2014年12月2日 星期二

adtComp

Double Linked List:

- 一個 List 會有很多的 Node，每個 Node 指向他的前一個以及後一個 Node 已達成串聯的效果
- iterator 沒辦法做到 random access
- 有一個 dummy node 來將整個 list 頭尾串起來，也作爲提供「是否爲空」和「是否爲尾」的資訊來源
- 在新增或刪除 node 的時候基本上只要 handle 與該 node 相連的前後兩個 node 即可，所以我實作了一個 linkFromTo(node front, node back) 來處理相連兩個 node 的 linking
- 在 sorting 的部分，因爲不能 random access，所以我用最簡單的 bubble sort 來處理，我實作了一個 swapData(node front, node back) 來交換兩個 node 的 data，然後用 for loop 搭配 iterator 就可以辦到 loop through list 並且 sort，而且 bubble sort 本來就是將兩個相鄰的 data 做 swap，所以處理起來很方便

Dynamic Array:

- dynamic array 其實就是一般的 array，但是克服了一般 array 固定長度的限制，簡單來說就是在新增資料時視情況增長整個 array 的 capacity
- 缺點是 expand 的時候要 copy 一次資料，但是如果資料不是瘋狂的新增，那麼其實不會太常遇到要 expand 的情況
- iterator 相對與 dlist 來說好做許多，只要 handle _node 這個 pointer 就好，新增刪除資料也很方便，例如我在 pop_back 的時候其實沒有真的刪掉東西，只有改 size 而已
- 因爲可以 random access，所以 sorting 可以很快

Binary Search Tree:

- 我沒有做出來...我做出 successor & predecessor，但是在 erase & iterator 的地方沒有想清楚該怎麼做，時間不夠所以就 GG 了....

2014年12月2日 星期二

Performance Compare:

- 首先是 insert，我原本以爲 dynamic array 會比較快，但實際上 adta -r 100000 的時候，dlist 在 memory usage 和 time usage 上都比較好，memory usage 上我可以理解，因爲 dynamic array 可能會有 overhead，但是速度上就有點怪了，我本來以爲 list 要 handle next & prev 會比較慢，結果反而更快
- 爲了實驗 dynamic array 的 memory overhead，我用了 adta -r 262144，也就是 2^18，這種情況下 dynamic array 應該不會有 overhead，的確 memory usage 上是比 dlist 少的（少 0.1 m），而再多新增一個的時候，dlist 沒有變，但是 dynamic array 就直接翻倍了
- 接下來就是 sorting 了，毫無懸念的 dlist 應該超慢，因爲我是用最笨的 bubble list …果然在 1000 筆資料的時候，dynamic array 只用了 0.02 second，但是我的 dlist 卻用了 11 秒...