1.如果一個大問題可以拆分成多個小問題，則一個大問題的最佳解，可以拆成多個小問題的最佳解相加，此規則可以用反證法來證明，若該大問題的最佳解可以用許多小問題的最佳解合成(相加)，則小問題的最佳解也會是他自身的最佳解，如果不是，就會形成矛盾

2.greedy 快於 dynamic programming

Greedy：在每個步驟都會找到當前的最佳解，貪婪演算法與動態規劃的不同在於它對每個子問題都做出選擇，不能回退(後悔)。動態規劃可以儲存以前的運算結果，並根據以前的結果對當前進行選擇，有回退(後悔)功能。

舉例：霍夫曼編碼、最小生成樹...

Dynamic programming：將所有的小問題的最佳解合成大問題的最佳解，如果所要處理的最佳化問題無法找到一個選擇程序來逐一檢查，就適用於

dynamic programming

動態規劃過程是：每次選擇依賴於當前狀態，又隨即導致狀態的轉變。一個決策序列就是在變化的狀態中產生出來的。

舉例：矩陣乘法