INSTITUT SUPERIEUR INFORMATIQUE ISI
المعهد العالي للإعلامية

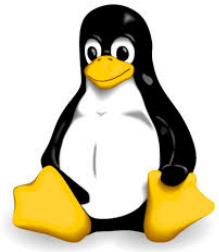
User Guide

Multi-task Scheduler

The Sudoers Group4 - 1ING3

Presented by:

Firas Kahlaoui - Scrum Master
Oussema Abdelmoumen - Developer
Hamza Bargoug - Developer
Seifeddine Ben Fredj - Developer
Malek Nasri - Developer



December 2025

Table of Contents

1 Introduction

This project implements a multi-task process scheduler on Linux, capable of managing multiple scheduling policies. It simulates process execution based on arrival time, burst time, and priority.

2 Features

- **Scheduling Policies**:
  - FIFO (First In First Out)
  - Round-Robin (configurable quantum)
  - Preemptive Static Priority
  - Multi-level Queues (Skeleton)
- **Interactive Mode**:
  - Dynamic menu to select policies at runtime.
  - Option to enable/disable graphical visualization.
- **Advanced Visualization**:
  - Real-time dashboard using `ncurses`.
  - Visual Process Table with progress bars.
  - CPU State indicator (IDLE/BUSY).
  - Visual Ready Queue.
  - Scrolling Gantt Chart timeline.

3 Prerequisites

This project is designed for Linux environments. Ensure you have the following tools installed:

- **GCC** (GNU Compiler Collection)
- **Make** (Build automation tool)
- **libncurses-dev** (Library for the graphical interface)

3.1 Installing Prerequisites

On Ubuntu/Debian:

```
sudo apt-get update
sudo apt-get install build-essential libncurses5-dev libncursesw5-dev
```

4 Installation & Build

4.1 Clone the Repository

```
git clone https://github.com/FirasKahlaoui/os-scheduling-system
cd os-scheduling-system
```

4.2 Compilation

To compile the project, run:

```
make
```

This will compile all source files and generate the executable in the `bin/` directory.

To clean the build (remove object files and executable):

```
make clean
```

4.3 Installation and Deployment

The project provides an automated installation procedure through the `Makefile`. The `install` target is designed to adapt to the user's system permissions.

- If the user has write access to `/usr/local/bin`, the executable is installed system-wide.
- If the user does **not** have administrative privileges, the installation automatically falls back to a local installation in `$HOME/.local/bin`.

This approach ensures that the software can be installed and executed without requiring root access, improving portability and usability across different Linux environments.

To install the scheduler, run:

```
make install
```

After installation, the scheduler can be executed from any directory using:

```
scheduler
```

If the local installation directory is not already included in the system `PATH`, the user may need to add the following line to their shell configuration file (e.g., `.bashrc` or `.zshrc`):

```
export PATH="$HOME/.local/bin:$PATH"
```

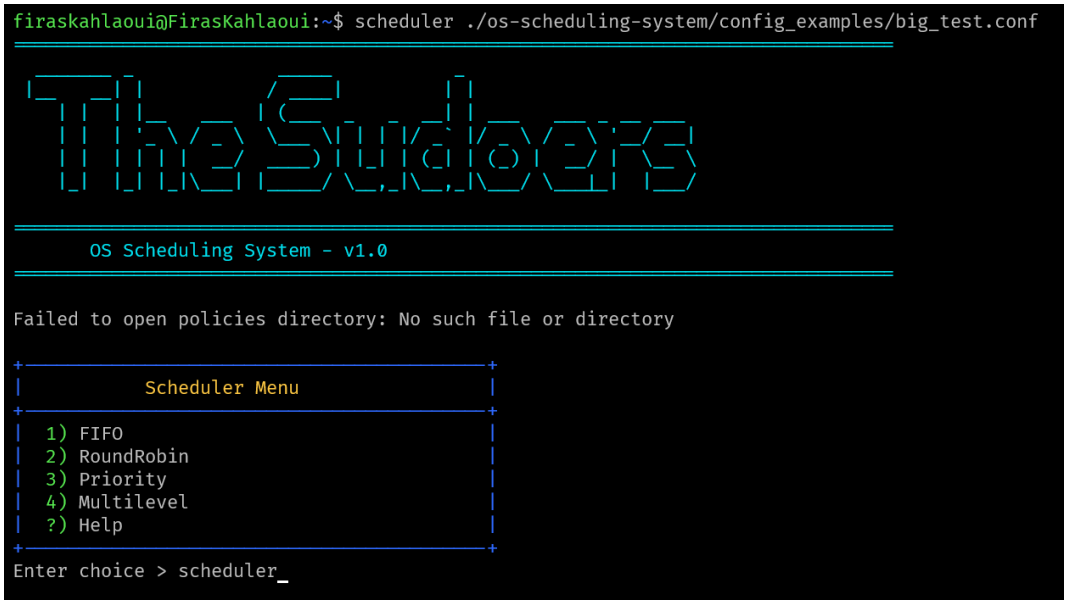


Figure 1: Application Launch Screen

5 Usage

5.1 Interactive Mode (Recommended)

Run the scheduler with a configuration file. You will be prompted to select a scheduling policy and enable visualization.

```
./bin/scheduler config_examples/test1.conf
```

5.2 Command Line Mode

You can specify the policy and quantum directly as arguments:

Syntax:

```
./bin/scheduler <config_file> <policy_name> [quantum]
```

Examples:

```
# Run FIFO
./bin/scheduler config_examples/test1.conf FIFO

# Run Round-Robin with quantum 4
./bin/scheduler config_examples/test2.conf RoundRobin 4

# Run Priority
./bin/scheduler config_examples/big_test.conf Priority
```

5.3 Help System

The application provides built-in help documentation for the various scheduling policies.

To view the general usage instructions, run the program without any arguments:

```
./bin/scheduler
```

Inside the interactive menu, you can access detailed information about each policy by selecting the help option or typing:

```
help <policy_name>
```

Supported help topics include:

- **fifo**: First-In-First-Out policy details.
- **roundrobin** (or **rr**): Round-Robin policy details.
- **priority**: Static Priority policy details.
- **multilevel**: Multilevel Queue policy details.

6 Configuration File Format

The configuration file defines the processes to be scheduled. Each line represents a process. You can use `#` for comments and include blank lines for better readability.

Format:

```
Name    Arrival_Time    Burst_Time    Priority
```

Example:

```
# Process definition: Name Arrival Burst Priority
P1      0               10            1

# High priority process
P2      2               5             2
P3      4               8             3
```

7 Graphical Interface

The application features a rich graphical interface using `ncurses` to visualize the scheduling process in real-time.

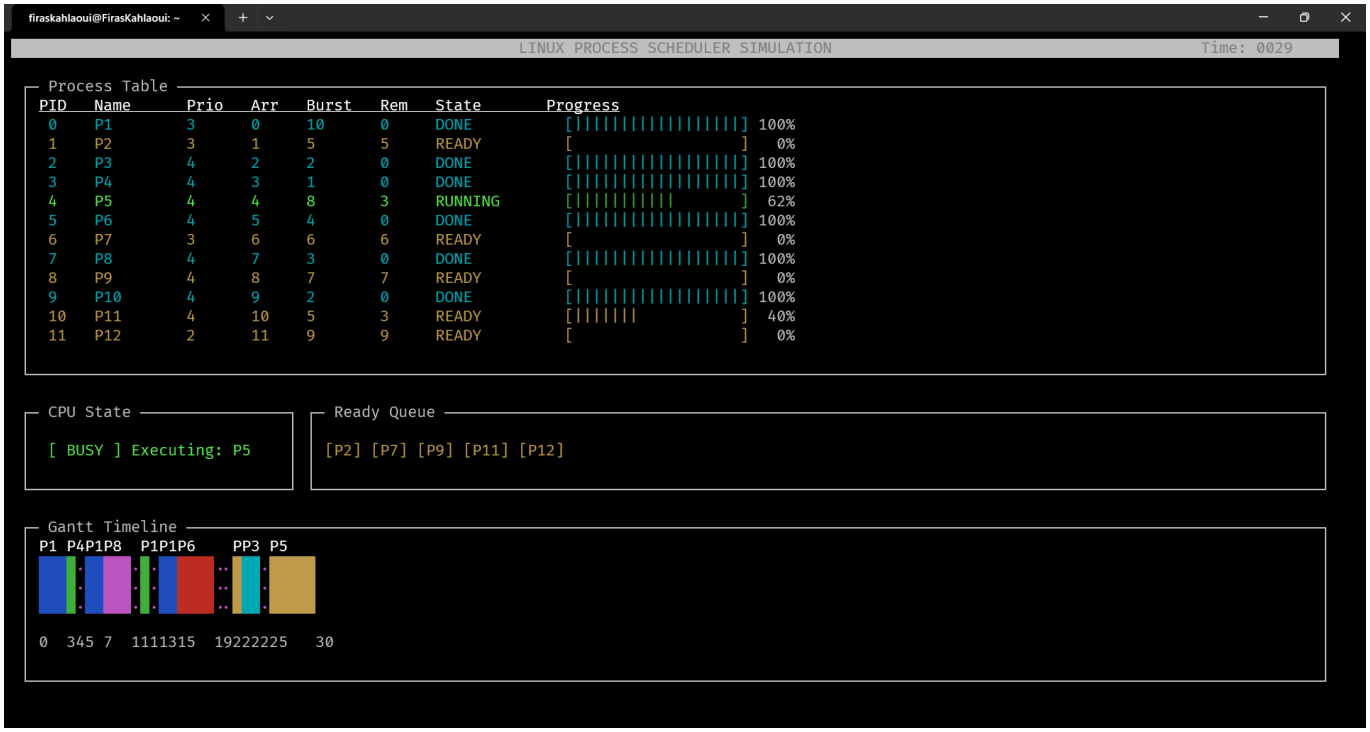


Figure 2: Real-time Graphical Dashboard