About Fixed-Parameter Algorithm

Bang Ye Wu（吳邦一）, 2008/9

Fixed-Parameter Algorithm, or parameterized complexity, 是演算法研究中一個新興領域（嗯，才十多年還算新），其目的在突破以往僅以 data size 做爲衡量 problem/algorithm complexity 唯一參數的的限制，而允許在研究中以其他參數做爲評估複雜度的依據，其最大的發展方向在提供我們對付諸如 NP-hard 此類難題的一種方式，說的具體一點兒，NP-hard 的 problem 只是說此問題在 worst case 很難有快速的（poly(n)）演算法，但可不是所有的 Instance 都難解，在某些狀況下也沒那麼可怕。講的再具體一點兒，以 MaxClique 爲例，這問題在找一個圖中最大 size 的 clique，在最壞的情形下固然沒有好的演算法，但是如果對於不是很大的 k，要找有無 size 爲 k 的 clique 就不會太難，最天真（naive，這個字有時跟愚笨是同義詞，以後有人稱讚你天真活潑時請留意）的做法就是檢查所有 k 個點的 vertex subset，所花的時間不會超過 $O(2^k \times poly(n))$，這在 k 很大的時候當然很糟，但是在 fixed k（constant）的限制下，就是 tractable 了，簡單吧！

在這個概念下，其實一大票的 graph 問題都有一個 universal 的 fixed-parameter algorithm，就是上述那個最天真的解法，下面這個觀察簡單之至，理由是只要是屬於 NP 的問題可以在 poly(n)驗證答案。

Observation 1：Every graph problem in NP with a vertex/edge subset of size k as its solution can be solved in $O(2^k \times poly(n))$, and is therefore parameterized tractable.

當然如果這麼簡單就沒有故事可說了，對於一個形如 $O(2^k \times poly(n))$ 的時間複雜度，通常我們要努力找出更好的方法，而努力的對象通常是前半部指數的部分而比較不在意後面那個 poly(n)。（Why? 因爲指數的底是個可怕的怪獸），舉例來說吧，對於 VertexCover 這個問題就已經改進到 $O(kn + 1.274^k)$。此外，對於 spanning tree 或是其他類型的問題就沒有那麼直接了，這時候就必須設計一

些好的方法來 bound 住所檢查的 candidate 個數，其中一個方法是用 search tree。Search tree 是以往在設計 exact solution algorithm 的常用技巧，演變出來的 strategy 包括 divide and conquer, branch and bound, A*等等，只要能限制住在 search tree 上的 node 是某個參數 k 的函數下就成功了。

理論上參數可以是任意選取的（當然，應該是有意義的，不過這個當然有時候也不那麼當然），不過，最常見的參數就是 solution 的大小，另外像是以 treewidth 當參數的其實可以歸類於 special graph 上的 exact algorithm，這個領域也是當初碰到 NP-hard 問題時的研究方向之一。

Fixed-parameter algorithm 當然也有可以爭論之處。能夠處理各種 case 的演算法固然是最好，當最好的不可求時，我們希望能找到能夠處理我們「通常需要處理」（或是說有意義）的 case，問題就在這裡了，當問題一旦被抽象化之後，也就無法定義什麼是有意義的 case。其實有很多問題 NP-hard 得很沒有意義（這裡 NP-hard 當動詞用），例如說 clustering 的問題，資料交雜在一塊兒不能夠 clustering 的 case 通常是計算上的難處，但是這種 instance 根本不應該發生，做出來的結果也沒有可信度與重要性，應該回頭重新處理 data 本身的關聯定義。對於 clique 問題以 parameterized 的方式處理在某些應用上感覺是有意義的，因爲 clique size 並不大，但是 vertex cover 感覺上意義就不明顯，通常來說，vertex cover 的解的大小跟 vertex 總數 n 幾乎成正例（true for regular graphs or degree bounded graphs）。此外，例如像 maximum leaf spanning tree 問題，希望找到越多葉子的樹越好，一個 graph 的 solution 通常不太小，而且這類 maximization 的問題要找到越好的解就必須付出計算上以指數成長的更大代價，意義上似乎有點那個給他點點點。

Anyway，這是個有趣的研究方向，很多問題可以拿來嘗試看看。兩個有趣的疑問：一個是這方法用在 weighted case 似乎不大管用；第二個是這方法通常用在 maximization 問題，如果用在 minimization 問題大概就不能以 solution size 爲

參數了，否則越好的解需要越少的時間在道理上就不通（See remark below for a correction of this point），MAX LEAF 問題有一個兄弟叫做 connected domination set(CDS)，找到 k 個 leaf 就是找到 n-k 個點的 CDS，那麼，對於 MAX LEAF 的好的演算法卻不是 CDS 好的演算法，這在邏輯上好像不易解釋。

Remark（根據張貿翔教授的更正）： Fixed-parameter algorithm 是可以處理 minimization problems 的，其原因在於，Fixed-parameter algorithm 所處理的問題定義爲 Decision Problem，例如 Vertex Cover 定義如下

Vertex Cover

Input: A graph *G* and an integer *k*.

Output: Is there any vertex cover of size at most k?

以一個 Time Complexity 爲 $O(kn + 1.274^k)$的 algorithm 來說，你給他很小的 k，演算法在很短的時間會回答你「沒有大小不超過 k 的解」，因此並沒有越好的解需要的時間越短的現象。

Remark 2: 所以這裡還是有個有趣的現象，對於 MAX 問題，假如我們不要求找到最佳解，而是只要找到「夠好」的解，我們可以把 k 值從小往大 run 到某個值之後就停下來；但是對於 MIN 的問題就沒這個好處了，Algorithm 一但回答了 Yes，就已經是找到了 Optimal 了。