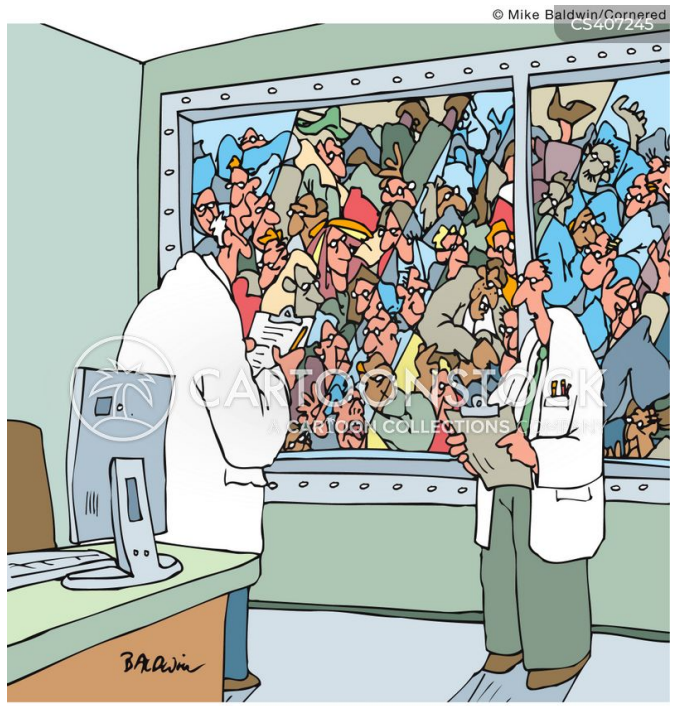
Laboratorio di Algoritmi e Strutture Dati

Secondo esercizio, seconda parte: alberi B (punti: 1 o 2)



"Basically, my study challenges the more the merrier theory."

Esperimenti con strutture dati

Con questo ultimo punto del secondo esercizio completiamo il programma di laboratorio.

Alberi B: una struttura ricorsiva

Assumiamo di aver fissato il parametro t.

```
struct B_tree_node{
    int[2 · t] keys
    bool leaf
    *struct B_tree_node parent
    *struct B_tree_node children[2 · t + 1]
}
```

```
struct Btree{
    int cardinality
    *struct B_tree_node root
}
```

Alberi B: gestione dell'esperimento singolo

Procediamo ancora una volta come negli altri casi.

Al livello più basso, il singolo passo consiste nel generare un numero fissato di chiavi casuali, scegliere un'operazione tra inserimento e cancellazione, ed eseguirla.

Usiamo la stessa strategia dell'esperimento semi-casuale.

Anche in questo caso non abbiamo visto la cancellazione in questi alberi. Se questa viene fatta in maniera autonoma, allora l'esercizio verrá valutato 2 punti. Altrimenti il confronto sará fatto sull'inserimento unicamente, e verrá valutato un punto.

Alberi B: gestione dell'esperimento

```
proc SingleExperimentB (max_keys, max_search, max_delete, max_instances)
{ t_tot = 0
  for (instance = 1 to max_instances)
  { Initialize(T)
    t_start = clock()
    for (key = 1 to max_keys)
    { BTreeInsert(T, Random())
    for (search = 1 to max_search)
    { BTreeSearch(T, Random())
    for (delete = 1 to max_delete)
    { BTreeDelete(T, Random())
    t_end = clock()
    t_elapsed = t_end − t_start
    t_tot = t_tot + t_elapsed
    Empty(T)
  return t_tot/max_keys
```

Alberi B: esperimento

```
proc Experiment (min_keys, max_keys)
{ step = 10
  max_instances = 5
  percentage_search = 60
  for (keys = min_keys to max_keys step step)
  { srand(SEED)
    max_search = keys * percentage_search/100
    max_delete = keys - max_search
    timeBST = SingleExperimentBST(keys, max_search, max_delete, max_instances)
    srand(SEED)
    timeRBT = SingleExperimentRBT(keys, max_search, max_delete, max_instances)
    srand(SEED)
    timeB = SingleExperimentB(keys, max_search, max_delete, max_instances)
    print(timeBST, timeRBT, timeB)
    SEED = SEED + 1
  }
}
```

Alberi B: esperimento

Vogliamo quindi realizzare un esperimento che consiste nel misurare il tempo necessario ad effettuare un certo numero, crescente, di operazioni standard di inserimento (e cancellazione) in un albero binario di ricerca, in un albero red-black, e in un albero B inizialmente vuoti. Il risultato richesto prevede una rappresentazione grafica delle curve di tempo e una dimostrazione sperimentale di correttezza attraverso test randomizzati e funzioni antagoniste.

Conclusione

Sperimentare di prima mano con le strutture dati ci dà l'occasione di capire davvero come queste sono fatte. Questa è condizione necessaria per un uso corretto delle strutture già implementate in librerie comuni.