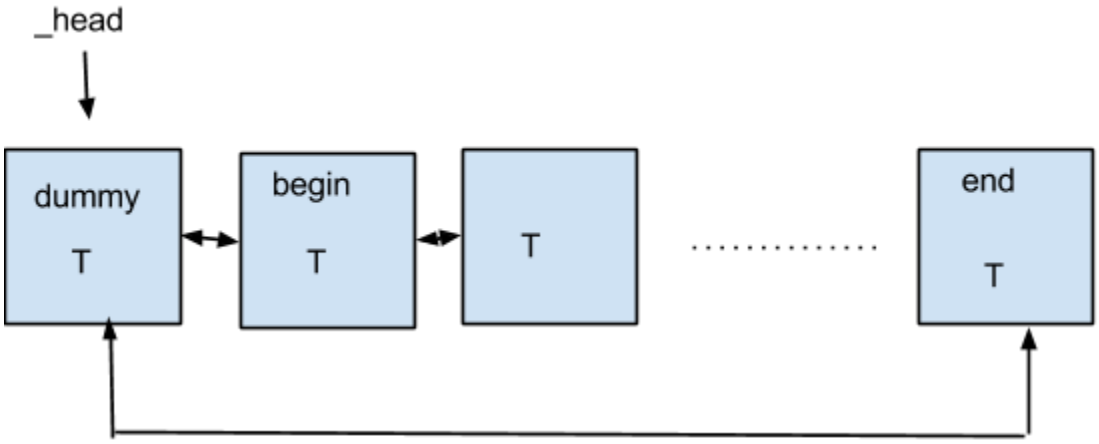
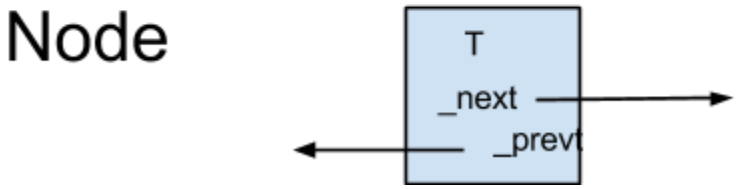
資料結構的實做

dynamic list

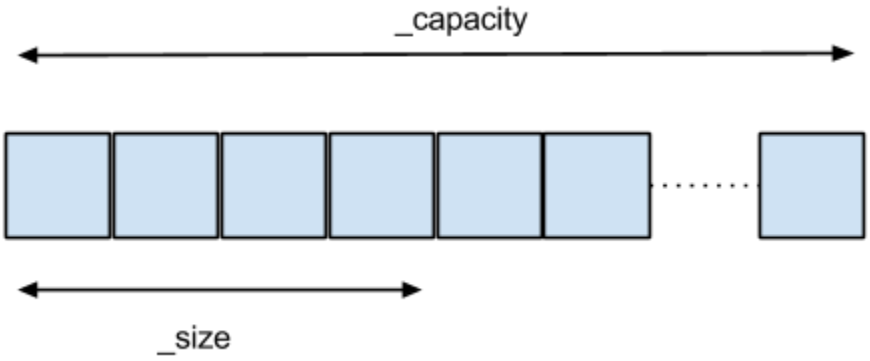
In dynamic list, there is a dummy block for connecting the begining node and the ending node. Each node has two pointers, _next and _prev, which are used for connecting nodes in dlists. Since each node has two pointers connecting the previous or next node, it allows us to travel bidirectional in dlist.

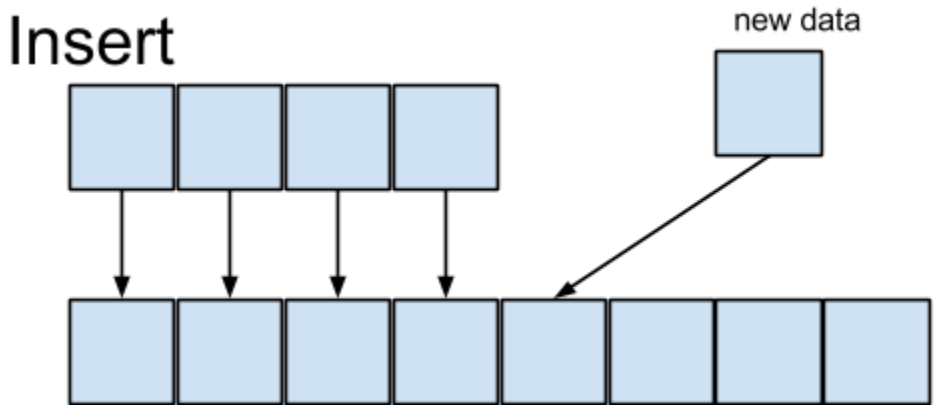




dynamic array

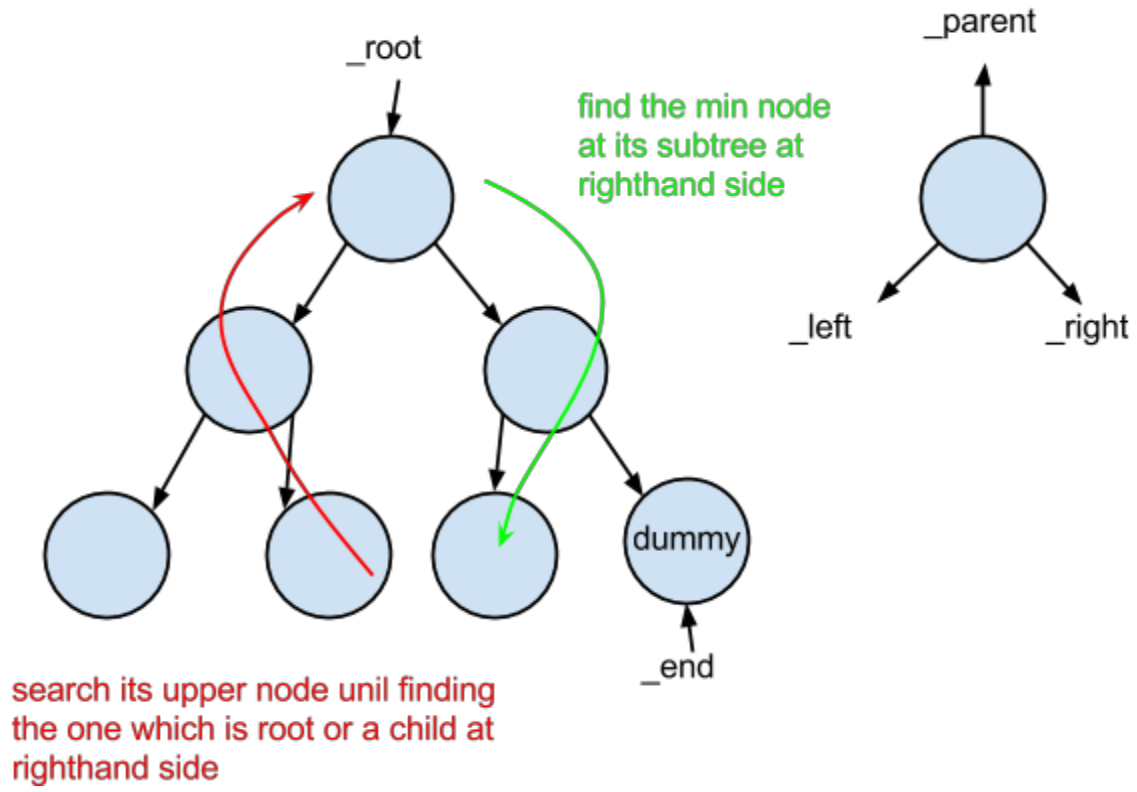
In dynamic array, I use _size to record how much data are stored and _capacity to record how much data one can store. When the array is full (_size = _capacity), a new array which doubles the size of the former one would be created. Then, the original data and new data would be copied into the new array. The reason why I choose to double the capacity of array instead of increasing it by a constant size is beacause the former method requires less times to reallocate the memory when one is going to store huge amount of data. In that case, the performance would be quite different.





binary search tree

For a BST node, nodes on the righthand side are equavilent or greater than itself, and smaller on the lefthand side. In my implementation, one node has three pointers connecting its two childs and its parent. Also, I have a dummy node which makes me implement iterator of the BST easier. In the figure, I shows the method to find next bigger node. Since each node has information about its parent, we do not need to perform a stack for trace storing inside the iterator. What a iterator needs to do is just use the provided method to find next node.



實驗比較,請說明你所設計的實驗以及比較這些 ADT 的 Performance

實驗設計

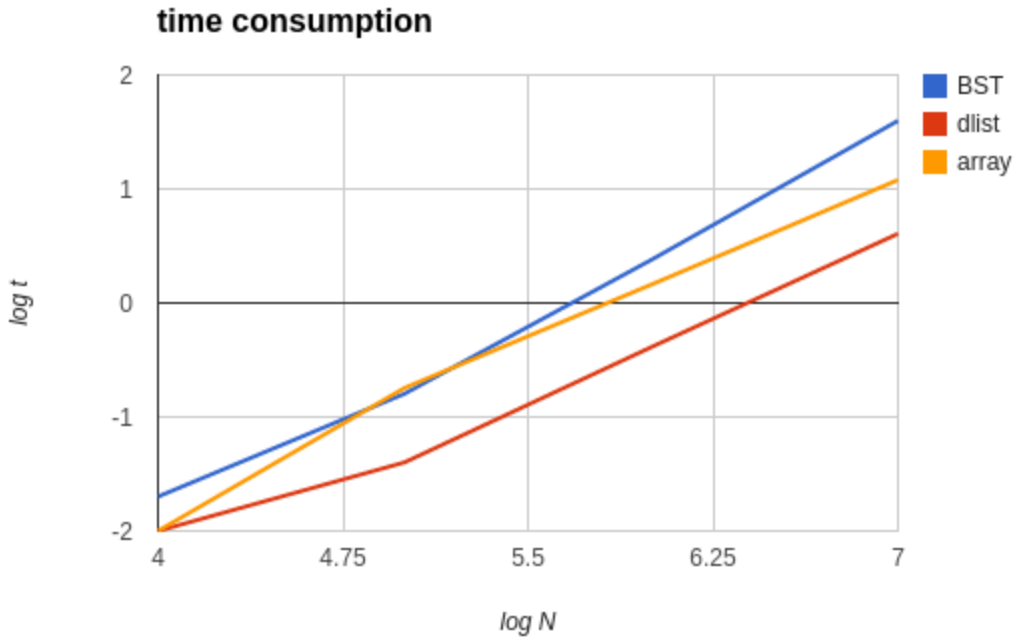
比較三種資料結構儲存資料的速度
dofile:
adtr 10
usage
adta -r {N}
usage
N為測試所使用的資料數目

實驗預期

dlist新增資料所需的時間應會正比於N,bst因為要排序時間複雜度應為大於N
而記憶體dlist只是用到兩個pointer消耗的記憶體應該會最少（array因未會allocate多餘的記憶體以備未來使用,故應多於dlist

結果比較與討論

	BST	dlist	array
	0.02	0.01	0.01
	0.16	0.04	0.18
	2.39	0.41	1.45
	39.68	4.05	12
100000000	Killed	Killed	Killed
SLOPE	1.19722584	1.002697516	0.9119543705



memory usage [MB]

10000	0	0	0
100000	7.445	5.902	12.35
1000000	89.95	74.74	112.4
10000000	913.9	761.3	912.3
100000000	Killed	Killed	Killed

在log t - log N的圖中,我們發現三者消耗時間隨資料筆數關係接近y=ax+b.其中dlist的斜率接近1表示其N 與t的關係相當接近正比,此點符合我們的假說.BST的線略微上升推測其時間複雜度可能是NlogN的形式,未來有機會可以做更多的驗證

綜合以上的數據,皆良好的符合我們先前的預期外.值得注意的是array消耗的時間複雜度低於N,推測是其在新增元素時有很多共通的工作.