The Unisex Bathroom: Fairness versus Performance

12-April-2006

Assistant Professor David J. Powers
Department of Mathematics and Computer Science
Northern Michigan University
Marquette, MI 49855
phone: 906/227-2501
fax: 906/227-2010
dpowers@nmu.edu

Keywords: Concurrent Programming, Parallel Computing, Distributed Processing, Multithreaded Applications, Resource Allocation

Abstract

The Unisex Bathroom is an interesting problem in resource allocation that is useful for illustrative purposes in the study of computer operating systems and parallel processing [1, 9]. The Unisex Bathroom problem consists of a bathroom with n stalls and some number of males and females that share the bathroom. A male or female may be using a stall in the bathroom, waiting to use the bathroom, or doing something else. This paper will analyze different strategies that can be used to allocate bathroom stalls to the males and females. Each strategy will be measured by the following criteria: deadlock, denial, fairness and performance (wait time). This paper will contain ideal and realistic goals for any strategy. Some strategies could result in denial, a situation where some individuals may never get to use the bathroom. The best strategies are measured by fairness and performance. Measured data for some strategies will be included in this paper.

1 Introduction

The Unisex Bathroom problem consists of a bathroom with *n* stalls and some number of males and females that share the bathroom. A male or female may be using a stall in the bathroom, waiting to use the bathroom, or doing something else. There are different ways or strategies that can be used in scheduling individuals to use the bathroom. This paper presents details on how to evaluate different strategies based on fairness and performance.

1.1 Problem Assumptions

- Males and females cannot use the bathroom at the same time.
- An individual will spend some time doing other things (other than using the bathroom)
- An individual will spend some time using the bathroom
- An individual cannot do other things while waiting to use the bathroom
- An individual does not like to wait to use the bathroom
- Each individual may spend different amounts of time in the bathroom

1.2 Problem Considerations

- **Deadlock**

  A state where all the individuals that are waiting to use the bathroom never get a chance. In this case, all the individuals are denied an opportunity to use the bathroom. This means that each individual is waiting for a stall to become available. In theory, since there are *n* ($n \geq 1$) stalls available in the Unisex Bathroom problem, at least one individual will always get a chance to use the bathroom, and deadlock would not occur. However, when simulating this problem in software, the improper use of semaphores can lead to a situation where all remaining individuals waiting to use the bathroom never get a chance, which is deadlock.

- **Denial**

  At least one individual never gets a chance to use the bathroom. If the bathroom is in use by one sex and the scheduling strategy allows more members of that sex to keep using the bathroom, then individuals of the opposite sex will be denied use of the bathroom. This case is not deadlock, because even though some individuals never get a chance to use the bathroom, other individuals continue to get a chance.

- **Fairness**

  Every individual waiting to use the bathroom gets to use the bathroom eventually. Denial would not be fair since some individuals would get to use the bathroom while others would not. A 100 % fair strategy would allow individuals to enter the bathroom in the order they arrive although this strategy may not be the most efficient in terms of throughput or performance.

- **Performance**

  Performance can be measured by the average amount of wait time for individuals waiting to use the bathroom. Smaller wait times mean better performance. Wait time is the elapsed time between requesting the bathroom and getting into the bathroom. Throughput or performance could also be measured by how many individuals are able to use the bathroom in a given amount of time.

1.3 Fair Strategies

The ideal strategy for using the Unisex Bathroom would avoid deadlock, avoid denial, be 100 % fair and would minimize the amount of wait time for individuals that need to use the bathroom. By definition, all strategies for this problem will avoid deadlock, and any fair strategy will avoid denial. However, some of these "fair" strategies may not be 100 % fair to all individuals waiting to use the bathroom. In order to find the best strategies from our set of "fair" strategies, fairness and performance must be evaluated. A strategy with 100 % fairness may exhibit poor performance. Likewise, a strategy with good performance may not be fair. The best strategy will provide the best combination of fairness and performance.

The following are possible "fair" strategies for using the Unisex Bathroom:

1 Individuals arriving at the bathroom will get to use the bathroom in the order that they arrived. This strategy is 100 % fair to all individuals. If someone of the opposite sex is currently using the bathroom, then the individual must wait for the bathroom to empty. In this case there is a single queue of individuals waiting to use the bathroom. The technique of *passing the baton* [1, 10] is used.

2 If someone of the opposite sex is currently using the bathroom, then an arriving individual must wait for the bathroom to empty. Until the bathroom is empty, up to *n-1* additional members of the sex that are currently using the bathroom may use any free stalls (*n* is the total number of stalls). This strategy is not 100% fair since some individuals may get to use the bathroom before others of the opposite sex that arrived earlier. In this case there are two (2) waiting queues, one for each sex. When the bathroom is empty, up to *n* waiting members of the opposite sex may enter the bathroom.

3 If someone of the opposite sex is currently using the bathroom, then an arriving individual must wait for the bathroom to empty. No additional members of the sex that are currently using the bathroom may enter the bathroom. This strategy is not 100% fair since some individuals may get to use the bathroom before others of the opposite sex that arrived earlier. However, this strategy is "more fair" than strategy 2, since an arriving individual of the opposite sex will be the next person to enter. In this case there are two (2) waiting queues, one for each sex. When the bathroom is empty, up to *n* waiting members of the opposite sex may enter the bathroom.

2 Implementation

The Unisex Bathroom problem can be simulated with a multi-threaded application [2,10] or as multiple tasks executing in parallel [6,7].

2.1 Individuals

The Person logic is quite simple and is shown below:

```
Person[i:1..k] ::  while (true)
        do_something;   // random delay
        get_stall;      // wait here if stall is not available
        use_stall;      // random delay
        release_stall;
end of while loop;
```

Each individual is in one of three (3) possible states: doing something, using the bathroom, or waiting to use the bathroom. These states are represented in the person logic above. Each individual is represented by a single thread or a single task. Doing something and using the bathroom time delays can be fixed or randomized. Fixed times insure that each individual will spend the same amount of time doing things and using the bathroom. Randomized times can be used to represent the fact that activity times may fluctuate slightly from one iteration to another. Averages over a large number of iterations (20 or more) using randomized times seemed to compare reasonably well with the fixed values for times.

2.2 Stalls

The bathroom stalls are the shareable resource that can be represented by shared memory. Mechanisms such as mutexes, semaphores, or monitors can be used to synchronize the use of the shared resources (stalls) [2,4]. Message queues could be used for the Unisex Bathroom but the implementation would be much more awkward [1,10].

The get_stall method in the individual logic above can be implemented to represent the various strategies listed above. The release_stall method would also depend on the strategy used because the notification logic may be different.

2.3 Individual States

Each individual is in one of 3 possible states: doing something (D), using the bathroom (B) or waiting to use the bathroom (W). With *k* individuals, there are 3 ^ *k* states in the sample space for the Unisex Bathroom. However, many of these states are not possible due to the special constraints and assumptions for this problem. For example, since males and females cannot be in the bathroom at the same time, the following state is not possible (assume 3 stalls and 4 men and 4 women):

M (B), M(B), M(D), M(D), F(D), F(W), F(W), F(B)

2 males using the bathroom
2 males doing something else
1 female doing something else
2 females waiting to use the bathroom
1 female using the bathroom

Two (2) males would not be sharing the bathroom with one female.

2.4 Arrival Patterns

The arrival patterns of individuals will affect how efficiently the bathroom stalls are utilized. Performance will also be affected by the allocation strategy and how much the bathroom is used.

Single Individual Alternating

In this pattern, the next individual to arrive would be of the opposite sex. This pattern would look as follows:

◄ **M F M F M F M F M F M F M F M....**

Group Alternating (Size *n*)

In this pattern, a group of individuals of the opposite sex would arrive next. The number of individuals in this group would equal *n*, the number of stalls available in the bathroom. If n = 4, the pattern would look as follows:

◄ **M M M M F F F F M M M M ....**

Random Pattern

In this pattern, individuals of both sexes arrive in a random sequence. This pattern could look as follows:

◄ **M M F M F M F F M M M F ....**

2.5 Test Methodology

The Unixsex Bathroom Problem will be simulated by using multiple tasks running concurrently, one task per individual. Each arrival pattern will be tested with each allocation strategy. The simulation could represent low or high usage of the bathroom resource. However, extremely low usage would not differentiate the performance between the different allocation strategies, since each individual would use the bathroom on a FIFO (first-in-first-out) basis. Also, there would be no waiting by any individual and only one (1) of the *n* stalls would be utilized.

Maximum or peak usage will be used to measure differences in performance for different arrival patterns and allocation strategies. Maximum or peak usage occurs when all individuals arrive to use the bathroom at about the same time. Except for the first few individuals, everyone else has to wait to use the bathroom. The goal of the test results is to find the best strategy to allocate usage of the bathroom when individuals are waiting. Actual or realistic usage of the bathroom is probably less than maximum or peak usage. However, maximum or peak usage will accentuate the differences in performance.

3 Test Results

This paper presents different strategies used to allocate bathroom stalls in the Unisex Bathroom problem. A "fair" strategy will avoid denial. The best strategy will provide the best combination of fairness and performance. The results of 3 strategies are summarized as follows:

Strategy 1

Individuals arriving at the bathroom will get to use the bathroom in the order that they arrived. This strategy is 100 % fair to all individuals. If someone of the opposite sex is currently using the bathroom, then the individual must wait for the bathroom to empty. In this case there is a single queue of individuals waiting to use the bathroom.

Strategy 2

If someone of the opposite sex is currently using the bathroom, then an arriving individual must wait for the bathroom to empty. Until the bathroom is empty, up to *n-1* additional members of the sex that are currently using the bathroom may use any free stalls (*n* is the total number of stalls). This strategy is not 100% fair since some individuals may get to use the bathroom before others of the opposite sex that arrived earlier. In this case there are two (2) waiting queues, one for each sex.

Strategy 3

If someone of the opposite sex is currently using the bathroom, then an arriving individual must wait for the bathroom to empty. No additional members of the sex that are currently using the bathroom may enter the bathroom. This strategy is not 100% fair since some individuals may get to use the bathroom before others of the opposite sex that arrived earlier. However, this strategy is "more fair" than strategy 2, since an arriving individual of the opposite sex will be the next person to enter. In this case there are two (2) waiting queues, one for each sex. When the bathroom is empty, up to *n* waiting members of the opposite sex may enter the bathroom.

Test Case: Maximum (Peak) Load

Allocation Strategy

Arrival Pattern	**1**	**2**	**3**
Single	540	135	135
Group	181	135	135
Random	263	135	135

Figure 1: Total Time to Service 60 Individuals with 3 stalls

4 Conclusions

- Strategy 1 is 100 % fair but demonstrates much lower performance than Strategies 2 or 3. The performance of this strategy varies considerably depending on the arrival pattern. Single alternating individuals effectively only use one (1) of the *n* stalls. Grouping individuals by sex and setting the group size equal to *n*, provides the best performance for this strategy. A random arrival pattern provides performance in between the single alternating and group patterns.

- Strategy 2 is not 100 % fair but provides good performance which is independent of the arrival pattern.

- Strategy 3 is not 100 % fair but provides good performance which is independent of the arrival pattern.

- The Unisex Bathroom Problem is an excellent teaching tool for undergraduate and graduate courses in concurrent and parallel programming.

Acknowledgments

I would like to thank Dr. Jailan Zalmai and Dr. Andrew Poe for their advice and suggestions.

References

[1] Gregory R. Andrews, *Foundations of Multithreaded, Parallel, and Distributed Processing*, Addison-Wesley, 2000.

[2] M. L. Liu, *Distributed Computing, Principles and Applications*, Addison-Wesley, 2004.

[3] Scott Oaks, Henry Wong, *JAVA Threads*, O'Reilly, 1999.

[4] Grama, Gupta, Karypis, Kumar, *Introduction to Parallel Computing*, Addison-Wesley, 2003.

[5] J. Beveridge, R. Wiener, *Multithreading Applications in Win32*, Addison-Wesley, 1997.

[6] C. Hughes, T. Hughes, *Parallel and Distributed Programming Using C++*, Addison-Wesley, 2004.

[7] J. Gray, *Interprocess Communications in Unix*, Prentice Hall, 1998.

[8] W. Gay, *Linux Programming*, Sams, 1999.

[9] David Powers, *Dining Philosophers: Strategies for Success*, International Conference on Parallel and Distributed Processing (PDP05), 2005

[10] Gregory R. Andrews, *Concurrent Programming, Principles and Practice*, Addison-Wesley, 1991.