Assignment 3 – Programming

2018 COMP3230a

Objectives

- An assessment task related ILO 4 – demonstrate knowledge in applying system software and tools available in modern operating system for software development

- Tools
  - Pthreads library

- Concurrency
  - Producer/Consumer (Bounded-buffer) model

Task – Counting Word Frequency

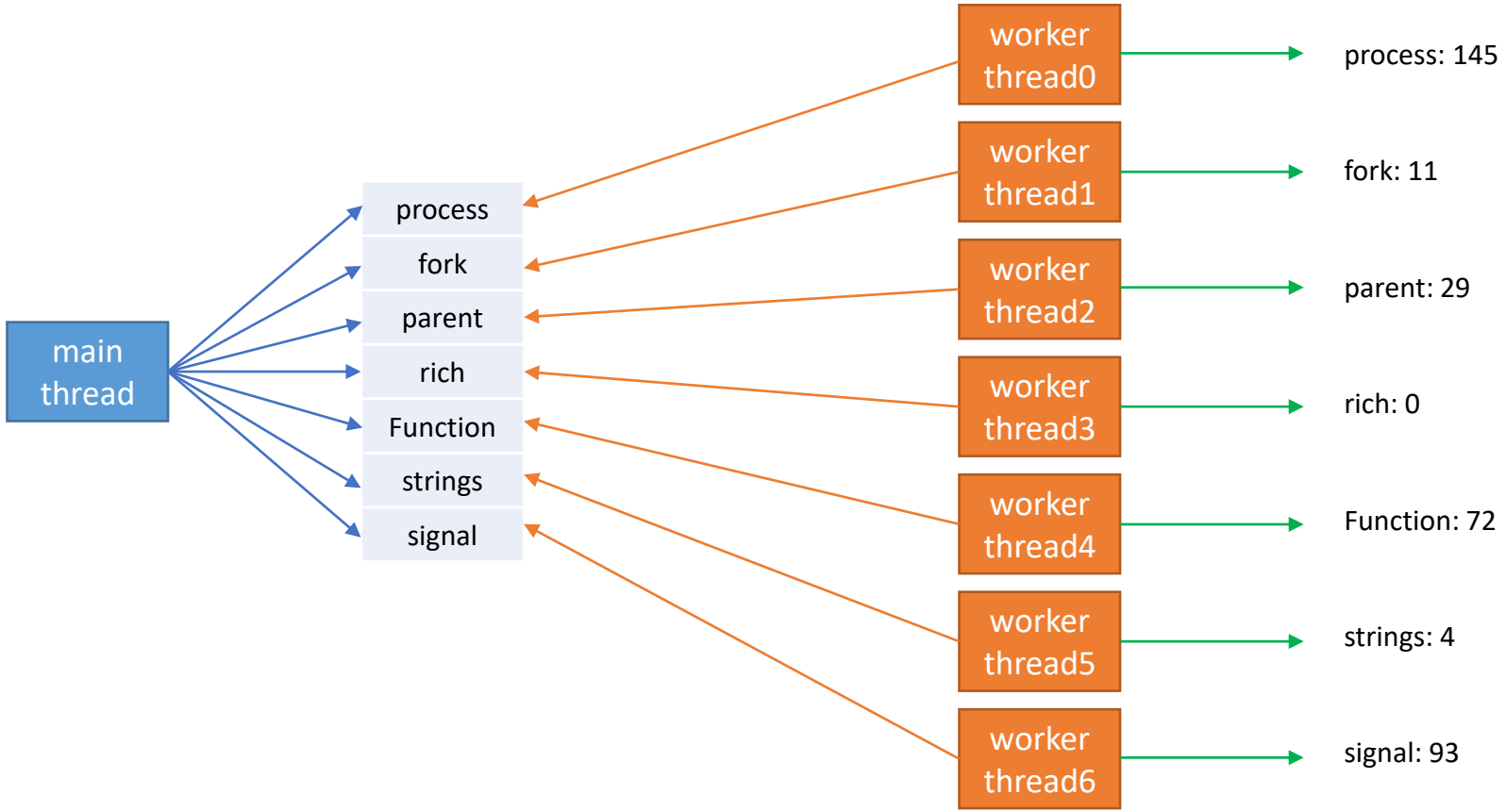
- You are going to implement a multithreaded program – to count the frequencies of the selected keywords in a document.
- You are going to structure the solution in the form of producer/consumer model, such that the master (main) thread works as the producer that places keywords to a task pool (bounded-buffer), while the worker threads work as the consumers that obtain a keyword from the task pool and process it.

Structure of the Assignment

- Step 1: Examine the sequential program
  - Provide File I/O, counting logics
- Step 2 [Optional]: Build a simple multithreaded program
  - Create ***N*** threads to work for searching ***N*** keywords
  - Master thread waits for all threads to terminate
- Step 3: Coordinated Multithreaded Program
  - Threads Management
  - Synchronization
  - Producer/consumer interaction

Step 2 [Optional]

- With each keyword, main process (main thread) creates one thread to perform the searching and counting



Step 2 – Note

- Pthreads library does not have a built-in method to label and identify a thread
  - i.e., you cannot use getpid( )
  - In Linux, you can use gettid( ) to find its thread id, but isn't a portable method
- Well, you can pass an ID to the thread during thread creation.
- The output is really a bit messy.

```
rich: 0
fork: 11
process: 145
Function: 72
Worker thread 0 has terminated
Worker thread 1 has terminated
parent: 29
signal: 93
strings: 4
Worker thread 2 has terminated
Worker thread 3 has terminated
Worker thread 4 has terminated
Worker thread 5 has terminated
Worker thread 6 has terminated
```

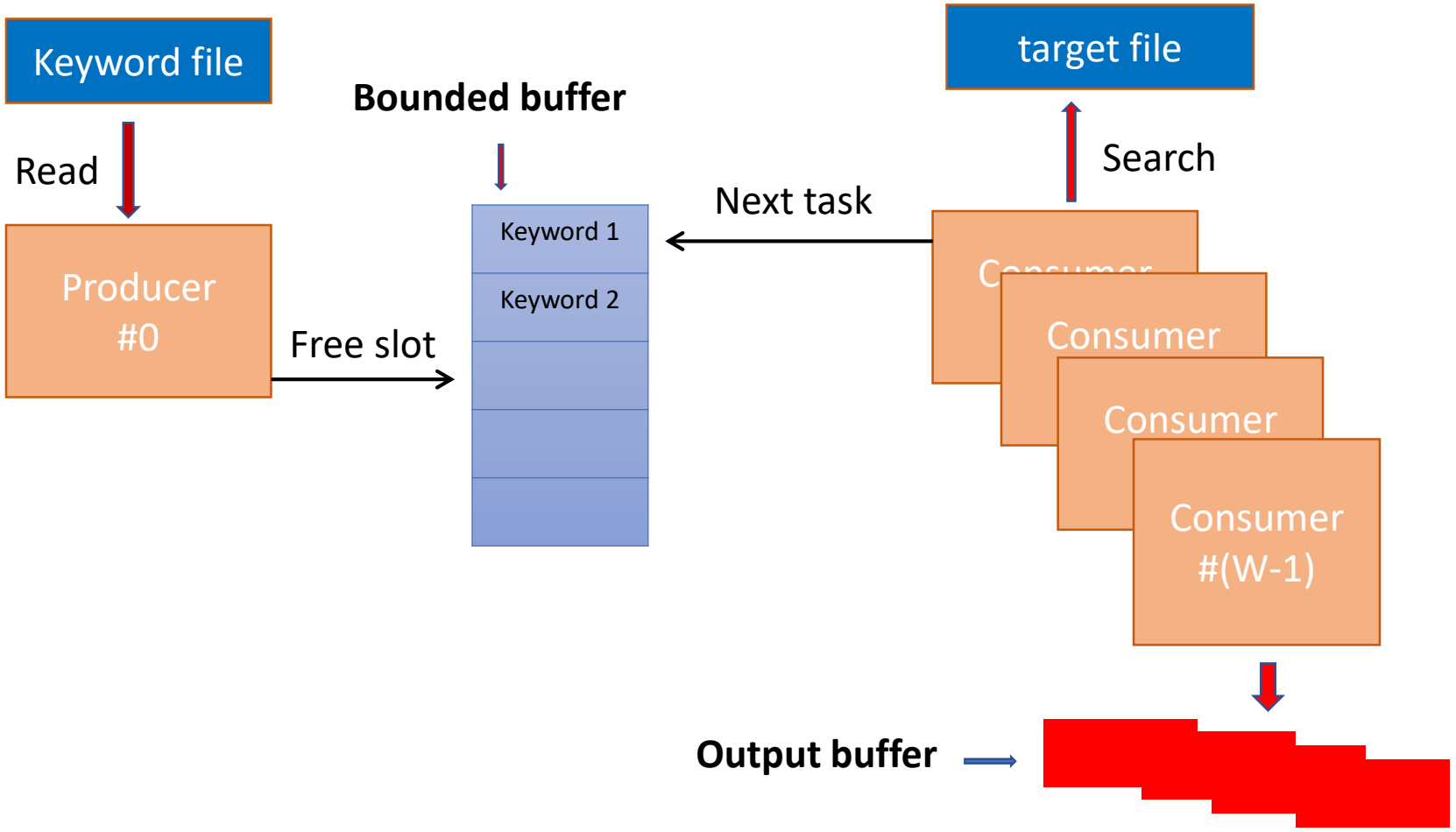
Step 3

- You are required to make use of bounded-buffer model to coordinate all activities between main thread (producer) and worker threads (consumers)

- We have two additional arguments in this part:
  - Number of workers → number of worker threads

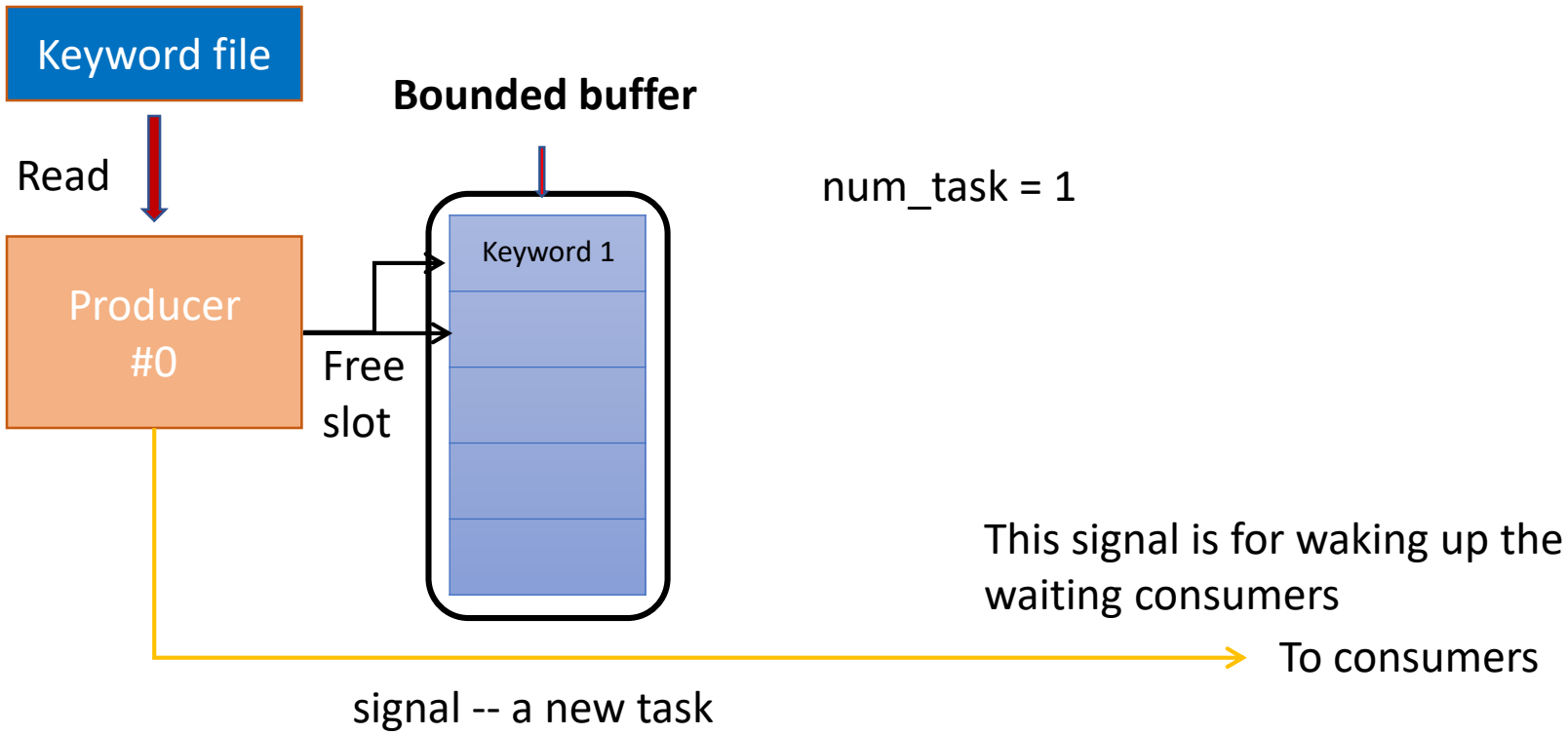
  - Size of bounded buffer → the number of entries in the string array
    - Each entry stores a keyword string

Producer/Consumer Interaction

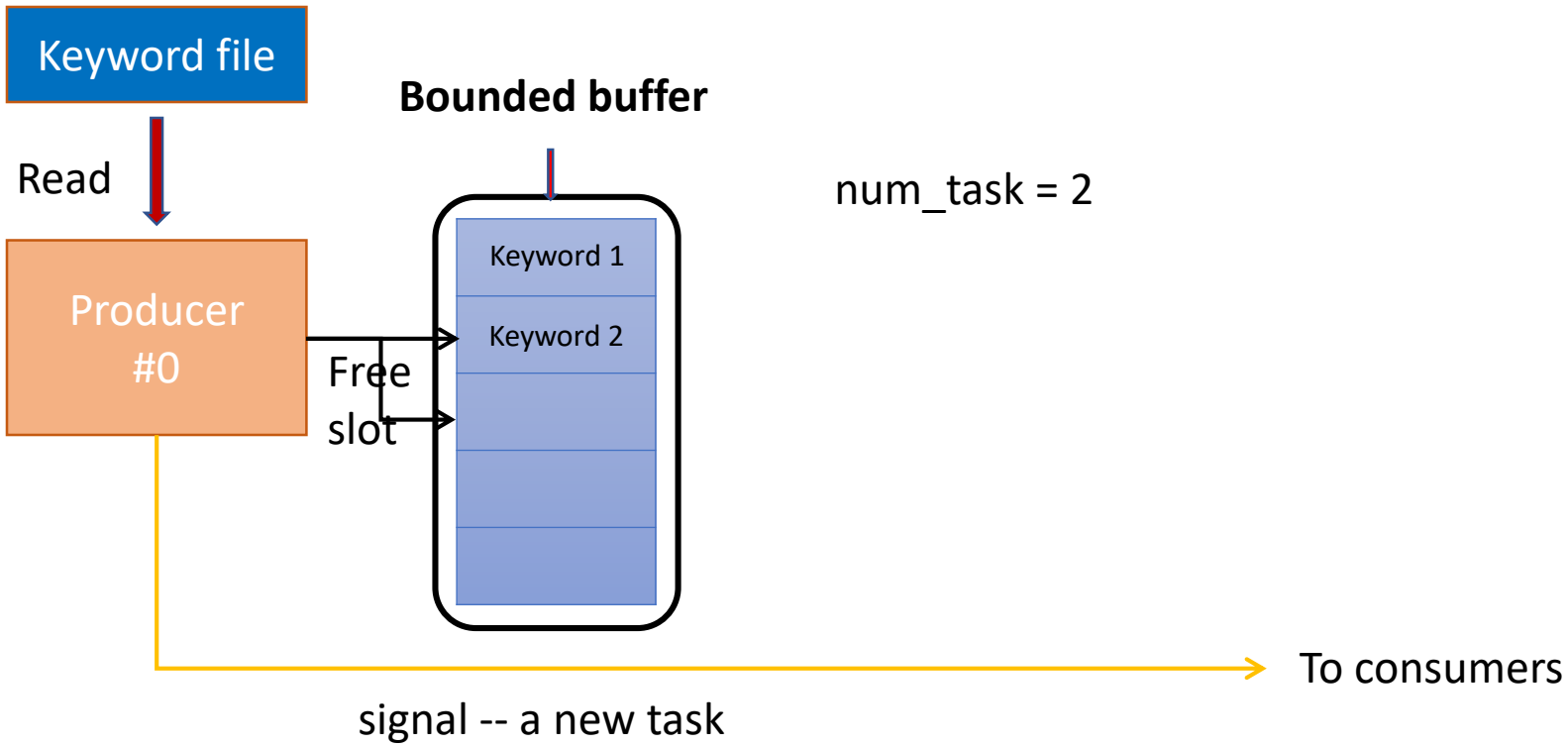


Producer/Consumer Interaction

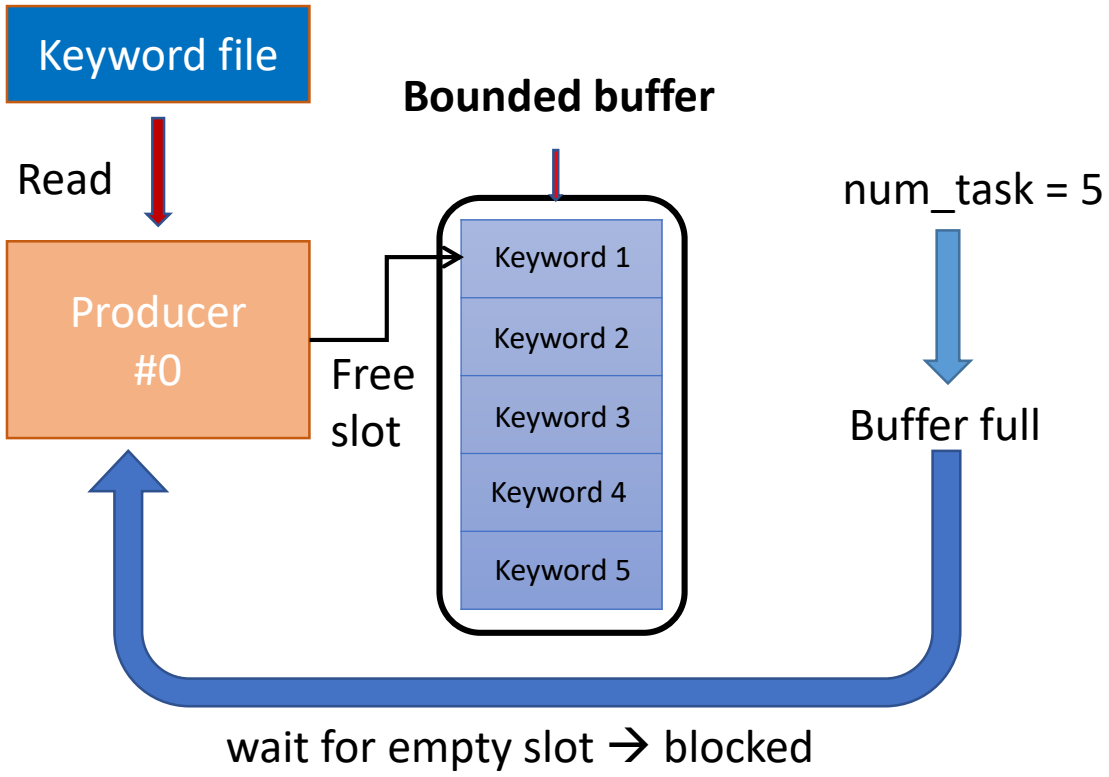
- Use num_task as the indicator of the number of tasks in the bounded buffer



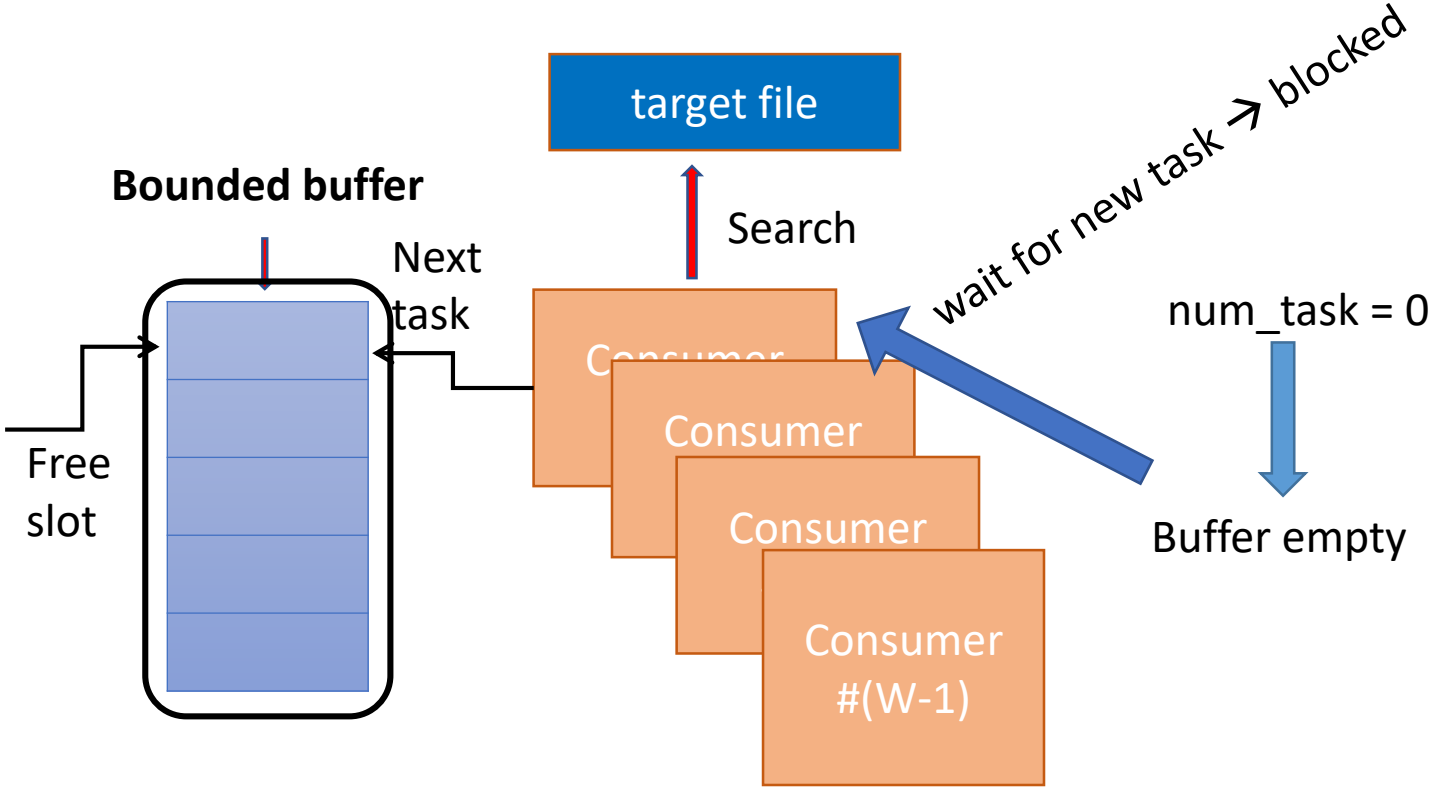
Producer/Consumer Interaction



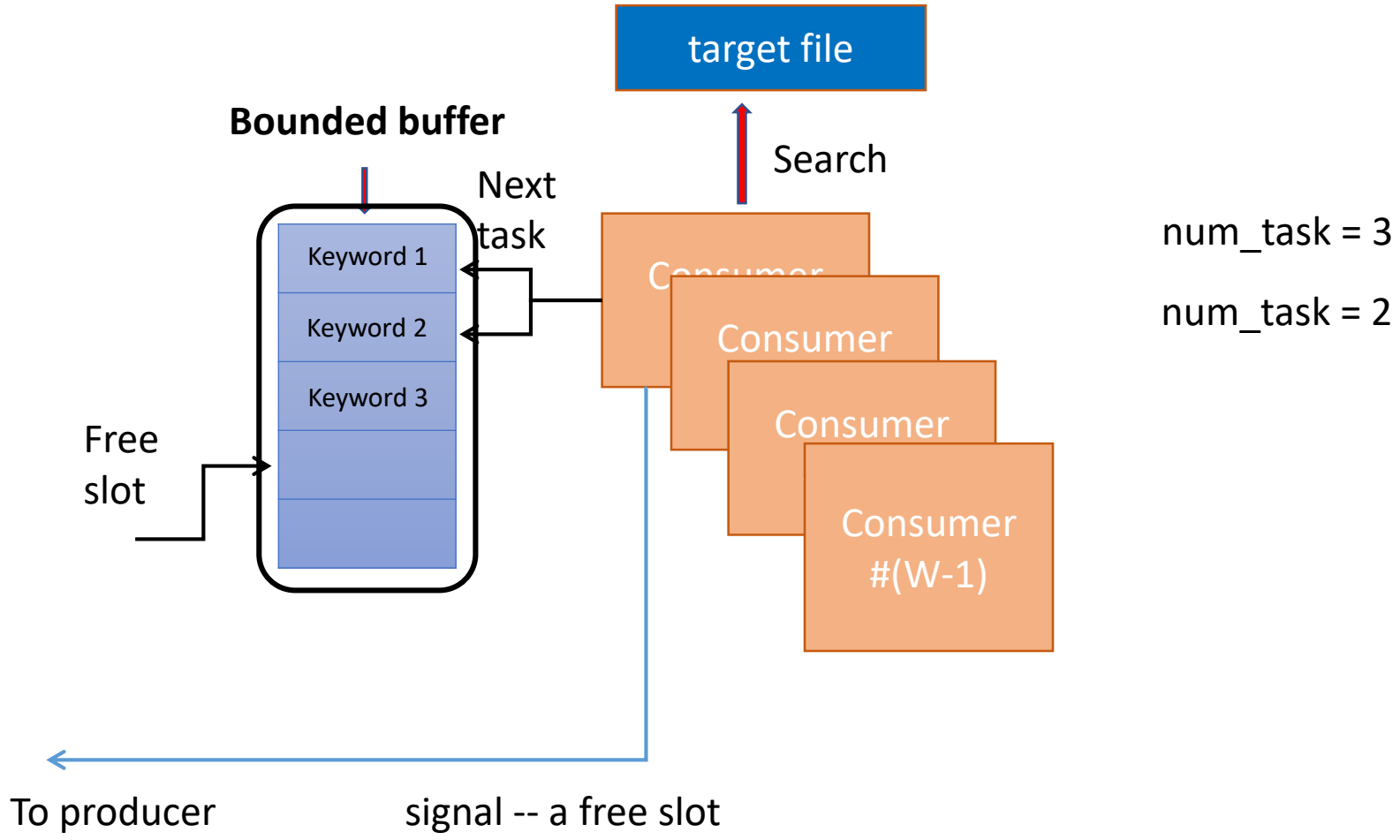
Producer/Consumer Interaction



Producer/Consumer Interaction



Producer/Consumer Interaction



Return Results

- Well, this looks like another Producer/Consumer interaction, where we have many producers and a single consumer
- However, as consumer does not touch on the buffers before termination of all producers; this simplifies the interaction
- What we need is to use a lock to guard against the concurrent access of the the output buffers

Termination

- One of the issues appears in Producer/Consumer Interaction is to have a mechanism to inform consumers to quit
- In particular, if some of the consumers are sleeping in the condition variable queue and waiting for new tasks to arrive
- Suggest mechanism is to take advantage of the Producer/Consumer logic by placing extra tasks into the task pool, which work as a “message” to inform all consumers to terminate
- Well, in our case, the task is a keyword; thus, we have to make use of specialized keyword as a sentinel symbol to represent terminal stage has reached.

Submission of Assignment

- You should name your program "thrwordcnt_StudentNumber.c" (replace StudentNumber with your HKU student number)
- Submit your program to the course's Moodle web site (to Assignment Three submission page)
- Add your signature in the header of the submitted program and make clear documentation

```
/*********************************************************
* Filename: thrwordcnt_3015234567.c
* Student name: Harry Potter
* Student no.: 3015234567
* Development platform: Course VM
* Compilation: gcc thrwordcnt_3015234567.c –pthread –o thrwordnt
* Remark: All done
*********************************************************/
```

Grading Criteria

Documentation (0.5 points)	• Include necessary documentation to clearly indicate the logic of the program • Include required student's info at the beginning of the program
Correctness of the program (11.5 points)	• The program should be compiled and executed successfully, and the total no. of threads created in this program should be equaled to the no. of workers (input parameter).
	• Worker threads must be executed in parallel.
	• Must use the producer/consumer model to coordinate the threads.
	• Can work with different numbers of worker threads, sizes of task pool, and numbers of keywords, and return "correct" results as compared to sequential program. (Note: counting results could be displayed in any order; should not have duplication.)
	• Worker threads must pass the counting results to the master thread.
	• Worker threads can detect the termination condition and terminate successfully.
	• Master thread should wait for all worker threads to terminate before displaying all counting results