05/15/2024

Report: Multi-Thread Ludo

Operating System Project

Ahmad Abdullah (I22-1609, CY-D)
Talha bin Obaid (I22-1577, CY-D)
Abdul Sami Qasim (I22-1725, CY-D)

Contents

Introduction

The project we were given was to make a popular game known as LUDO using Operating System practices such as multi-threading, semaphores etc. The game would have a maximum of 4 players with a maximum of 4 tokens(goti).

Work Distribution

Making Grid

Among the three of our group members, Abdul Sami Qasim was tasked to make the grid and implement semaphores into the existing function that required blocking and enabling through sephamores.

Creating Ludo

Ahmad and Talha continued from where Abdul Sami left the grid of LUDO and made the game playable using multithreading but without semaphores which Abdul Sami was tasked to implement.

Implementing OS Concepts

After making LUDO playable, Abdul Sami used his knowledge along with the rest of the team member's cooperation and made the game use semaphores and multithreading in a way required in the project.

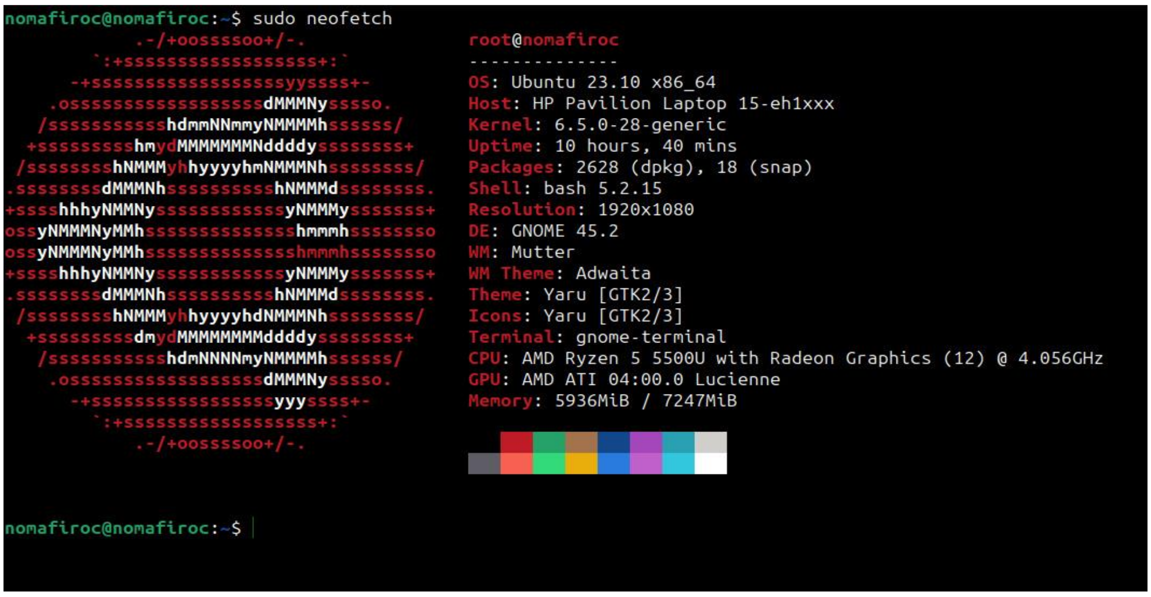
Documentation

Ahmad Abdullah made the document while Talha bin Obaid and Abdul Sami made pseudo codes for all the phases.

Machine Specification

Abdul Sami

6 Cores & 12 threads in CPU.

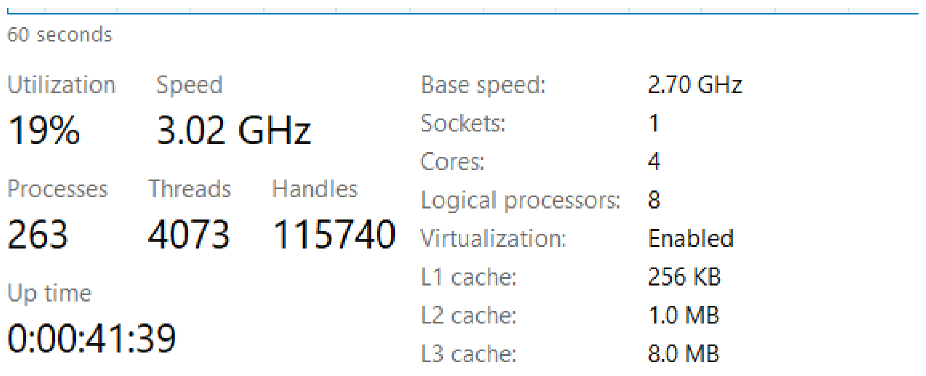


```
nomafiroc@nomafiroc:~$ sudo neofetch
            .-/+oossssoo+/-.               root@nomafiroc
        `:+ssssssssssssssssss+:`           --------------
      -+ssssssssssssssssssyyssss+-         OS: Ubuntu 23.10 x86_64
    .ossssssssssssssssssdMMMNysssso.       Host: HP Pavilion Laptop 15-eh1xxx
   /ssssssssssshdmmNNmmyNMMMMhssssss/      Kernel: 6.5.0-28-generic
  +ssssssssshmydMMMMMMMNddddyssssssss+     Uptime: 10 hours, 40 mins
 /sssssssshNMMMyhhyyyyhmNMMMNhssssssss/    Packages: 2628 (dpkg), 18 (snap)
.ssssssssdMMMNhsssssssssshNMMMdssssssss.   Shell: bash 5.2.15
+sssshhhyNMMNyssssssssssssyNMMMysssssss+   Resolution: 1920x1080
ossyNMMMNyMMhsssssssssssssshmmmhssssssso   DE: GNOME 45.2
ossyNMMMNyMMhsssssssssssssshmmmhssssssso   WM: Mutter
+sssshhhyNMMNyssssssssssssyNMMMysssssss+   WM Theme: Adwaita
.ssssssssdMMMNhsssssssssshNMMMdssssssss.   Theme: Yaru [GTK2/3]
 /sssssssshNMMMyhhyyyyhdNMMMNhssssssss/    Icons: Yaru [GTK2/3]
  +sssssssssdmydMMMMMMMMddddyssssssss+     Terminal: gnome-terminal
   /ssssssssssshdmNNNNmyNMMMMhssssss/      CPU: AMD Ryzen 5 5500U with Radeon Graphics (12) @ 4.056GHz
    .ossssssssssssssssssdMMMNysssso.       GPU: AMD ATI 04:00.0 Lucienne
      -+sssssssssssssssssyyyssss+-         Memory: 5936MiB / 7247MiB
        `:+ssssssssssssssssss+:`
            .-/+oossssoo+/-.


nomafiroc@nomafiroc:~$
```

Ahmad Abdullah

16GB RAM

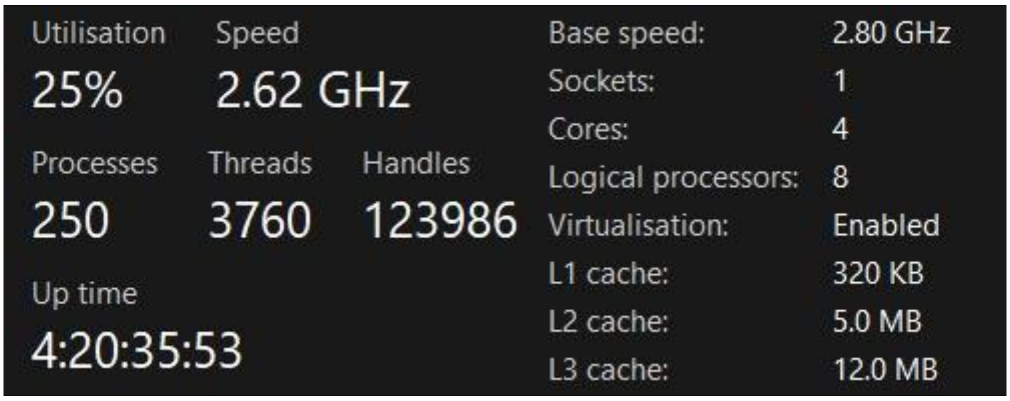


60 seconds

Utilization	Speed		Base speed:	2.70 GHz
19%	3.02 GHz		Sockets:	1
			Cores:	4
Processes	Threads	Handles	Logical processors:	8
263	4073	115740	Virtualization:	Enabled
Up time			L1 cache:	256 KB
0:00:41:39			L2 cache:	1.0 MB
			L3 cache:	8.0 MB

Talha bin Obaid

16GB RAM



Utilisation	Speed		Base speed:	2.80 GHz
25%	2.62 GHz		Sockets:	1
			Cores:	4
Processes	Threads	Handles	Logical processors:	8
250	3760	123986	Virtualisation:	Enabled
Up time			L1 cache:	320 KB
4:20:35:53			L2 cache:	5.0 MB
			L3 cache:	12.0 MB

Phase-I

Phase I required to implement and output a full grid with token and players shown wit the LUDO board.

Pseudo Code

```
FUNCTION draw_LudoGrid()
  // Loop through each cell in the grid
  FOR each row from 0 to 14
    FOR each column from 0 to 14
      // Check the number of tokens
      IF there are 4 token
        CALL makegrid(row, column)
      // Check the number of tokens
      ELSE IF there are 3 tokens
        CALL makegrid(row, column)
      // Check the number of tokens
      ELSE IF there are 2 tokens
        CALL makegrid(row, column)
      // Check the number of tokens
      ELSE IF there is 1 tokens
        CALL makegrid(row, column)
      END IF
    NEXT column
    PRINT nextLine
  NEXT row
END FUNCTION


FUNCTION makegrid(row, column)


  // Check for various positions and prints corresponding symbols
  IF the position is (4, 1) and the first token of player 1 is not open and there is at least 1 token
    PRINT "&"
  ELSE IF the position is (4, 4) and the second token of player 1 is not open and there are at least
2 token
    PRINT "&"
  //... Continue for all other positions and symbols
  ELSE
    PRINT " "
  END IF
END FUNCTION



// PLAYER THREAD FUNCTION


FUNCTION playerthread()
  temp = attr as integer
  tempPlayer = get player based on temp value
  tempsym = get symbol based on temp value


  SEMAPHORE() TO BLOCK OTHER PLAYER WHEN EXECUTING ONE
  INITIALIZE dice value for all the players to 0
  threesix = true
  FOR each dice roll
    ASSIGN dice value
    PRINT dice value
    SLEEP FOR 1 second
    IF dice value is not 6
      INCREMENT tempPlayer's withoutsixturns
    END IF


    IF dice value is 6
      SET tempPlayer's withoutsixturns to 0
      BREAK from loop
    END IF

    IF dice value is not 6
      SET threesix to false
      BREAK from loop
    END IF
  END FOR

  IF threesix
    ASSIGN zero on 3 consective sixes
  END IF

  FOR each dice roll and token
    IF token is open or dice value is 6
      GET user input for token to move
      WHILE user input is invalid
        GET user input for token to move
      END WHILE

      IF token value is 56
        PRINT "Token reached the end! Player has won!"
        BREAK from loop
      END IF

      WHILE token is not open and dice value is not 6 or token is win
        GET user input for token to move
      END WHILE

      IF token is open and notmoveflag is false
        CALL pathway function
      END IF
```

```
      IF dice value is 6
        ASSIGN token value to 0
        ASSIGN token open to 1
        ASSIGN token position based on symbol
      END IF

      IF token value is 56
        PRINT "Token reached the end! Player has won!"
        BREAK from loop
      END IF

      IF token value is safe point
        ASSIGN token stop to 1
      ELSE
        ASSIGN token stop to 0
      END IF
    END IF
  END FOR

  PRINT token values
  INITIALIZE semaphore
  CREATE hitRatio thread
  JOIN hitRatio thread
  CALL draw_frame function
  Using SEMAPHORE to Signal other player thread to start executing
  EXIT thread
END FUNCTION
```

Phase-II

Phase II asked to show the Hit Rate of a player, Cancelled Threads if no 6 is rolled on dice for too long and Winner of the game.

Pseudo Code

```
FUNCTION hitRatio()
  temp = attr as integer
  tempPlayer = get player based on temp value

  IF tempPlayer is in game and has not won
    WAIT for semaphore
    FOR each token of tempPlayer
      IF token symbol is '&' (player 1)
        FOR each token of other players
          IF token positions match and other player's token is open and not stopped
            UPDATE other player's token status and position
            INCREMENT tempPlayer's hit rate
      ELSE IF token symbol is '%' (player 2)
        FOR each token of other players
          IF token positions match and other player's token is open and not stopped
            UPDATE other player's token status and position
            INCREMENT tempPlayer's hit rate
      ELSE IF token symbol is '#' (player 3)
        FOR each token of other players
          IF token positions match and other player's token is open and not stopped
            UPDATE other player's token status and position
            INCREMENT tempPlayer's hit rate
      ELSE IF token symbol is '@' (player 4)
        FOR each token of other players
          IF token positions match and other player's token is open and not stopped
            UPDATE other player's token status and position
            INCREMENT tempPlayer's hit rate
        SIGNAL semaphore
    END IF
    EXITING thread
END FUNCTION



// MASTER THREAD

FUNCTION mthread()
    temp = attr as integer
    tempPlayer = get player based on temp value
    WAIT for semaphore2
    FOR each token
        IF token value is 56
            ASSIGN token win to 1
        END IF
        IF hitRate is greater than 0 and token value is greater than or equal to 50
            ASSIGN token home to 1
        ELSE
            ASSIGN token home to 0
        END IF
    END FOR

    IF tempPlayer's withoutsixturns is greater than or equal to 20
        ASSIGN tempPlayer's inGame to 0
    END IF

    notwinflag = 0
    FOR each token
        IF token is not win
```

```
      ASSIGN notwinflag to 1
    END IF
  END FOR
  IF notwinflag is 0
    ASSIGN tempPlayer's is_win to 1
  END IF
  SIGNAL semaphore2
  EXIT thread
END FUNCTION
```

Main Function

```
FUNCTION main()
  //asking about the number of token
  DO
    GET user input for number of Tokens for each player
  WHILE number of Tokens for each player is less than 1 or greater than 4

  //creating four players
  CREATE player with symbol '&' and number of Tokens
  CREATE player with symbol '%' and number of Tokens
  CREATE player with symbol '#' and number of Tokens
  CREATE player with symbol '@' and number of Tokens

  p1 = temp
  p2 = temp1
  p3 = temp2
  p4 = temp3

  //creating an array for random turns
  INITIALIZE array with values 1, 2, 3, 4

  WHILE true
```

```
    INITIALIZE semaphore with value 1


    //random_shuffle(arr,arr+4); //random values for different player selection
    GET user input for any key to Continue


    //creating 4 threads each for one player
    FOR each value in array
      CREATE thread with playerthread function and value
    END FOR


    FOR each value in array
      JOIN thread
    END FOR


  END WHILE


  RETURN 0
END FUNCTION
```

Other Projects

According to our team, these concepts could well be implemented in the Airline Control system if not implemented already where admins and control tower personnel can monitor and allow only chosen aeroplanes to land at a time. The semaphores and multi-threading in the Operating System Course are well aligned with the needs of the Airline Control System.