'algorithme vérifie la propriété d'exclusion mutuelle.*

Réponse. Supposons que l'exclusion mutuelle puisse être violée. Alors il existe i et j dans $\{0, \ldots, n-1\}$ tels que $i \neq j$, et une histoire finie h, avec $q = res(q_0, h), X_i(q) \in C_i$ et $X_j(q) \in C_j$. Il existe une configuration où p_i met $drapeau[i]$ à $haut$ pour la dernière fois avant d'entrer en section critique. Soit q_a cette configuration pour p_i, et q_b la configuration similaire pour p_j, dans le calcul $q_0, q_1, \ldots, q_k$ par h à partir de q_0 ($q = q_k$). On peut supposer sans perte de généralité que $a < b$. Mais alors, pour tout c tel que $a \leq c \leq b$, on a $drapeau[i] = haut$ en q_c. Comme p_j doit tester $drapeau[i]$ après q_b (disons, en q_d), soit il répète un nouveau cycle dans sa section d'entrée, ou alors il boucle dans sa section de passage pendant $q_b, q_{b+1}, \ldots, q_k$, ce qui contredit notre hypothèse. L'exclusion mutuelle est donc bien assurée.

$\square$

III Une borne inférieure

On veut maintenant montrer le résultat suivant : tout système avec $n \geq 2$ processus vérifiant les propriétés d'exclusion mutuelle et de progrès global doit avoir au moins n variables partagées. Dans toute la suite, on désignera par S un système qui satisfait par hypothèse ces deux propriétés.

On commence par donner la définition suivante.

Configurations équivalentes. Soient q et q' deux configurations, et p_i un processus. On dit que q et q' sont équivalentes pour p_i, que l'on note $q \underset{i}{\sim} q'$, si $V(q) = V(q')$ et $X_i(q) = X_i(q')$.

Question 6. *Soient q et q' deux configurations, $\mathcal{P}'$ un sous-ensemble de $\mathcal{P}$, et h une histoire finie de S qui ne contient que des pas de processus de $\mathcal{P}'$. Montrer que si $q \underset{i}{\sim} q'$ pour tout $p_i \in \mathcal{P}'$, alors $V(res(q,h)) = V(res(q',h))$ et pour tout $p_j \in \mathcal{P}', X_j(res(q,h)) = X_j(res(q',h))$.*

Réponse. Par récurrence sur la longueur n de h.

Pour $n = 0$, le résultat est immédiat.

Pour $n = 1$: h est constituée d'un pas d'un processus $p_i \in \mathcal{P}'$. Si le pas i est une lecture, alors d'après la condition 1 sur les lectures, $V(res(q,h)) = V(res(q',h))$, et d'après la condition 2 sur les lectures, $X_i(res(q,h)) = X_i(res(q',h))$. D'autre part, pour tout $j \in \mathcal{P}' \backslash \{i\}$, $X_j(res(q,h)) = X_j(res(q',h))$, puisque j n'a pas évolué. Si le pas i est une écriture de la variable k, $V_j(res(q,h)) = V_j(res(q',h))$ pour $j \neq k$ (condition 1 sur les écritures), et $V_k(res(q,h)) = V_k(res(q',h))$ (condition 2 sur les écritures). De plus, toujours par cette même condition, $X_j(res(q,h)) = X_j(res(q',h))$ pour tout $j \in \mathcal{P}'$. On a donc bien $res(q,h) \underset{i}{\sim} res(q',h)$ pour tout $i \in \mathcal{P}'$.

Supposons maintenant la propriété vérifiée pour n, et considérons h de longueur $n+1$: $h = h'i$. Par hypothèse de récurrence, $res(q,h') \underset{i}{\sim} res(q',h')$ pour tout $i \in \mathcal{P}'$. On applique le résultat du cas $n = 1$, et on conclut immédiatement que $res(q,h) \underset{i}{\sim} res(q',h)$ pour tout $i \in \mathcal{P}'$.

$\square$

Écriture oblitérée. Soient q_1 une configuration, h une histoire, $q_1 q_2 \ldots$ étant le calcul de h en partant de q_1. Soit $i \in \{0, \ldots, n-1\}, v \in \mathcal{V}$, et $j > 0$ tel que $q_j \xrightarrow{i} q_{j+1}$ est une écriture de v par p_i. S'il existe $k > j$ tel que $q_k \rightarrow q_{k+1}$ est une écriture de v, et pour tout l tel que $j < l < k$, $q_k \rightarrow q_{k+1}$ n'est *pas* une lecture de v par un processus autre que p_i, on dit que l'écriture de v par p_i en q_j est *oblitérée* par h en partant de q_1.

Processus caché. Soit q une configuration, h une histoire, et $i \in \{0, \ldots, n-1\}$. On dit que p_i est *caché* de q par h s'il existe des histoires h_1 et h_2, h_1 finie, telles que $h = h_1 h_2$, $X_i(res(q,h_1)) \in R_i$, et chaque écriture de p_i dans le calcul de h_2 en partant de $res(q,h_1)$ est oblitérée par h_2 en partant de $res(q,h_1)$.

Soit q une configuration, h une histoire finie, et $\mathcal{P}'$ un sous-ensemble de $\mathcal{P}$ tel que chaque processus de $\mathcal{P}'$ est caché par h en partant de q. Soit $q' = res(q,h)$. Pour chaque $p \in \mathcal{P}'$, soit h_p le plus long préfixe de h tel que p est dans sa section restante à $res(q,h_p)$, et h_1 défini par $h = h_p h_1$. Soit k_p le nombre de pas de p dans le suffixe de h après h_p. On appelle ces pas des *pas cachés*.

Question 7. *S'il existe $p \in \mathcal{P}'$ tel que $k_p > 0$, soit p_i le processus de $\mathcal{P}'$ qui exécute le dernier pas caché de h. Soit h' l'histoire obtenue en supprimant de h le dernier pas de p_i. Montrer que tous les processus de $\mathcal{P}'$ sont cachés par h'.*

Réponse. Si le dernier pas de p_i est une lecture, alors tous les processus de $\mathcal{P}'$ sont encore cachés par h', puisque aucune écriture après cette lecture n'est affectée.

Si le dernier pas de p_i est une écriture, et s'il existe un processus de $\mathcal{P}'$ qui n'est pas caché par h', alors il y a une écriture w_j dans une variable v par un certain $p_j \in \mathcal{P}'$ telle que w_j n'est pas oblitérée dans le calcul par h' à partir de q. Comme par ailleurs w_j est oblitérée dans le calcul par h à partir de q, c'est que la dernière écriture de p_i se fait dans la variable v. Mais comme p_i est caché par h, il doit y avoir une écriture qui oblitère la dernière écriture de p_i avant qu'un processus distinct de p_i ne lise v. Enfin, p_i n'ayant pas d'autre pas après cette dernière écriture, celle-ci va oblitérer w_j avant qu'un autre processus ne lise v, d'où une contradiction. Donc tous les processus de $\mathcal{P}'$ sont cachés par h' à partir de q.

$\square$

Question 8. *Montrer qu'il existe une configuration atteignable q'' de S telle que chaque p_i dans $\mathcal{P}'$ est dans sa section restante en q'', et $q'' \underset{j}{\sim} q'$ pour tout $p_j \in \mathcal{P}' \backslash \mathcal{P}$.*

Réponse. On utilise le résultat de la question précédente pour faire une preuve par récurrence. Soit

$$k = \sum_{p \in \mathcal{P}'} k_p$$

Nous prouvons le résultat par récurrence sur k, nombre de pas cachés de h.

La cas de base avec $k = 0$ est trivial.

Si $k > 0$, avec les notations de la question 7, tous les processus de $\mathcal{P}'$ sont cachés par h', qui contient un pas caché de moins que h. Par hypothèse de récurrence, il existe une configuration atteignable q'' telle que $\forall_j \in \mathcal{P} \backslash \mathcal{P}', q'' \underset{j}{\sim} q''' = res(q, h')$, et telle que tout p_i de $\mathcal{P}'$ est dans sa section restante en q''. Comme le dernier pas de p_i dans h est soit une lecture, soit une écriture oblitérée, on a aussi $q' \underset{j}{\sim} q'''$, et donc $q' \underset{j}{\sim} q''$ pour tout $p_j \in \mathcal{P} \backslash \mathcal{P}'$.

$\square$

Variable couverte. Si une variable est sur le point d'être écrite, tout processus en train d'écrire dans cette variable risque de voir son écriture oblitérée. Plus formellement, on donne la définition suivante : on dit que v est couverte par p_i en q lorsque p_i est prêt à écrire v en q, où q est une configuration, $i \in \{0, \ldots, n-1\}$, et $v \in \mathcal{V}$.

Question 9. *Soient h une histoire finie de S, $q = res(q_0, h)$, et $i \in \{0, \ldots, n-1\}$. Supposons que S a au moins deux processus, et que p_i est caché par h en partant de q_0. Montrer que si p_i atteint sa section critique à partir de q par une histoire finie $h_1 = i\ i\ \ldots i$, alors p_i doit écrire au moins une variable dans le calcul de h_1 à partir de q qui n'est couverte par aucun autre processus en q.*

Réponse. On établit le résultat par l'absurde. Remarquons d'abord que si p_i est caché par h en partant de q_0, il y a un moment dans le calcul par h en partant de q_0 où p_i est dans sa section restante et où toutes ses écritures suivantes ont été oblitérées en q. Supposons que p_i atteigne sa section critique à partir de q par une histoire h_1, sans écrire dans une variable non couverte. Soit h_2 une histoire obtenue en prenant un pas de chaque processus de $\mathcal{P} \backslash \{p_i\}$ qui est prêt à écrire en q. Alors toute écriture de p_i en partant de q est oblitérée (en partant de q) par $h_1 h_2$, donc p_i est caché par $h h_1 h_2$ en partant de q_0.

On utilise alors le résultat de la question 8 : il existe une configuration atteignable q'' telle que $q'' \underset{j}{\sim} q' = res(q, h_1 h_2)$ pour tout $j \neq i$, et où p_i est dans sa section restante. Soit h_3 une histoire équitable non vide (en partant de q'') qui ne contient pas de pas de p_i (h_3 existe car il y a au moins deux processus et que p_i est dans sa section restante en q''). Comme le système a la propriété de progrès global, il y a un processus p_j distinct de p_i qui atteint sa section critique dans un calcul en partant de q'' par un préfixe fini h_4 de h_3. Mais alors h_4 appliquée à q' amène aussi p_j en section critique (d'après le résultat de la question 6, et puisque $q' \underset{j}{\sim} q''$ pour $j \neq i$), ce qui viole la propriété d'exclusion mutuelle.

□

Question 10. *Pour un système vérifiant la propriété de progrès global et une configuration q atteignable, montrer que l'on peut définir une histoire $lat(q)$ telle que $res(q, lat(q))$ est une configuration dans laquelle tous les processus sont dans leur section restante. On appellera* latente *une telle configuration.*

Réponse. On définit $lat(q)$ de la façon suivante : on prend successivement un pas de chaque processus qui n'est pas dans sa section restante en q, et on recommence jusqu'à ce que l'un d'entre eux arrive en section restante (ce qui doit arriver grâce à la propriété de progrès global). On répète alors cette procédure jusqu'à ce que tous les processus soient dans leur section restante.

□

Variable annulée. La combinaison des notions de variable couverte et de processus caché va nous mener au but poursuivi. Informellement, une variable est annulée par un processus si celui-ci ne "communique" pas avec d'autres processus, et couvre cette variable. Plus formellement : si q est une configuration, h une histoire finie et v une variable, on dit que v est *annulée* par h en partant de q s'il existe un processus caché par h en partant de q qui couvre v en $res(q, h)$.

Question 11. *Si S a au moins deux processus, montrer que pour toute configuration atteignable latente q_1, il existe une histoire finie n'utilisant que le processus p_0 et une variable w telles que w est annulée par h en partant de q_1.*

Réponse. Par la propriété de progrès global, il existe une histoire finie h' ne contenant que des 0 telle que p_0 est dans sa section critique en $res(q_1, h')$. Comme q_1 est latente, p_0 est trivialement caché par l'histoire vide en partant de q_1. On applique alors le résultat de la question 9 : p_0 doit écrire une variable dans le calcul par h' à partir de q_1. Soit h le plus court préfixe de h' dans lequel p_0 n'écrit pas, mais est prêt à écrire dans une variable w. Comme p_0 n'écrit pas, il est caché par h à partir de q_1, et couvre w en $res(q_1, h)$. Donc w est annulée par h en partant de q_1.

□

Question 12. *Soit k tel que $0 < k < n$, et tel que pour toute configuration atteignable latente q_1, il existe une histoire finie h utilisant seulement $p_0, \ldots, p_{k-1}$ et telle que k variables distinctes sont annulées par h en partant de q_1. Montrer que l'on peut construire des suites infinies $q_1, q_2, q_3, \ldots$, $h_1, h_2, h_3, \ldots$, et $W_1, W_2, W_3, \ldots$ telles que pour tout i*

1. *W_i est un ensemble de k variables distinctes,*
2. *h_i utilise seulement $p_0, \ldots, p_{k-1}$,*
3. *les variables de W_i sont annulées par h_i en partant de q_i.*

Réponse. Soit $h_1 = h$, et W_1 l'ensemble des k variables annulées par h_1 en partant de q_1. On pose $q_2 = res(q_1, h_1)$. $lat(q_2)$ est une histoire qui commence par un pas de chaque $p_0, p_1, \ldots, p_{k-1}$, et amène ensuite tous ces processus dans leur section restante, en arrivant à une configuration latente. En appliquant les mêmes hypothèses à partir de $res(q_2, lat(q_2))$, on obtient une histoire h'_2 telle que k variables distinctes sont annulées par h'_2 en partant de $res(q_2, lat(q_2))$. On note W_2 cet ensemble de variables, $h_2 = lat(q_2).h'_2$, et $q_3 = res(q_2, h_2)$. On peut alors recommencer cette construction à l'infini, obtenant les propriétés désirées.

□

Question 13. *Avec les hypothèses de la question précédente, montrer qu'il existe une histoire h' utilisant seulement $p_0, \ldots, p_k$ telle que $k + 1$ variables distinctes sont annulées par h en partant de q_1.*

Réponse. Il faut maintenant introduire p_k dans le calcul. On va donc construire des histoires s_i pour $i > 1$ telles que s_i part de q_i et n'utilise que des pas de p_k. D'après la question 8, pour tout $i > 1$ il existe une configuration q'_i atteignable à partir de q_{i-1} telle que $p_0, \ldots, p_{k-1}$ sont dans leur section restante, et $q'_i \underset{k}{\sim} q_i$. En partant de q'_i, p_k peut atteindre sa section critique par une histoire finie $s_i = kk \ldots k$, et donc aussi par s'_i en partant de q_i. On remarque que p_k est trivialement caché par l'histoire vide en partant de q_i, puisqu'il est en section restante en q_i.

D'après le résultat de la question 9, p_k doit écrire dans une variable w_i pendant le calcul par s'_i en partant de q_i, cette variable n'étant pas couverte par $p_0, \ldots, p_{k-1}$. Donc $w_i \notin W_{i-1}$. Soit s_i le plus court préfixe de s'_i où p_k couvre une variable $w_i \notin W_{i-1}$ en partant de q_i. Dans la configuration $res(q_i, s_i)$, chacun des $p_0, \ldots, p_k$ couvre une variable différente.

Il reste à trouver un calcul dans lequel ces processus sont cachés. Soient i et j avec $i < j$ tels que $W_i = W_j$ (i et j existent par un argument de « cage aux pigeons » puisque la suite des q_i est infinie mais qu'il n'y a qu'un nombre fini de variables). On crée alors l'histoire $h = h_1 h_2 \ldots h_{i-1} s_i h_i h_{i+1} \ldots h_{j-1}$. Toutes les écritures de p_k dans s_i se font dans des variables de W_{i-1}. Comme h_i commence avec une série d'écritures dans chacune des variables de W_{i-1}, toutes les écritures de p_k sont oblitérées par h_i en partant de q_i, et p_k n'a plus de pas après. Donc p_k est caché par h. Il reste à montrer que les autres processus le sont aussi.

Soit $q''_{i+1} = res(q_1, h_1 h_2 \ldots h_{i-1} s_i h_i)$. Comme toutes les écritures de p_k sont oblitérées, $q''_{i+1} \underset{l}{\sim} q_{i+1}$ pour $l \in \{0, \ldots, k-1\}$. Ces processus se comportent donc de la même façon dans le calcul original atteignant q_j et dans le calcul par h à partir de q_1. Donc $p_0, \ldots, p_{k-1}$ sont cachés par h en partant de q_1, et couvrent W_{j-1} en $q''_j = res(q_1, h)$.

Comme $p_0, \ldots, p_k$ sont cachés par h en partant de q_1, les variables qu'ils couvrent en q''_j sont annulées. Comme $w_i = w_j \notin W_{j-1}$, les $k+1$ processus annulent $k+1$ variables distinctes.

□

Question 14. *En déduire le résultat final attendu : tout système avec $n \geq 2$ processus vérifiant les propriétés d'exclusion mutuelle et de progrès global doit avoir au moins n variables partagées.*

Réponse. Le résultat final s'obtient simplement à partir des deux questions précédentes, par récurrence sur le nombre de processus.

□

IV Bornes sur l'attente

On appelle *tour* une utilisation de la section critique par un processus. Lorsqu'un processus p_i est désireux d'utiliser la ressource (donc qu'il se trouve dans sa section d'essai), on appelle *attente* de ce processus le nombre de tours effectués par les autres processus avant que p_i ne puisse entrer lui-même en section critique. Le but de cette partie est de donner des bornes supérieures sur l'attente pour certains protocoles d'exclusion mutuelle.

Question 15. *On considère le protocole d'exclusion mutuelle suivant.*

```
repeat
        drapeau[i] ← demande
        j ← tour
        while j ≠ i do begin if drapeau[j] ≠ passif then j ← tour
                                                     else j ← (j − 1) mod n
                             endif
                       end
        drapeau[i] ← dans_sc
until seul(i)
tour ← i
section critique
tour ← (i − 1) mod n
drapeau[i] ← passif
end.
```

---

où seul(i) *est le prédicat défini par*

$$(\forall j \neq i \;:\; drapeau[j] \neq dans\_sc) \wedge drapeau[i] = dans\_sc$$

Montrer que l'attente d'un processus est bornée par $2^{n-1} - 1$ *tours.*

Réponse. Considérons l'exemple d'exécution suivant, avec 4 processus, numérotés de 0 à 3. Nous partons d'une configuration où le processus 3 est en début de ligne 4, et tous les autres en début de ligne 8. On peut alors avoir la séquence d'étapes suivantes.

1. 0 exécute (très vite !) les lignes 8 à 14 ; *tour* vaut 0 puis 3 ;
2. 1 exécute les lignes 8 à 14 ; *tour* vaut 1 puis 0 ;
3. 0 « remonte » et exécute les lignes 1 à 7 ; *tour* vaut 0 ;
4. 0 exécute à nouveau les lignes 8 à 14 ; *tour* vaut 0 puis 3 ;
5. 2 exécute à son tour les lignes 8 à 14 ; *tour* vaut 2 puis 1 ;
6. 0 « remonte » et exécute les lignes 1 à 7 ; *tour* vaut 1 ;
7. 1 « remonte » et exécute les lignes 1 à 7 ; *tour* vaut 1.

Il est alors possible de répéter les étapes 1 à 4. Pendant que 3 « manquait » les valeurs de *tour* qui lui auraient permis d'accéder à la section critique, 0 est passé quatre fois en section critique, 1 deux fois, et 2 une fois. On peut généraliser à un nombre quelconque de processus pour obtenir la borne désirée.

□

Question 16. *On considère la variante suivante.*

```
repeat
        drapeau[i] ← demande
        j ← tour
        while j ≠ i do begin if drapeau[j] ≠ passif then j ← tour
                                                     else j ← (j − 1) mod n
                             endif
                       end
        drapeau[i] ← dans_sc
until seul(i)
section critique
if drapeau[tour] = passif ∨ tour = i
  then tour ← (tour − 1) mod n
endif
drapeau[i] ← passif
end.
```

Montrer que dans un intervalle de temps pendant lequel la variable tour est constante, tout processus peut passer au plus une fois en section critique.

Réponse. Faisons d'abord la remarque suivante : si à un moment *tour* vaut i et $drapeau[i] \neq passif$ alors *tour* ne change pas de valeur jusqu'à ce que p_i ait exécuté la section critique.

Supposons maintenant que $tour = k$ pendant un intervalle de temps où le processus p_j passe deux fois en section critique. On a $tour \neq j$ et $drapeau[tour] \neq passif$; sinon *tour* aurait été modifié après le premier passage de p_j en section critique. De plus, p_k n'est pas passé en section critique avant p_j : sinon il aurait modifié la valeur de *tour* avant le second passage de p_j. Donc $drapeau[k] \neq passif$ et $k = tour$ pendant les deux passages de p_j : p_j reste dans la boucle while après son premier passage, d'où une contradiction.

□

Question 17. *Si $j \in \{1, \ldots, n-1\}$, montrer que dans un intervalle de temps où $drapeau[0] \neq passif$, j peut pénétrer au plus $(n-j)$ fois en section critique.*

Réponse. Considérons $j \in \{1, \ldots, n-1\}$, et examinons le cas du processus p_0 pendant un intervalle de temps où $drapeau[0] \neq passif$. Si $j \geq tour \geq 0$, p_j ne peut passer au plus qu'une fois en section critique : après son premier passage, on ne peut avoir $tour = j$, donc on a $tour > j \geq 0$, ce qui fait ensuite boucler p_j dans la boucle while tant que $drapeau[0] \neq passif$. D'après la question précédente, dans un intervalle de temps où $drapeau[0] \neq passif$, p_j peut pénétrer au plus $n-j$ fois en section critique : $n-j-1$ fois pour les valeurs de *tour* prises cycliquement dans $\{n-1, n-2, \ldots, 1, 0\}$ et vérifiant $j < tour$, et une fois pour $j \geq tour$.

□

Question 18. *En déduire que l'attente d'un processus quelconque est bornée par $\frac{n(n-1)}{2}$ tours. Cette borne peut-elle être atteinte ?*

Réponse. L'attente de p_0 est donc bornée par

$$\sum_{j=1}^{n-1}(n-j) = \frac{n(n-1)}{2}$$

Le même raisonnement est valable pour tous les processus, puisque *tour* varie cycliquement.

Cette borne peut effectivement être atteinte. Prenons l'exemple de $n = 3$, et considérons une exécution qui commence à un instant où *tour* vaut 2. Les processus 0 et 1 demandent la ressource, alors que le processus 2 reste passif. Le processus 1 va donc pouvoir entrer en section critique.

Supposons que le processus 2 demande la ressource pendant que 1 est en section critique. Quand 1 sort de la section critique, il remet son drapeau à passif, et 2 peut alors à son tour entrer en section critique. Supposons maintenant que 1 demande à nouveau à entrer en section critique lorsque 2 y est. Quand 2 sort de section critique, il positionne *tour* à la valeur 1, et le processus 1 peut à nouveau y accéder. Ce n'est qu'ensuite que *tour* sera mis à 0 par le processus 1 et que le processus 0 pourra enfin accéder à la section critique. Au total, le processus 1 a bien accédé deux fois à la section critique, et le processus 2 une fois, pendant que le processus 0 était en attente. La généralisation de cet exemple à n processus est immédiate.

□

Question 19. *Proposer un protocole d'exclusion mutuelle sans interblocage pour lequel l'attente d'un processus est bornée par* $(n-1)$.

Réponse. Le protocole suivant, dû à Eisenberg et Mac Guire, permet d'avoir une attente linéaire.

```
repeat
        drapeau[i] ← demande
        j ← tour
        while j ≠ i do begin if drapeau[j] ≠ passif then j ← tour
                                                     else j ← (j + 1) mod n
                             endif
                   end
        drapeau[i] ← dans_sc
        j ← 0
        while (j < n) ∧ (j = i ∨ drapeau[j] ≠ dans_sc) do
                   begin
                             j ← j + 1
                   end
until (j ≥ n) ∧ (tour = i ∨ drapeau[tour] = passif)
tour ← i
section critique
j ← tour + 1 mod n
while (j ≠ tour) ∧ (drapeau[j] = passif) do begin j ← (j + 1) mod n end
tour ← j
drapeau[i] ← passif
end.
```

Cette fois, lorsqu'un processus quitte la section critique, il désigne comme unique successeur le premier (au sens de l'ordre cyclique précédent) processus en attente. Cela assure que l'attente d'un processus sera bornée par $n-1$ tours.

□