Bu projede, data setini m-boyutlu bir hash tablosuna depolamak için verilen giriş dosyalarında birkaç hash fonksiyonunu uygulamanız istenir. İki girdi dosyası var : “vocab.txt” ve “search.txt . İlk dosya bir ingilizce derlemeden toplanan benzersiz belirteçler içerir. Her belirteç bir satırda bulunur ve belirteçlerin indeks değerleri, hangi satırda olduklarına karşılık gelir. Diğer dosya arama işlemlerinde kullanılmak üzere verilir. Bu dosya belirteçler içerir ve hash table farklı hash fonksiyonlarına göre oluşturulduktan sonra hash table içersinde aranır.

1)(15 pt) Aşağıdaki hash fonksiyonunu kullanarak dize öğelerini depolamak için linear probing stratejisi kullanan bir m boyutlu hash table uygulayın. Karma tablonuzun her bir slotu yalnızca string içerebilir. (Not: Ekleme işlemi sırasında int değerlerini kaydetmezsiniz. Arama işleminde int değerlerini, ekleme işleminde yaptığınız gibi kullanacaksınız.)

ℎ(𝑘,𝑖) =(𝑘+𝑖) 𝑚𝑜𝑑 𝑚 and 𝑖∈[0,𝑚−1]

2) (15 pt) Aşağıdaki karma işlevini kullanarak dize string elemanları depolamak için double hashing yazma stratejisi kullanan bir m boyutlu karma tablosu oluştur. Hash tablonuzun her bir slotu sadece string değişkenini içerebilir.

h(k, i) = (h1(k) + i \* h2(k)) mod m and 𝑖∈[0,𝑚−1]

h1(k) = k mod m

h2(k) = p – ( k mod p ), p is a prime number and p∈[0,𝑚−1]

3)(20 pt) String elemanlarını saklamak için universal hashing stratejisi kullanan bir m boyutlu karma tablosu oluştur. Herhangi bir çakışma durumunda, open adressing stratejilerinden yardım alabilirsiniz(linear probing). Universal hashin stratejisinin detayları aşağıda verilmiştir.

Adım # 1: Asal olmak için tablo boyutunu m seçin.

Adım # 2: k keyini 2 haneye (sonunda, r chunks) ayırın, böylece k = <k0, k1, ..., kr> ve 0 ≤ ki <m.( Bu ödev için r = 3).

Adım # 3: 𝑎𝑖 = [𝑎0, 𝑎1,…, 𝑎𝑟] öğesini seçin, burada her 𝑎𝑖 [0, 𝑚 − 1] aralığında rasgele oluşturulur.

Adım # 4: Aşağıda verilen hash fonksiyonunu uygulayın:

K için ayrıştırma örneği:

𝑘= 96356→k0=9, 𝑘1= 63, 𝑘2=56

𝑘= 1398→k0=0, 𝑘1= 13, 𝑘2= 98

𝑘= 548→k0=0, 𝑘1= 5, 𝑘2= 48

𝑘= 83→k0=0, 𝑘1= 0, 𝑘2= 83

𝑘= 7→k0=0, 𝑘1= 0, 𝑘2= 7

4. (30 points) Arama Fonksiyonları

Üç fonksiyon yazın ve bunların her biri, sırasıyla (1), (2) ve (3) 'te uyguladığınız hash tablosunda verilen key kümesini arasın.

5. (20 points) Yeni eklenen keyler hash tablosundaki zaten dolu bir slota eşlendiği durumlara collision denir. Bu bölümde, bu ödevde uygulanmakta olan hash fonksiyonları, ekleme / arama işlemleri sırasında meydana gelen collision sayılarına göre karşılaştırılmaktadır. Bir öğeyi eklerken / ararken collasion sayısı şöyle tanımlanır:

çarpışmalar = 0

Compute h (k, i)

İf h(k , i ) not empty

İncrease collisions by 1;

Yukarıdaki tanımı kullanarak, “vocab.txt” içindeki tüm öğeleri hash tablolarına ekleyin, her strateji için collision sayısını hesaplayın ve her strateji ve m değeri için ortaya çıkan collision sayısını içeren tabloyu doldurun. Benzer şekilde, “search.txt” deki tüm öğeleri arayın ve her strateji için çarpışma sayısını hesaplayın. Sonuçlara yorum yapın.