GPH: Similarity Search in Hamming Space

Brief Intro of the paper

What is the Hamming distance?

길이가 N 인 2^N 개의 binary strings 집합

- Main focus of this paper
  : Similarity search on binary vector in Hamming space
  - 하나의 object를 n 차원 binary vector x 라고 봄 (e.g. $x = 0100$)
    - $x[i]$는 x의 i차원의 값
  - 만약 $\Delta(x[i], y[i]) = 0$ 이면, $x[i] = y[i]$, 1 이면, $x[i] \neq y[i]$
- Hamming distance between two vectors x and y, denoted $H(x, y)$

$$H(x, y) = \sum_{i=1}^{n} \Delta(x[i], y[i])$$

- Hamming distance search 란?
  Data object들의 collection D, query object q 가 주어졌을 때, q와의 Hamming distance가 threshold τ 이하인 objects를 모두 찾는 것 (i.e. $\{x \mid x \in D, H(x, q) \leq \tau\}$) ← Hamming distance constraint

Application of the hamming distance

1. Image retrieval
   - Image는 binary vector로 표현되고 Hamming distance는 image간의 비유사성(dissimilarity)을 측정하기 위해 활용됨

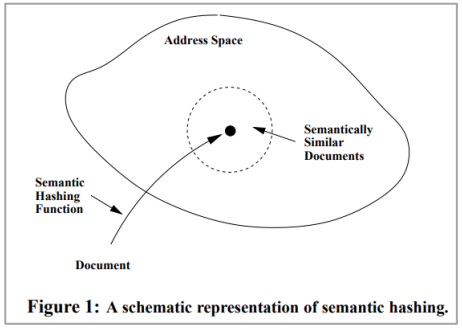

Figure 1: A schematic representation of semantic hashing.

2. Document information retrieval
   - Document가 hashing을 통해 binary vector로 표현됨
   - Web page 중복 제거를 위해 Google은 SimHash를 사용
     - Web page를 64-bit vector로 바꾸고, vector들의 Hamming distance가 3 이내라면 두 pages가 거의 중복된다 라고 고려함
3. Finding similar molecules in chem-informatics
   - 분자(molecule)가 binary vector로 변환되어, 분자들 간의 유사성을 측정할 수 있음
   - 이때, 유사성 측정에 Hamming distance constraint을 적용

Basic Pigeonhole Principle

- Hamming distance search에서, 주어진 query q에 답하기 위한 State-of-the-art 접근 방법들은 주로 *비둘기집 원리(pigeonhole principle)*에 의해 candidate들의 집합을 생성하고 이들을 검증(verify) 함
  - 이는 두 vector가 유사하다면, 두 vector들로부터 유사한 partition들의 쌍이 있을 것이라는 직관에 기반함
- *Lemma 1* (Basic Pigeonhole Principle)
  x 그리고 y 가 m개의 partition들로 나눠질 때, 각 partition은 $\frac{n}{m}$ 차원으로 구성되어 있다. x_i와 y_i $(1 \leq i \leq m)$를 각각 x와 y의 partition 이라고 하자. $H(x,y) \leq \tau$이면, $H(x_i,y_i) \leq \left\lfloor \frac{\tau}{m} \right\rfloor$ 를 만족하는 partition i 가 적어도 하나 이상 존재한다.
  - 이 논문에서는 $n \bmod m = 0$ 을 가정
- 위 조건을 만족하는 data object x,
  즉, $\exists i, H(x_i,q_i) \leq \left\lfloor \frac{\tau}{m} \right\rfloor$ 를 *candidate*라 함
  - 이러한 candidate들은 후에 query에 대한 실제 Hamming distance 계산을 통해 검증하므로, query 처리 performance는 candidates의 개수에 크게 좌우됨

Overview of Existing Approaches

- Hamming distance search의 state-of-the-art method 중 Multi-index Hamming(MIH)를 간략히 소개

1. MIH는 n차원 data object를 같은 m 차원(equi-width)을 가지는 partition들로 나눔
2. Basic pigeonhole principle에 의해, MIH는 각 $n' = \left\lfloor \frac{n}{m} \right\rfloor$ 차원에 대해 threshold $\tau' = \left\lfloor \frac{\tau}{m} \right\rfloor$ 를 이용하여 Hamming distance 계산을 수행함
   - 또한 각 data object의 partition이 object ID로 mapping되는 inverted index 를 생성
3. Query의 각 partition마다 Hamming distance가 τ' 이내에 있는 n' 차원 vector들을 모두 조사
   - 해당 vector들을 *signature*라 부름
4. Index내에 있는 signature를 look up하여 Candidate들을 찾아내고 이들을 검증함

Weaknesses of Basic Pigeonhole Principle

- Basic pigeonhole principle에 기반한 filtering condition은 major한 단점(drawbacks)을 지니고 있음
  - Filtering condition은 해당되는 partition 마다 대응되는 threshold 들의 vector로 unique하게 특징지어짐
  - Basic pigeonhole principle의 threshold vector는 다음과 같음

$$T_{basic} = \left[\left\lfloor\frac{\tau}{m}\right\rfloor, \ldots, \left\lfloor\frac{\tau}{m}\right\rfloor\right]$$

  - 또한 threshold vector들 간의 $domainance$ 관계는 다음과 같음

T_1이 T_2를 dominate 한다 또는 $T_1 \prec T_2$ 라는 의미는, $\forall i \in \{1, \ldots, m\}$, $T_1[i] \leq T_2[i]$ 그리고 $[T_1[i], T_2[i]] \cap [-1, n_i - 1] \neq \emptyset$, 그리고 $\exists i, T_1[i] < T_2[i]$를 만족한다. (e.g. $T_1 = [2,2,3]$, $T_2 = [3,3,3]$)

1. T_{basic}은 언제나 tight 하지만은 않음
2. Filtering condition은 partition들의 데이터 분포를 고려하지 않음

1. T_{basic}은 언제나 tight 하지만은 않음

- Threshold vector T가 tight 하다 라는 의미는 **두가지 조건**을 충족해야 함

1. Correctness: query에 대한 Hamming distance가 threshold 이내인 모든 vector들을 T에 기반한 filtering condition으로 찾을 수 있어야 함
2. Minimality: correctness를 T 외에 보장할 수 있으면서 T를 dominate 할 수 있는 또 다른 vector T'는 존재해서는 안됨
   - Candidate size와 threshold는 단조 관계이기 때문에, T_{basic}을 dominate 할 수 있는 threshold vector에 기반한 알고리즘은 T_{basic}에 기반한 알고리즘 보다 더 작거나 같은 candidates 수를 생성함

Example 1)

- $\tau = 9$ 그리고 $m = 3$인 경우, $T_{basic} = [3,3,3]$
- $T = [2,2,3]$ 은 correctness와 minimality를 보장하는 dominating threshold vector
- $[2,3,2]$ 또는 $[4,3,0]$ 과 같이, 같은 τ에 multiple tight threshold vector가 존재할 수 있음

2. Filtering condition은 partition들의 데이터 분포를 고려하지 않음

- 실제 데이터에서는 차원 간의 correlations 그리고 skewness가 종종 존재함
- T_{basic} 처럼 각 partition마다 동등한 threshold를 할당한다면 특정 partition에 지나친 candidate 발생 가능(poor selectivity)
- 이러한 문제를 해결하기 위해, 최근 연구에서는 less skew partition을 위한 partition 재배치를 시도하거나 heuristic 방식으로 threshold를 조정

TABLE I
BENEFITS OF ADAPTIVE PARTITIONING AND THRESHOLDING

	Equi-width Partitioning		Variable Partitioning	
	Partition 1	Partition 2	Partition 1	Partition 2
$x^1 = 00000000$	0000	0000	000000	00
$x^2 = 00000111$	0000	0111	000001	11
$x^3 = 00001111$	0000	1111	000011	11
$x^4 = 10011111$	1001	1111	100111	11
$q^1 = 10000000$	1000 $\tau_1 = 1$	0000 $\tau_2 = 1$	100000 $\tau_1 = 2$	00 $\tau_2 = 0$

Example 2)

- $n = 8, m = 2$ 그리고 $\tau = 2$
- $T_{baisc} = [1,1]$일 경우 4개의 data vector들이 candidate 선정 (x^1, x^2, x^3, x^4)
- partition 차원을 달리하고 $T = [2,0]$을 사용하면 candidate size 는 2 (x^1, x^2)

Proposed (novel) method

- Hamming distance search 문제를 풀고 언급된 weakness들을 해결하기 위해 새로운 방법을 제안 : GPH algorithm (=1+2+3)

1. Pigeonhole의 새로운 형태인 General Pigeonhole Principle(GPP)을 제안
   - $\tau - m + 1 \sim \tau$사이의 값이 나오도록 m 개의 partition들의 threshold를 분배
     → 더욱 엄격한 filtering condition
   - 각 partition 에 대한 threshold는 variable 로서 $[-1, \tau]$ 범위의 값
     → 다른 partition들에 대한 적절한 threshold를 선택
   - GPP에 기반한 candidate condition은 tight 함을 증명
     (i.e. 각 partition에 할당된 threshold는 그 이상 줄을 수 없음)
2. Query processing에 대한 비용(cost) 모델을 기반으로 하여 partition 마다 threshold를 할당하는 online algorithm을 고안
   - 데이터 skewness와 차원 correlation들의 문제들을 완화시킬 수 있음
3. 차원들의 분산 정도를 고려하여 vector들을 나누는 방법(partitioning)을 최적화하는 offline algorithm을 고안

Summary

- Hamming 공간에서의 유사성 검색은 query vector로 부터 threshold 이하의 Hamming distance를 가진 binary vector들을 찾는 문제
- State-of-the-art 접근법들은 주로 pigeonhole principle을 이용하여 candidate들의 집합을 생성하고 검증하여 query에 대한 해답을 제시하고 해당 문제를 해결
- 하지만 기존의 pigeonhole principle은 언제나 tight 하지 않으므로 불필요한 candidate을 가져올 수 있음
- 또한 같은 차원을 가진 partition들에게 모두 같은 threshold를 분배함으로써 데이터 분포의 skewness를 고려하지 않음
- 논문에서는 pigeonhole principle의 새로운 형태를 제안하여 다른 차원을 가진 partition들에게 각기 다른 threshold를 부여함
- 또한 차원을 나누거나 threshold를 할당하는데 있어서 cost 기반 방식을 고안하고 적은 수의 candidate를 위한 tight constraint를 고안