Exercice 3 – Lecteurs-Écrivains

On va illustrer l'accès à des données partagées dans la sémantique lecteurs/écrivains (les lectures ne sont pas destructrices, contrairement à l'accès en producteur/consommateur).

Scénario

Ici, il y a une seule donnée partagée qui s'appelle Data, utilisée par des threads lecteurs et des threads écrivains. Les threads écrivains incrémentent la variable partagée de différentes valeurs tant que cette variable reste inférieure à une valeur maximum, donnée par VALMAX, alors tous les threads s'arrêtent, aussi bien les lecteurs que les écrivains.

L'application doit fonctionner de la façon suivante :

1. Le premier lecteur interdit l'accès aux éventuels écrivains suivants, mais pas aux autres lecteurs.
2. Un écrivain interdit tout autre accès.
3. Le point 1 peut provoquer la famine pour les écrivains : si un écrivain arrive, il doit attendre la fin du flot des lecteurs avant d'accéder à la ressource.
4. Pour éviter ce dysfonctionnement, qui peut se traduire par la famine pour les écrivains, l'arrivée d'un écrivain doit bloquer tous les lecteurs suivants. Voici le comportement qui sera adopté : lorsqu'un écrivain se présente, il attend la sortie des lecteurs courants, les lecteurs qui le suivent sont bloqués. Lorsque tous ces lecteurs sont sortis, l'écrivain modifie la donnée, et on sert le (ou les) écrivains(s) qui seraient arrivés entre-temps.
5. Lorsqu'un écrivain a fini une mise à jour, il donne d'abord accès aux écrivains qui attendent, si il y en a, les lecteurs passeront après. Pour cela, donner une priorité aux threads écrivains afin que l'OS réveille d'abord des écrivains en attente avant des lecteurs en attente (cf *The Little Book of Semaphores*, p. 85, listing 4-23 et 4-24).
6. Pour que le déroulement soit lisible, implémenter une durée aléatoire de lecture/écriture (sous forme de sleep).

1 Compilation

La compilation est réalisée en utilisant cmake via le fichier *CMakeLists.txt* :

```
[1]  cmake_minimum_required(VERSION 3.10)

[2]  project(project(Exercice3_Lecteurs_Ecrivains VERSION 1.0) VERSION 1.0)

[3]  set(CMAKE_C_FLAGS "${CMAKE_C_FLAGS} -Wall -Wpedantic -Wextra")

[4]  include_directories(PUBLIC include)

[5]  set(SOURCE_FILES
         src/main.c
         src/reader_writer.c
         src/lightswitch.c
         src/data.c)

[6]  find_package(Threads REQUIRED)

[7]  add_executable(reader_writer ${SOURCE_FILES})

[8]  target_link_libraries(reader_writer PRIVATE Threads::Threads)
```

`[1]` version de `cmake` utilisée
`[2]` nom et version du projet
`[3]` ajout des flags habituels pour l'affichage de tous les warnings
`[4]` le répertoire *include* contient les headers nécessaires à la compilation
`[5]` définition de la variable `SOURCE_FILES` contenant les fichiers sources du projet
`[6]` ajout du package pour l'utilisation des threads
`[7]` la compilation des fichiers sources produit l'exécutable `reader_writer`
`[8]` le projet utilise la librairie `Threads` du package `Threads`

La commande `cmake -DCMAKE_C_FLAGS=-g ..` [1] lancée depuis le sous-répertoire *build* permet de générer le *Makefile* :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ cmake -DCMAKE_C_FLAGS=-g ..
-- The C compiler identification is GNU 7.5.0
-- The CXX compiler identification is GNU 7.5.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Looking for pthread.h
-- Looking for pthread.h - found
-- Looking for pthread_create
-- Looking for pthread_create - not found
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
-- Configuring done
-- Generating done
-- Build files have been written to: /home/lbin/2021_TR/ex3-le
cteurs-ecrivains/build
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile
```

Par défaut, le *Makefile* est généré en mode debug. Pour le générer en mode release, j'utilise la commande `cmake -DCMAKE_BUILD_TYPE=Release` ***`path`***, où ***`path`*** est l'endroit où se trouve le fichier *CMakeLists.txt*.

1. `-g` permet d'ajouter des informations sur le code source

La commande `make` permet de compiler le projet à partir du *Makefile* pour produire l'exécutable *reader_writer* :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ make
Scanning dependencies of target reader_writer
[ 20%] Building C object CMakeFiles/reader_writer.dir/src/main
.c.o
[ 40%] Building C object CMakeFiles/reader_writer.dir/src/read
er_writer.c.o
[ 60%] Building C object CMakeFiles/reader_writer.dir/src/ligh
tswitch.c.o
[ 80%] Building C object CMakeFiles/reader_writer.dir/src/data
.c.o
[100%] Linking C executable reader_writer
[100%] Built target reader_writer
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ make clean
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ make
[ 20%] Building C object CMakeFiles/reader_writer.dir/src/main
.c.o
[ 40%] Building C object CMakeFiles/reader_writer.dir/src/read
er_writer.c.o
[ 60%] Building C object CMakeFiles/reader_writer.dir/src/ligh
tswitch.c.o
[ 80%] Building C object CMakeFiles/reader_writer.dir/src/data
.c.o
[100%] Linking C executable reader_writer
[100%] Built target reader_writer
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ls
CMakeCache.txt  cmake_install.cmake  reader_writer
CMakeFiles      Makefile
```

2 Tests

2.1 Cas nominal

La programme a été testé avec des temporisations aléatoires plus longues pour les threads lecteurs que pour les threads écrivains :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ./reader_writer 10 3 8
writer 2 writes   2 TOTAL>   2
reader 4 reads   2
reader 2 reads   2
reader 7 reads   2
writer 3 writes   3 TOTAL>   5
writer 1 writes   1 TOTAL>   6
reader 10 reads   6
reader 1 reads   6
reader 6 reads   6
reader 5 reads   6
reader 8 reads   6
writer 1 writes   1 TOTAL>   7
writer 3 writes   3 TOTAL>   8
reader 5 reads   8
writer 2 writes   2 TOTAL>   8
reader 3 reads   8
reader 7 reads   8
reader 6 reads   8
reader 9 reads   8
reader 4 reads   8
reader 2 reads   8
writer 1 writes   1 TOTAL>   8
reader 8 reads   8
reader 1 reads   8
reader 10 reads   8
```

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ./reader_writer 10 3 8
reader 5 reads   0
reader 8 reads   0
writer 3 writes   3 TOTAL>   3
writer 1 writes   1 TOTAL>   4
writer 2 writes   2 TOTAL>   6
reader 4 reads   6
writer 3 writes   3 TOTAL>   8
reader 8 reads   8
reader 2 reads   8
reader 1 reads   8
reader 3 reads   8
writer 2 writes   2 TOTAL>   8
reader 6 reads   8
reader 7 reads   8
writer 1 writes   1 TOTAL>   8
reader 5 reads   8
reader 9 reads   8
reader 4 reads   8
reader 10 reads   8
```

Le lightswitch des lecteur et celui des écrivains permettent bien de laisser passer plusieurs lecteurs ou plusieurs écrivains dans la zone d'accès à la donnée :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ./reader_writer 5 2 2
lightswitch locked
reader 4 reads   0
lightswitch locked
writer 1 writes   1 TOTAL>   1
writer 2 writes   2 TOTAL>   2
lightswitch unlocked
lightswitch unlocked
lightswitch locked
reader 2 reads   2
reader 1 reads   2
reader 3 reads   2
lightswitch unlocked
lightswitch locked
writer 1 writes   1 TOTAL>   2
lightswitch unlocked
lightswitch locked
reader 4 reads   2
lightswitch unlocked
lightswitch locked
reader 5 reads   2
lightswitch unlocked
```

2.2 Cas d'erreur

Le programme attend en paramètres le nombre de threads lecteurs, de threads écrivains et la valeur maximale de la donnée.

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer
usage: ./reader_writer readers writers maxvalue
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 255
usage: ./reader_writer readers writers maxvalue
```

Le nombre de *readers* (threads de lecture de la donnée partagée) doit être supérieur à 0 :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer a a a
invalid readers count
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 0 a a
invalid readers count
```

Le nombre de *writers* (threads d'incrémentation de la donnée partagée) doit être supérieur à 0 :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 1 a a
invalid writers count
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 1 0 a
invalid writers count
```

La valeur maximale de la donnée doit être supérieure à 0 :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 1 1 a
invalid max value
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ ./reader_writer 1 1 0
invalid max value
```

Des valeurs trop grandes stoppent également le programme : La valeur maximale de la donnée doit être supérieure à 0 :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build
$ ./reader_writer 1000000000 1000000000 1
malloc write_threads: Cannot allocate memory
```

2.3 Valgrind

Le test du programme avec *valgrind* permet de voir qu'il n'y a pas de fuite mémoire (tous les espaces mémoire alloués ont bien été libérés) :

```
lbin@tr-ubuntu18-server:~/2021_TR/ex3-lecteurs-ecrivains/build$ valgrind ./reader
_writer 1 1 1
==30025== Memcheck, a memory error detector
==30025== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==30025== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==30025== Command: ./reader_writer 1 1 1
==30025==
reader 1 reads   0
writer 1 writes   1 TOTAL>   1
reader 1 reads   1
==30025==
==30025== HEAP SUMMARY:
==30025==     in use at exit: 0 bytes in 0 blocks
==30025==   total heap usage: 10 allocs, 10 frees, 2,232 bytes allocated
==30025==
==30025== All heap blocks were freed -- no leaks are possible
==30025==
==30025== For counts of detected and suppressed errors, rerun with: -v
==30025== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```