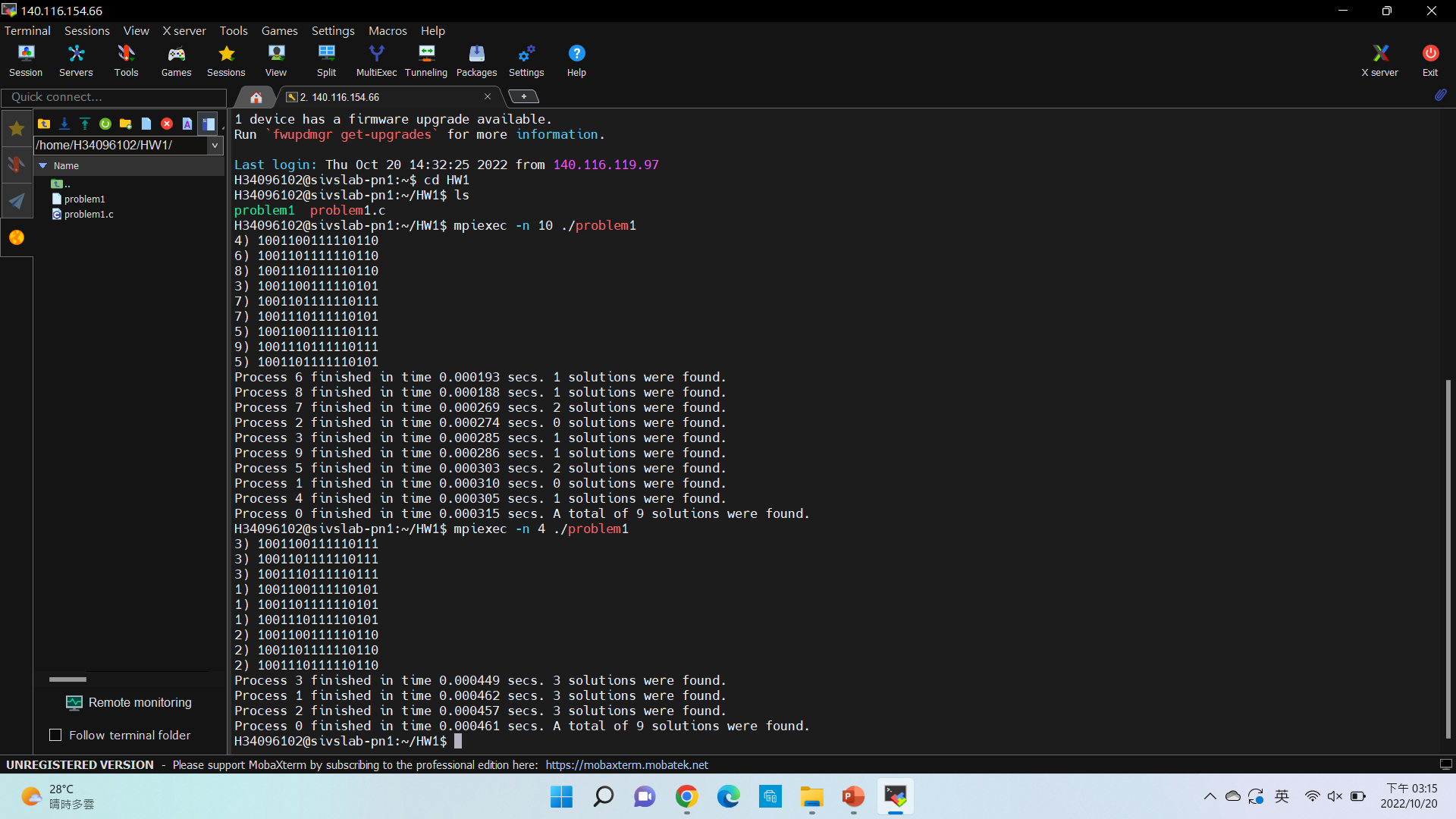
HW1 H34096102 謝欣蓉

**Problem 1**

這題因為大多數的程式碼已經有附上了所以比較難的是在算總合的地方利用tree的方式加總。在算每個process找到的數量只需要用題目給的程式碼並將迴圈的地方改成由process id開始且每隔總process數量跳著計算, 就可以將所有要計算的次數全部平分給每個process來分工完成, 這也代表越多台process分工每台跑的時間就越少。由下圖結果可以得知我使用10台process跑程式碼, 花費時間最長的一台只需0.000315秒, 當我換成4台process時則花費時間最短的一台就需要0.000449秒, 很明顯就可以看出差別。



在計算solution總和是用tree structure的方法, 一開始設一個變數”process”用來計算還沒被tree structure計算過count數目的process數量(也就是一開始等於總process數), 以及變數”offset”用作分辨是否要重新進入迴圈繼續send&receive message, 每次的迴圈都會處理掉一半的process, 當仍有多個process尚未處理則由offset來判斷進入下一次的迴圈。接著分為兩大類: 當process id小於(”process”/2)則做為接收方, 當process id小於”process”且不小於offset則作為傳遞方。不停地將大id傳給小id直到剩下一個便停止。這樣的做法就可以利用tree structure完成點對點傳遞。

這次的作業較困難的地方應該是思考樹狀傳遞的那塊, 花了不少時間在研究上課的簡報, 除了對指令不熟悉之外也要特別搜尋那些傳遞參數該放甚麼比較正確, 其餘的部分題目以及上課的時候就講得蠻清楚所以在多平行處理機的分配上比較不花時間。

**Problem 2**

這題的做法與上議題幾乎一樣, 不論是在使用tree structure來點對點傳送資料、加總數量, 或者用迴圈分配process跑程式分工合作都是使用相同的方法, 但這題我覺得比較有趣的是有多了一個可以分析的數據。從下方第一張圖可以很明顯的看出用10台process跑程式時間大約都需要花0.0003~0.0004秒, 但改成20台來跑之後僅需0.0001~0.0002秒即可完成, 這與上一題相同的分析結果。不過從第二張圖可以看到, 將toss的數量由1000增至10000000, 計算出來的pi值可以差很多且更精確, 當然, 因為要計算的次數倍增所以花費的時間也會增加很多。

