**프로그래밍 실습 #6**

201501489 최영진

다음과 같은 기수 탐색 알고리즘을 파이썬으로 구현하고, 원소 수(N)와 키를

구성하는 비트 수(maxb)에 따른 실행 시간을 측정한 후 실행 시간을 비교해

본다.

1. 디지털 탐색 트리

class bitskey:  
 def \_\_init\_\_(self, x):  
 self.x = x  
  
 def get(self):  
 return self.x  
  
 def bits(self, k, j):  
 return (self.x >> k) & ~(~0 << j)  
  
class node:  
 def \_\_init\_\_(self, key):  
 self.key = bitskey(key)  
 self.left = None  
 self.right = None  
  
class Dict:  
 itemMin = bitskey(0)  
  
 z = node(itemMin)  
 head = node(itemMin)  
 head.left = z  
 head.right = z  
  
 def search(self, v):  
 v = bitskey(v)  
 x = self.head.left  
 b = maxb  
 self.z.key = v  
 while v.get() != x.key.get():  
 b = b - 1  
 if v.bits(b, 1):  
 x = x.right  
 else:  
 x = x.left  
 if x == self.z:  
 return -1  
 else:  
 return x.key.get()  
  
 def check(self, v):  
 v = bitskey(v)  
 x = self.head.left  
 p = x  
 b = maxb  
 self.z.key = v  
 while v.get() != x.key.get():  
 b = b - 1  
 if v.bits(b, 1):  
 if p != x:  
 p = x  
 x = x.right  
 else:  
 if p != x:  
 p = x  
 x = x.left  
 if x == self.z:  
 return -1  
 else:  
 print('key :', x.key.get(), ', parents :', p.key.get())  
 return  
  
 def insert(self, v):  
 v = bitskey(v)  
 b = maxb - 1  
 x = self.head.left  
 p = self.head  
  
 while x.key.get() != self.z.key.get():  
 p = x  
 if v.bits(b, 1):  
 x = x.right  
 else:  
 x = x.left  
 b -= 1  
 x = node(self.itemMin)  
 x.key = v  
 x.left = self.z  
 x.right = self.z  
 if v.bits(b+1, 1):  
 p.right = x  
 else:  
 p.left = x  
  
import random, time  
  
# N = 10000  
# maxb = 14  
# key = list(range(1, N+1))  
# s\_key = list(range(1, N+1))  
# random.shuffle(key)  
  
# N = 9  
# key = [2, 1, 8, 9, 7, 3, 6, 4, 5]  
# s\_key = list(range(1, 10))  
  
N = 7  
key = [1, 19, 5, 18, 3, 26, 9]  
s\_key = [1, 3, 5, 9, 18, 19, 26]  
maxb = 5  
  
d = Dict()  
for i in range(N):  
 d.insert(key[i])  
start\_time = time.time()  
for i in range(N):  
 result = d.search(s\_key[i])  
 if result.get() == -1 or result.get() != s\_key[i]:  
 print('탐색 오류')  
end\_time = time.time() - start\_time  
  
for i in range(N):  
 d.check(s\_key[i])  
  
print('디지털 탐색 트리의 실행 시간 (N = %d) : %0.3f'%(N, end\_time))  
print('탐색 완료')

(2) 기수 탐색 트라이

class bitskey:  
 def \_\_init\_\_(self, x):  
 self.x = x  
  
 def get(self):  
 return self.x  
  
 def bits(self, k, j):  
 return (self.x >> k) & ~(~0 << j)  
  
class node:  
 def \_\_init\_\_(self, key):  
 if key.get() == 0:  
 self.key = bitskey(0)  
 self.external = False  
 else:  
 self.key = key  
 self.external = True  
 self.left = 0  
 self.right = 0  
  
class Dict:  
 itemMin = bitskey(0)  
 head = 0  
 head\_check = False  
  
 def search(self, v):  
 v = bitskey(v)  
 return self.searchR(self.head, v, maxb-1)  
  
 def check(self, v):  
 v = bitskey(v)  
 print(v.get(), end = ' ')  
 return self.checkR(self.head, v, maxb-1)  
  
 def checkR(self, h, v, d):  
 if h == 0:  
 return self.itemMin  
  
 if v.get() == h.key.get():  
 print()  
 return  
 if v.bits(d, 1) == 0:  
 print('left', end = ' ')  
 return self.checkR(h.left, v, d - 1)  
 else:  
 print('right', end = ' ')  
 return self.checkR(h.right, v, d - 1)  
  
 def insert(self, v):  
 v = bitskey(v)  
 self.insertR(self.head, v, maxb-1)  
  
 def insertR(self, h, v, d):  
 if h == 0:  
 h = node(v)  
 if self.head\_check == False:  
 self.head = h  
 return h  
 if h.external:  
 leaf = node(v)  
 h = self.split(leaf, h, d)  
 if self.head\_check == False:  
 self.head = h  
 self.head\_check = True  
 return h  
 if v.bits(d, 1) == 0:  
 h.left = self.insertR(h.left, v, d-1)  
 else:  
 h.right = self.insertR(h.right, v, d-1)  
 return h  
  
 def split(self, p, q, d):  
 t = node(self.itemMin)  
 if ((p.key.bits(d, 1)) \* 2 + (q.key.bits(d, 1))) == 0:  
 t.left = self.split(p, q, d-1)  
 elif ((p.key.bits(d, 1)) \* 2 + (q.key.bits(d, 1))) == 1:  
 t.left = p  
 t.right = q  
 elif ((p.key.bits(d, 1)) \* 2 + (q.key.bits(d, 1))) == 2:  
 t.right = p  
 t.left = q  
 elif ((p.key.bits(d, 1)) \* 2 + (q.key.bits(d, 1))) == 3:  
 t.right = self.split(p, q, d-1)  
 return t  
  
 def searchR(self, h, v, d):  
 if h == 0:  
 return self.itemMin  
  
 if v.get() == h.key.get():  
 return v  
 if v.bits(d, 1) == 0:  
 return self.searchR(h.left, v, d-1)  
 else:  
 return self.searchR(h.right, v, d-1)  
  
import random, time  
  
N = 7  
key = [1, 19, 5, 18, 3, 26, 9]  
s\_key = [1, 3, 5, 9, 18, 19, 26]  
maxb = 5  
  
d = Dict()  
for i in range(N):  
 d.insert(key[i])  
d.head.external = True  
  
start\_time = time.time()  
for i in range(N):  
 result = d.search(s\_key[i])  
 if result.get() == -1 or result.get() != s\_key[i]:  
 print('탐색 오류')  
end\_time = time.time() - start\_time  
  
for i in range(N):  
 d.check(s\_key[i])  
  
print('디지털 탐색 트리의 실행 시간 (N = %d) : %0.3f'%(N, end\_time))  
print('탐색 완료')

(3) 패트리샤 트리

maxb = 5  
  
class bitskey:  
 def \_\_init\_\_(self, x):  
 self.x = x  
  
 def get(self):  
 return self.x  
  
 def bits(self, k, j):  
 return (self.x >> k) & ~(~0 << j)  
  
class node:  
 def \_\_init\_\_(self, key):  
 self.key = key  
 self.b = None  
 self.left = None  
 self.right = None  
  
class Dict():  
 itemMin = bitskey(0)  
 head = node(itemMin)  
 head.b = maxb  
 head.left = head.right = head  
  
 def search(self, v):  
 v = bitskey(v)  
 p = self.head  
 x = self.head.left  
 while p.b > x.b:  
 p = x  
 if self.bits(v, x.b, 1):  
 x = x.right  
 else:  
 x = x.left  
 if v.get() != x.key.get(): return self.itemMin  
 return x.key  
  
 def check(self, v):  
 v = bitskey(v)  
 x = self.head.left  
 p = x  
 b = maxb  
 self.head.key = v  
 while v.get() != x.key.get():  
 b = b - 1  
 if v.bits(b, 1):  
 if p != x:  
 p = x  
 x = x.right  
 else:  
 if p != x:  
 p = x  
 x = x.left  
 if x == self.head:  
 return -1  
 else:  
 print('key :', x.key.get(), ', parents :', p.key.get())  
 return  
  
 def insert(self, v):  
 v = bitskey(v)  
 i = maxb  
 p = self.head  
 t = self.head.left  
 while p.b > t.b:  
 p = t  
 if self.bits(v, t.b, 1):  
 t = t.right  
 else:  
 t = t.left  
 if v.get() == t.key.get() : return  
 while self.bits(t.key, i, 1) == self.bits(v, i, 1):  
 i -= 1  
 p = self.head  
 x = self.head.left  
 while p.b > x.b and x.b > i:  
 p = x  
 if self.bits(v, x.b, 1):  
 x = x.right  
 else:  
 x = x.left  
 t = node(self.itemMin)  
 t.key = v  
 t.b = i  
 if self.bits(v, t.b, 1):  
 t.left = x  
 t.right = t  
 else:  
 t.left = t  
 t.right = x  
  
 if self.bits(v, p.b, 1):  
 p.right = t  
 else:  
 p.left = t  
  
 def bits(self, item, bit, cmp):  
 if item.bits(bit, 1) == cmp:  
 return 1  
 else:  
 return 0  
  
import random, time  
  
N = 7  
key = [1, 19, 5, 18, 3, 26, 9]  
s\_key = [1, 3, 5, 9, 18, 19, 26]  
  
d = Dict()  
for i in range(N):  
 d.insert(key[i])  
start\_time = time.time()  
for i in range(N):  
 result = d.search(s\_key[i])  
 if result.get() == -1 or result.get() != s\_key[i]:  
 print('탐색 오류')  
end\_time = time.time() - start\_time  
  
for i in range(N):  
 d.check(s\_key[i])  
  
print('패트리샤 트리의 실행 시간 (N=%d) : %0.3f' %(N, end\_time))  
print('탐색 완료')

**결과**

사용된 배열 [108, 355, 230, 867, 240, 305, 538, 772, 233, 107, 893, 451, 435, 126, 761, 814, 949, 15, 997, 578, 20, 987, 258, 310, 820, 464, 254, 528, 115, 59, 714, 735, 64, 861, 807, 509, 314, 504, 540, 107, 552, 981, 856, 681, 672, 568, 5, 10, 667, 430, 705, 410, 622, 325, 628, 151, 725, 346, 228, 337, 151, 776, 937, 585, 118, 872, 155, 513, 236, 602, 249, 904, 7, 294, 746, 947, 179, 744, 199, 176, 774, 774, 749, 626, 835, 469, 481, 690, 139, 140, 289, 394, 703, 955, 821, 532, 581, 487, 265, 412]

(1) 디지털 탐색 트리

===============================================================

key : 5 , parents : 20

key : 7 , parents : 5

key : 10 , parents : 5

key : 15 , parents : 107

key : 20 , parents : 15

key : 59 , parents : 15

key : 64 , parents : 126

key : 107 , parents : 230

key : 107 , parents : 230

key : 108 , parents : 108

key : 115 , parents : 126

key : 118 , parents : 115

key : 126 , parents : 107

key : 139 , parents : 151

key : 140 , parents : 139

key : 151 , parents : 240

key : 151 , parents : 240

key : 155 , parents : 151

key : 176 , parents : 179

key : 179 , parents : 151

key : 199 , parents : 233

key : 228 , parents : 254

key : 230 , parents : 355

key : 233 , parents : 240

key : 236 , parents : 228

key : 240 , parents : 230

key : 249 , parents : 254

key : 254 , parents : 233

key : 258 , parents : 305

key : 265 , parents : 310

key : 289 , parents : 294

key : 294 , parents : 314

key : 305 , parents : 355

key : 310 , parents : 258

key : 314 , parents : 310

key : 325 , parents : 258

key : 337 , parents : 346

key : 346 , parents : 325

key : 355 , parents : 108

key : 394 , parents : 410

key : 410 , parents : 435

key : 412 , parents : 410

key : 430 , parents : 435

key : 435 , parents : 451

key : 451 , parents : 305

key : 464 , parents : 451

key : 469 , parents : 464

key : 481 , parents : 509

key : 487 , parents : 481

key : 504 , parents : 509

key : 509 , parents : 464

key : 513 , parents : 540

key : 528 , parents : 578

key : 532 , parents : 540

key : 538 , parents : 867

key : 540 , parents : 528

key : 552 , parents : 528

key : 568 , parents : 552

key : 578 , parents : 538

key : 581 , parents : 585

key : 585 , parents : 622

key : 602 , parents : 585

key : 622 , parents : 578

key : 626 , parents : 628

key : 628 , parents : 622

key : 667 , parents : 681

key : 672 , parents : 681

key : 681 , parents : 761

key : 690 , parents : 672

key : 703 , parents : 690

key : 705 , parents : 735

key : 714 , parents : 761

key : 725 , parents : 735

key : 735 , parents : 714

key : 744 , parents : 746

key : 746 , parents : 714

key : 749 , parents : 744

key : 761 , parents : 538

key : 772 , parents : 867

key : 774 , parents : 776

key : 774 , parents : 776

key : 776 , parents : 814

key : 807 , parents : 820

key : 814 , parents : 893

key : 820 , parents : 814

key : 821 , parents : 820

key : 835 , parents : 856

key : 856 , parents : 861

key : 861 , parents : 893

key : 867 , parents : 108

key : 872 , parents : 861

key : 893 , parents : 772

key : 904 , parents : 937

key : 937 , parents : 949

key : 947 , parents : 937

key : 949 , parents : 772

key : 955 , parents : 947

key : 981 , parents : 987

key : 987 , parents : 997

key : 997 , parents : 949

디지털 탐색 트리의 실행 시간 (N = 100) : 0.0009982586

탐색 완료

===============================================================

(2) 기수 탐색 트라이

===============================================================

5 left left left left left left left right left

7 left left left left left left left right right

10 left left left left left left right left

15 left left left left left left right right

20 left left left left left right

59 left left left left right

64 left left left right left

107 left left left right right left right left right right 107 left left left right right left right left right right 108 left left left right right left right right

115 left left left right right right left left

118 left left left right right right left right

126 left left left right right right right

139 left left right left left left right left

140 left left right left left left right right

151 left left right left left right left right right right 151 left left right left left right left right right right 155 left left right left left right right

176 left left right left right right left left left

179 left left right left right right left left right

199 left left right right left

228 left left right right right left left right left

230 left left right right right left left right right

233 left left right right right left right left

236 left left right right right left right right

240 left left right right right right left

249 left left right right right right right left

254 left left right right right right right right

258 left right left left left left left

265 left right left left left left right

289 left right left left right left left left

294 left right left left right left left right

305 left right left left right right left left

310 left right left left right right left right

314 left right left left right right right

325 left right left right left left

337 left right left right left right left

346 left right left right left right right

355 left right left right right

394 left right right left left left

410 left right right left left right right left

412 left right right left left right right right

430 left right right left right left

435 left right right left right right

451 left right right right left left

464 left right right right left right left left

469 left right right right left right left right

481 left right right right right left left left

487 left right right right right left left right

504 left right right right right right right left

509 left right right right right right right right

513 right left left left left left

528 right left left left left right left left

532 right left left left left right left right

538 right left left left left right right left

540 right left left left left right right right

552 right left left left right left

568 right left left left right right

578 right left left right left left left left

581 right left left right left left left right

585 right left left right left left right

602 right left left right left right

622 right left left right right left

626 right left left right right right left left

628 right left left right right right left right

667 right left right left left

672 right left right left right left left

681 right left right left right left right

690 right left right left right right left

703 right left right left right right right

705 right left right right left left left

714 right left right right left left right

725 right left right right left right left

735 right left right right left right right

744 right left right right right left right left left

746 right left right right right left right left right

749 right left right right right left right right

761 right left right right right right

772 right right left left left left left right left

774 right right left left left left left right right left 774 right right left left left left left right right left 776 right right left left left left right

807 right right left left right left left

814 right right left left right left right

820 right right left left right right left right left left 821 right right left left right right left right left right 835 right right left right left left

856 right right left right left right right left

861 right right left right left right right right

867 right right left right right left left

872 right right left right right left right

893 right right left right right right

904 right right right left left

937 right right right left right left

947 right right right left right right left left

949 right right right left right right left right

955 right right right left right right right

981 right right right right left right left

987 right right right right left right right

997 right right right right right

기수 탐색 트리의 실행 시간 (N = 100) : 0.0010020733

탐색 완료

===============================================================

(3) 패트리샤 트리 : 디지털 탐색 트리와 동일한 check() 함수 사용

===============================================================

key : 5 , parents : 15

key : 7 , parents : 5

key : 10 , parents : 15

key : 15 , parents : 20

key : 20 , parents : 59

key : 59 , parents : 108

key : 64 , parents : 108

key : 107 , parents : 126

key : 107 , parents : 126

key : 108 , parents : 230

key : 115 , parents : 126

key : 118 , parents : 115

key : 126 , parents : 64

key : 139 , parents : 179

key : 140 , parents : 139

key : 151 , parents : 230

key : 151 , parents : 230

key : 155 , parents : 139

key : 176 , parents : 179

key : 179 , parents : 151

key : 199 , parents : 151

key : 228 , parents : 233

key : 230 , parents : 355

key : 233 , parents : 240

key : 236 , parents : 233

key : 240 , parents : 199

key : 249 , parents : 254

key : 254 , parents : 240

key : 258 , parents : 305

key : 265 , parents : 258

key : 289 , parents : 294

key : 294 , parents : 258

key : 305 , parents : 451

key : 310 , parents : 314

key : 314 , parents : 294

key : 325 , parents : 305

key : 337 , parents : 346

key : 346 , parents : 325

key : 355 , parents : 867

key : 394 , parents : 410

key : 410 , parents : 435

key : 412 , parents : 394

key : 430 , parents : 410

key : 435 , parents : 451

key : 451 , parents : 355

key : 464 , parents : 509

key : 469 , parents : 464

key : 481 , parents : 509

key : 487 , parents : 481

key : 504 , parents : 481

key : 509 , parents : 435

key : 513 , parents : 552

key : 528 , parents : 513

key : 532 , parents : 528

key : 538 , parents : 867

key : 540 , parents : 528

key : 552 , parents : 578

key : 568 , parents : 552

key : 578 , parents : 761

key : 581 , parents : 585

key : 585 , parents : 602

key : 602 , parents : 622

key : 622 , parents : 578

key : 626 , parents : 628

key : 628 , parents : 622

key : 667 , parents : 681

key : 672 , parents : 690

key : 681 , parents : 761

key : 690 , parents : 667

key : 703 , parents : 690

key : 705 , parents : 735

key : 714 , parents : 681

key : 725 , parents : 735

key : 735 , parents : 714

key : 744 , parents : 749

key : 746 , parents : 714

key : 749 , parents : 746

key : 761 , parents : 538

key : 772 , parents : 949

key : 774 , parents : 776

key : 774 , parents : 776

key : 776 , parents : 814

key : 807 , parents : 820

key : 814 , parents : 772

key : 820 , parents : 814

key : 821 , parents : 820

key : 835 , parents : 861

key : 856 , parents : 835

key : 861 , parents : 772

key : 867 , parents : 867

key : 872 , parents : 893

key : 893 , parents : 861

key : 904 , parents : 997

key : 937 , parents : 904

key : 947 , parents : 955

key : 949 , parents : 538

key : 955 , parents : 937

key : 981 , parents : 987

key : 987 , parents : 997

key : 997 , parents : 949

패트리샤 트리의 실행 시간 (N=100) : 0.0009880066

탐색 완료

===============================================================

