Real-Time System hw1

廖怡誠

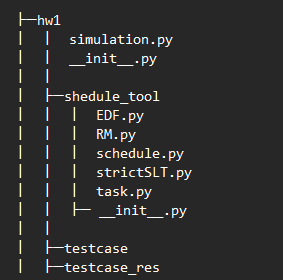
這次的作業是使用RM、EDF、strictLST排程機制，模擬實際的工作排程，以下將分成架構設計、遇到問題與心得三個部分。

1. 架構設計

圖1.為這次作業的樹狀架構圖，為了模擬實際的timer運行，首先我設計了simulation.py，這份程式負責的工作是將task.txt的所有task儲存起來，並以計數器的方式模擬timer，在對應的時間點，將task傳入到Ready queue當中等待執行。

接著我將RM、EDF、strictSLT三個排程機制相同的部分寫成一份shedule.py，並以繼承的方式延伸成三種不同的排程機制的程式碼，這樣的設計方式可以有效的減少程式碼重複性過高的問題，只要針對對應的優先度判斷即可。

而在Ready queue的設計上，需要使用記憶體不連續的資料結構使用，在Python中我使用List來實現此概念。



1. 遇到問題
   1. Context-Switch效能考量

在每個時間點下，都需要判斷Ready queue中是否有job的優先度比目前正在執行的job還高，而也有可能出現兩者優先度相同的情況。例如，在RM的排程機制下，正在執行的job與Ready queue中的job兩者period相同。在面對這樣的問題時，我選擇不進行交換job，因為我考量交換job所需要Context-Switch造成的時間成本，只是題目規定上不考慮此類消耗，但在實際中仍會面臨。

在這樣的改動之下，排程順序會與使用Tid較小優先度較大規則的排程順序略有不同，但經過比較發現，差別只在於兩種Tid相反，期間所missing的schedule數量與沒有job的時間點數量仍然會相同。

1. 心得

simulator

Context-Switch