關於程式作業二的一些回饋

這個題目在實作上的時候，應該是比作業一要有挑戰性一些。我稍微整理一下大家的結果與觀察：

- 在工作量上（不論是用 expanded nodes 計算或時間計算），大部分同學發現 forward-checking 有最顯著的改善。部分同學可能在 BFS 本身就加入了類似 forward-checking 的功能，所以差別就不明顯了。
- 如同有同學在討論區問的，當一個 variable 的所有值都試過而失敗，是不需要在同一個地方去試其它 variables 的。這差別可能主要顯示在沒有 forward-checking 的時候。
- 有些同學有提到「把可以確定的格子先指定」的做法可以加速，其實這就是這題目中 MRV 的功能。這在這類問題（也就是 Boolean satisfiability 問題）是標準的一個 heuristic，有專門的名稱叫做 unit-propagation。
- 有興趣的同學可以去查一查 DPLL 演算法，這是這類型問題的標準演算法，和我們這個題目很像，可以用在像是數位電路驗證等的問題。
- 在大家的實驗結果中，degree heuristic 的改善程度就比較不一定了。LCV 大部分的同學覺得用處不大，甚至可能變慢（本身的運算量較大）。不過 LCV 通常都是用在選擇 variable 的 heuristic 之後，較不會單獨使用。（我們的 constraints 不是 binary constraints，所以有些著色問題看起來好瞭解又有效的做法，在這裡可能會比較難講，或受一些問題本身因素影響。）
- 在我給的測資中，很多人都有發現第三個特別難。這一個的確是特別設計來難程式的。但只有少數人有討論原因。
- 在隨機產生的問題中，因為 DFS 較受到運氣的影響，所以自己多隨機產生一些測資來分析（不限於給的測資），效能比較才比較可靠。