**4.17  字典的並集與交集**

感謝：Tom Good、Andy McKay、Sami Hangaslammi、Robin Siebler

任務

給定兩個字典，需要找到兩個字典都包含的鍵（交集），或者同時屬於兩個字典的鍵（並集）。

解決方案

有時，尤其是在Python 2.3中，你會發現對字典的使用完全是對集合的一種具體化的體現。在這個要求中，只需要考慮鍵，不用考慮鍵的對應值，一般可以通過調用dict.fromkeys來創建字典，像這樣：

1. a = dict.fromkeys(xrange(1000))
2. b = dict.fromkeys(xrange(500, 1500))

最快計算出並集的方法是：

1. union = dict(a, \*\*b)

而最快且最簡潔地獲得交集的方法是：

1. inter = dict.fromkeys([x for x in a if x in b])

如果字典a和b的條目的數目差異很大，那麼在for子句中用較短的那個字典，在if子句中用較長的字典會有利於提升運算速度。在這種考慮之下，犧牲簡潔性以獲取性能似乎是值得的，交集計算可以被改為：

1. if len(a) **<** **len**(b):
2. inter = dict.fromkeys([x for x in a if x not in b])
3. else:
4. inter = dict.fromkeys([x for x in b if x not in a])

Python也提供了直接代表集合的類型（標準庫中的sets模組，在Python 2.4中已經成為了內建的部分）。可以把下面的代碼片段用在模組的開頭，這個代碼片段確保了名字set被綁定到了適合的類型，這樣在整個模組中，無論你用Python 2.3還是2.4，都可以使用同樣的代碼：

1. try:
2. set
3. except NameError:
4. from sets import Set as set

這樣做的好處是，可以到處使用set類型，同時還獲得了清晰和簡潔，以及速度的提升（在Python 2.4中）：

1. a = set(xrange(1000))
2. b = set(xrange(500, 1500))
3. union = a | b
4. inter = a & b

討論

雖然Python 2.3的標準庫模組sets已經提供了一個優雅的資料類型set來代表集合（帶有可雜湊（hashable）的元素），但由於歷史原因，使用dict來代表集合仍然是很普遍的。基於這個目的，本節展示了如何用最快的方法來計算這種集合的交集和並集。本節的代碼在我的電腦上，並集計算耗時260 s，交集計算則耗時690 s（Python 2.3；在Python 2.4中，這兩個數位分別是260 s和600 s），而其他的基於迴圈或者生成器運算式的方法會更慢。

不過，最好還是用set類型而不是字典來代表集合。如同本節所示，使用set能讓代碼更加直接和易讀。如果你不喜歡或操作符（|）和與操作符（&），可以使用等價的a.union(b)和a.intersection(b)。這樣操作除了清晰，速度也有提升，特別是在Pyton 2.4中，計算並集需要260 s，但計算交集只需要210 s。即使是在Python 2.3，其速度也是可以接受的：並集計算耗時270 s，交集計算耗時650 s，沒有在Python 2.4快，但如果你仍然用字典來代表集合的話，速度其實是相當的。最後一點，一旦你引入set類型（無論是Python 2.4內建的，還是通過Python標準庫sets模組引入的，介面是一樣的），你將獲得豐富的集合操作。舉個例子，屬於a或者b但卻不屬於a和b的交集的集合是a^b，可以等價地被表示為a.symmetric\_difference(b)。

即使由於某些原因使用了dict，也應當盡可能地用set來完成集合操作。舉個例子，假設你有個字典phones，將人名映射到電話號碼，還有個字典address，將人名映射到位址。最清楚簡單地列印所有同時知道位址和電話號碼的人名及其相關資料的方法是：

1. for name in set(phones) & set(addresses):
2. print name, phones[name], addresses[name]

跟下面的方法比，這非常簡潔，雖然清晰度可能還有爭議：

1. for name in phones:
2. if name in addresses:
3. print name, phones[name], addresses[name]

另一個很好的可選方法是：

1. for name in set(phones).intersection(addresses):
2. print name, phones[name], addresses[name]

如果使用intersection方法，而不是&交集操作，就不需要將兩個字典都轉化成set，只需要其中一個。然後再對轉化後的set調用intersection，並傳入另一個dict作為intersection方法的參數。

更多資料

Library Reference和Python in a Nutshell的映射類型、sets模組及Python 2.4中的內建set類型。

**範例碼:**

# -\*- coding: utf-8 -\*-  
#!/usr/local/bin/python  
  
import pdb  
#创建字典  
  
pdb.set\_trace()  
  
a = dict.fromkeys(xrange(1000))  
b = dict.fromkeys(xrange(500,1000))  
  
#最快计算出并集的方法  
union = dict(a,\*\*b)  
print union  
  
#最快,最简洁计算出交集的方法  
inter = dict.fromkeys([x for x in a if x in b])  
print inter  
  
  
#如果字典a,b 的条目的数目差异很大，那么在for 子句中用较短的那个字典，在if 语句中用较长的那个字典会有利于运算速度的提升。  
if len(a) < len(b) :  
    inter = dict.fromkeys([x for x in a if x not in b ])   
else:  
    inter = dict.fromkeys([x for x in b if x not in a ])   
print inter   
  
#集合的运算，最好还是用set   
a = set(xrange(1000))  
b = set(xrange(500,1500))  
union = a | b   
inter = a & b   
print union  
print inter  
  
#属于a或b ,但却不属于a 和 b 的交集的集合  
print a.symmetric\_difference(b)