

## Homework 3. Producers-consumers.

A ready-to-run program is enclosed which implements the producers-consumers program from the lecture using the pthreads library.

Use

make

to build the executable binary "prodcon".

The program "prodcon" expects 3 command-line arguments:

<nr\_producers> is the number of producers. The number of consumers is automatically set to this number (the number 1 means 1 producer and 1 consumer). Reasonable values range from 1 to ca. 1000.

<queue\_size> is the size of the queue shared by the producers and consumers. Reasonable values range from 1 to 1000000000.

<nr\_iterations> is the number of reads/writes to the shared queue, in each producer and consumer. Reasonable values range from 1 to  $2^{30}$ .

The file prodcon.c includes sync.h in which wrapping macros are defined for synchronization operations and structures (typedefs). If the symbol SYNC\_SEM is not defined (i.e. -DSYNC\_SEM is omitted in CFLAGS in Makefile), the synchronization involves mutex and cond operations of the pthread library.

a) Define the symbol SYNC\_SEM in CFLAGS in Makefile, write macro and type definitions to designated places in sync.h (feel free to also extend sync.c if needed) so that semaphores are used for thread synchronisation instead of mutex and conditional variables (i.e. simulate the mutex/cond synchronisation with semaphores). Do NOT make any changes to the files prodcon.c and prodcon.h (modify only Makefile, sync.h, and/or sync.c)! Test your implementation.

The files sync.c and sync.h already contain (if the symbol SYNC\_SEM is defined)

```
#include <semaphore.h>
```

Do not include any other header files.

Refer to the manual pages

```
man sem_overview
```

and manuals to semaphore operations of your choice (also available at e.g. [https://man7.org/linux/man-pages/man7/sem\\_overview.7.html](https://man7.org/linux/man-pages/man7/sem_overview.7.html)).

b) Read the manual on System V semaphores

```
man semop
```

(also available at e.g. <https://man7.org/linux/man-pages/man2/semop.2.html>, and/or a programming tutorial at <https://www.softprayog.in/programming/system-v-semaphores> which includes a simplified producer-consumer example). Extend the functionality of the program so that it uses System V semaphores for synchronization of threads when compiled with the flag -DSYNC\_SYSV, while retaining the original functionality (the program must also work for CFLAGS -DSYNC\_SEM and -DSYNC\_MUTEX). Test your implementation with System V semaphores.

c) Compare the performance mutex/cond synchronisation with the semaphore synchronization and System V semaphore synchronisation. If one of them is a clear winner or loser, state which one it is. Enclose a table and/or a graph which supports your statement.

d) Copy prodcon.c to prodcon1.c. Modify the code of prodcon1.c so that there is at most one thread sleeping on a condition variable (i.e. each thread uses its own condition to sleep). Include prodcon1 in Makefile. Compare the performance of prodcon1 with its prodcon counterparts.

Hand out a ZIP file hw3.zip containing relevant files, and a report hw3.pdf. Report the maximal number of threads which your operating system allows per process.