2020 OS Project 1 Report

1. Design

Inter-process communication

To emulate the scheduling, I used two cores in the VM, one is for main.c and scheduler and the other is for running child process. In the lecture, we learnt that there are two main categories of synchronization tools: message passing and shared memory. In this project, I choose to use pipe (a message passing tool) to synchronize the scheduler and the child process by letting the child process on the read pipe to wait for the "run" message wrote by the scheduler.

Self-defined system call

In this project, I added a system call named print_time which uses copy_from_user to get the buffer to store the process's start and finished time and use printk to print it out in dmseg. The way I added system call is the same as the one demonstrated in HW1.

Program

There are three main parts in my design: main.c (read inputs and pick a scheduler), process.c (create, remove, and resume child process), and program for each scheduling policy. In the following sections, I'll briefly introduce their functions.

1. main.c

This program first read the input from test set (create a list of process, a structure defined in process.h, simultaneously) and then quick sort the processes by their ready times so that it's easier for scheduler to process. At the end of this program, it sends the processes to the specified scheduler which will be explained later respectively.

2. process.c and process.h

process.h defines the structure process of which members are name, ready time, execution time, and two file descriptors for pipe.
In process.c, there are five functions defined:

1. TIME_UNIT: Function to define a time unit.
2. assign_core: Assign process to specified core.
3. proc_create: Create a child process and use self-defined system call to its start (first time to put on the core) and finished time.
4. proc_remove: Set the priority of the process to the lowest, i.e. remove it out of the core.
5. proc_exec: Set the priority of the process to the highest, i.e. execute the process.

3. scheduler.h and shecduler_X.c for each scheduling policy

In the following four schedulers, pipe is used to synchronize them and the child process so that they can move one time unit forward simultaneously.

1. FIFO: Straightforward implementation.
2. RR: Straightforward implementation.
3. SJF: Before each process is executed, find which ready process has the shortest deadline (this function takes linear time and the number of the processes in test data is less than ten, so it wouldn't affect the accuracy of the scheduling emulation too much).
4. PSJF: In each time unit iteration, find which ready process has the shortest deadline.

2. Kernel version

Linux version - 4.14.25

3. Analysis

Comparing the expexted and experimental results

Since I only output the start and finish time of each process but the start time of the scheduler, I offset the expected results by the start time of P1 if necessary.

1. FIFO

Input	Expected result	pid	Experimental result
P1 0 500 P2 0 500 P3 0 500 P4 0 500 P5 0 500	P1, start at 0 P1, end at 500 P2, end at 1000 P3, end at 1500 P4, end at 2000 P5, end at 2500	P1 2316 P2 2317 P3 2318 P4 2319 P5 2320	2316 finish at 503.282752366 unit 2317 finish at 1002.399053239 unit 2318 finish at 1503.988571509 unit 2319 finish at 2013.437254728 unit 2320 finish at 2515.917291692 unit
P1 0 80000 P2 100 5000 P3 200 1000 P4 300 1000	P1, end at 80000 P2, end at 85000 P3, end at 86000 P4, end at 87000	P1 2327 P2 2520 P3 2523 P4 2524	2327 finish at 81898.959303531 unit 2520 finish at 86840.443207841 unit 2523 finish at 87834.388585982 unit 2524 finish at 88826.120082406 unit
P1 0 8000 P2 200 5000 P3 300 3000 P4 400 1000 P5 500 1000 P6 500 1000 P7 600 4000	P1, start at 0 P1, end at 8000 P2, end at 13000 P3, end at 16000 P4, end at 17000 P5, end at 18000 P6, end at 19000 P7, end at 23000	P1 2531 P2 2532 P3 2533 P4 2534 P5 2535 P6 2536 P7 2537	2531 finish at 7959.398762500 unit 2532 finish at 12921.552577077 unit 2533 finish at 15887.777253294 unit 2534 finish at 16876.073014647 unit 2535 finish at 17865.788790011 unit 2536 finish at 18857.473411435 unit 2537 finish at 22817.748867781 unit
P1 0 2000 P2 500 500 P3 500 200 P4 1500 500	P1, start at 0 P1, end at 2000 P2, end at 2500 P3, end at 2700 P4, end at 3200	P1 2544 P2 2545 P3 2546 P4 2547	2544 finish at 1985.013048690 unit 2545 finish at 2484.233270318 unit 2546 finish at 2681.160864046 unit 2547 finish at 3176.619693045 unit
P1 0 8000 P2 200 5000 P3 200 3000 P4 400 1000 P5 400 1000 P6 600 1000 P7 600 4000	P1, start at 0 P1, end at 8000 P2, end at 13000 P3, end at 16000 P4, end at 17000 P5, end at 18000 P6, end at 19000 P7, end at 23000	P1 2554 P2 2555 P3 2558 P4 2559 P5 2560 P6 2561 P7 2562	2554 finish at 7928.014949383 unit 2555 finish at 12878.532634534 unit 2558 finish at 15848.875542414 unit 2559 finish at 16836.381177891 unit 2560 finish at 17825.314517244 unit 2561 finish at 18811.474190796 unit 2562 finish at 22769.777391619 unit

2. RR

Input	Expected result	pid	Experimental result
P1 0 500 P2 0 500 P3 0 500 P4 0 500 P5 0 500	P1, start at 0 P1, end at 500 P2, end at 1000 P3, end at 1500 P4, end at 2000 P5, end at 2500	P1 2570 P2 2571 P3 2572 P4 2573 P5 2574	2570 finish at 499.399324767 unit 2571 finish at 996.422478252 unit 2572 finish at 1492.670786968 unit 2573 finish at 1991.853671641 unit 2574 finish at 2489.145860870 unit
P1 600 4000 P2 800 5000	P1, start at 600 P1, end at 8100 P2, end at 9600	P1 2581 P2 2582	2581 finish at 8041.234578645 unit 2582 finish at 9526.188913351 unit
P1 1200 5000 P2 2400 4000 P3 3600 3000 P4 4800 7000 P5 5200 6000 P6 5800 5000	P1, start at 1200 P3, end at 18200 P1, end at 20200 P2, end at 20700 P6, end at 28200 P5, end at 30200 P4, end at 31200	P3 2591 P1 2589 P2 2590 P6 2594 P5 2593 P4 2592	2591 finish at 18072.788579962 unit 2589 finish at 20050.821881147 unit 2590 finish at 20547.038474297 unit 2594 finish at 27990.716336350 unit 2593 finish at 29974.141109779 unit 2592 finish at 30965.635360311 unit
P1 0 8000 P2 200 5000 P3 300 3000 P4 400 1000 P5 500 1000 P6 500 1000 P7 600 4000	P1, start at 0 P4, end at 5500 P5, end at 6000 P6, end at 6500 P3, end at 14500 P7, end at 18000 P2, end at 20000 P1, end at 23000	P4 2606 P5 2607 P6 2608 P3 2605 P7 2609 P2 2604 P1 2603	2606 finish at 5473.985449582 unit 2607 finish at 5969.294919873 unit 2608 finish at 6466.622046497 unit 2605 finish at 14389.646202841 unit 2609 finish at 17856.162652670 unit 2604 finish at 19837.823618539 unit 2603 finish at 22806.016089926 unit
P1 0 8000 P2 200 5000 P3 200 3000 P4 400 1000 P5 400 1000 P6 600 1000 P7 600 4000	P1, start at 0 P4, end at 5500 P5, end at 6000 P6, end at 6500 P3, end at 14500 P7, end at 18000 P2, end at 20000 P1, end at 23000	P4 2621 P5 2622 P6 2623 P3 2620 P7 2624 P2 2619 P1 2618	2621 finish at 5471.835149493 unit 2622 finish at 5969.047008202 unit 2623 finish at 6465.778728124 unit 2620 finish at 14389.264190514 unit 2624 finish at 17845.826278100 unit 2619 finish at 19827.493013289 unit 2618 finish at 22798.082913874 unit

3. SJF

Input	Expected result	pid	Experimental result
P1 0 7000 P2 0 2000 P3 100 1000 P4 200 4000	P1, start at 0 P2, end at 2000 P3, end at 3000 P4, end at 7000 P1, end at 14000	P2 2632 P3 2633 P4 2634 P1 2635	2632 finish at 2000.966909504 unit 2633 finish at 2996.665483856 unit 2634 finish at 6991.703971602 unit 2635 finish at 13984.699768048 unit
P1 100 100 P2 100 4000 P3 200 200 P4 200 4000 P5 200 7000	P1, start at 100 P1, end at 200 P3, end at 400 P2, end at 4400 P4, end at 8400 P5, end at 15400	P1 2642 P3 2643 P2 2644 P4 2647 P5 2648	2642 finish at 199.299486531 unit 2643 finish at 400.422975839 unit 2644 finish at 4399.441530006 unit 2647 finish at 8391.760592134 unit 2648 finish at 15369.319228631 unit
P1 100 3000 P2 100 5000 P3 100 7000 P4 200 10 P5 200 10 P6 300 4000 P7 400 4000 P8 500 9000	P1, start at 100 P1, end at 3100 P4, end at 3110 P5, end at 3120 P6, end at 7120 P7, end at 11120 P2, end at 16120 P3, end at 23120 P8, end at 32120	P1 2655 P4 2656 P5 2657 P6 2658 P7 2659 P2 2660 P3 2662 P8 2663	2655 finish at 3099.641189828 unit 2656 finish at 3109.940691088 unit 2657 finish at 3120.557825115 unit 2658 finish at 7133.302187077 unit 2659 finish at 11117.282168966 unit 2660 finish at 16093.986321801 unit 2662 finish at 23068.053866193 unit 2663 finish at 32027.047578153 unit
P1 0 3000 P2 1000 1000 P3 2000 4000 P4 5000 2000 P5 7000 1000	P1, start at 0 P1, end at 3000 P2, end at 4000 P3, end at 8000 P5, end at 9000 P4, end at 11000	P1 2673 P2 2674 P3 2675 P5 2676 P4 2677	2673 finish at 3012.160586755 unit 2674 finish at 4014.935909898 unit 2675 finish at 8020.436399801 unit 2676 finish at 9015.272044960 unit 2677 finish at 11007.480002382 unit
P1 0 2000 P2 500 500 P3 1000 500 P4 1500 500	P1, start at 0 P1, end at 2000 P2, end at 2500 P3, end at 3000 P4, end at 3500	P1 2684 P2 2685 P3 2686 P4 2687	2684 finish at 2013.869116690 unit 2685 finish at 2514.639511993 unit 2686 finish at 3015.777645215 unit 2687 finish at 3518.049154806 unit

4. PSJF

Input	Expected result	pid	Experimental result
P1 0 10000 P2 1000 7000 P3 2000 5000 P4 3000 3000	P1, start at 0 P4, end at 6000 P3, end at 10000 P2, end at 16000 P1, end at 25000	P4 2698 P3 2697 P2 2696 P1 2695	2698 finish at 5929.009887700 unit 2697 finish at 9859.070975629 unit 2696 finish at 15773.930600383 unit 2695 finish at 24664.343258176 unit
P1 0 3000 P2 1000 1000 P3 2000 4000 P4 5000 2000 P5 7000 1000	P1, start at 0 P2, end at 2000 P1, end at 4000 P4, end at 7000 P5, end at 8000 P3, end at 11000	P2 2708 P1 2707 P4 2710 P5 2711 P3 2709	2708 finish at 1977.224735132 unit 2707 finish at 3955.858869087 unit 2710 finish at 6932.726048582 unit 2711 finish at 7916.495369303 unit 2709 finish at 10866.972956115 unit
P1 0 2000 P2 500 500 P3 1000 500 P4 1500 500	P1, start at 0 P2, end at 1000 P3, end at 1500 P4, end at 2000 P1, end at 3500	P2 2719 P3 2720 P4 2721 P1 2718	2719 finish at 990.191620942 unit 2720 finish at 1483.654415908 unit 2721 finish at 1982.249382018 unit 2718 finish at 3463.844982760 unit
P1 0 7000 P2 0 2000 P3 100 1000 P4 200 4000	P1, start at 0 P3, end at 1100 P2, end at 3000 P4, end at 7000 P1, end at 14000	P3 2729 P2 2728 P4 2730 P1 2731	2729 finish at 1092.900339153 unit 2728 finish at 2975.963598021 unit 2730 finish at 6924.427843482 unit 2731 finish at 13804.468961330 unit
P1 100 100 P2 100 4000 P3 200 200 P4 200 4000 P5 200 7000	P1, start at 100 P1, end at 200 P3, end at 400 P2, end at 4400 P4, end at 8400 P5, end at 15400	P1 2738 P3 2739 P2 2740 P4 2743 P5 2744	2738 finish at 198.584776783 unit 2739 finish at 397.179763386 unit 2740 finish at 4354.907814485 unit 2743 finish at 8304.726012720 unit 2744 finish at 16083.285978037 unit

Possible reasons for the discrepancy

Clearly, the error of all the experimental results are pretty small (<5%).

1. Workload from the host:
   Since VM is in the user space, it needs to compete with the host's applications to get the control of the CPU. I have run the emulation when I used the host to do other works, and the error of that emualtion is way larger than the one of the results I shown above (I closed all the applications in my host to run te that).

2. Overhead in pipe communication:
   It takes time to use PIPE to do inter-process communication since to read from or write to the pipe is like an I/O operation. This effect is not negligible.

3. Variation of CPU clock rate:
   The CPU may alter its clock rate from time to time (depends on the workload and other factors), so if the clock rate is not the same through the whole emulation, we'll have systematic error.