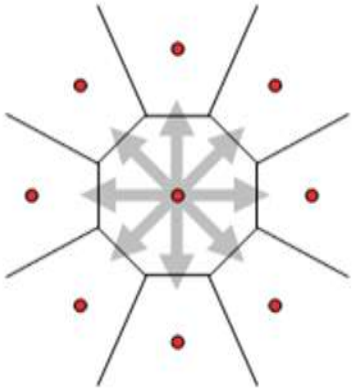
هشینگ حساس به محل

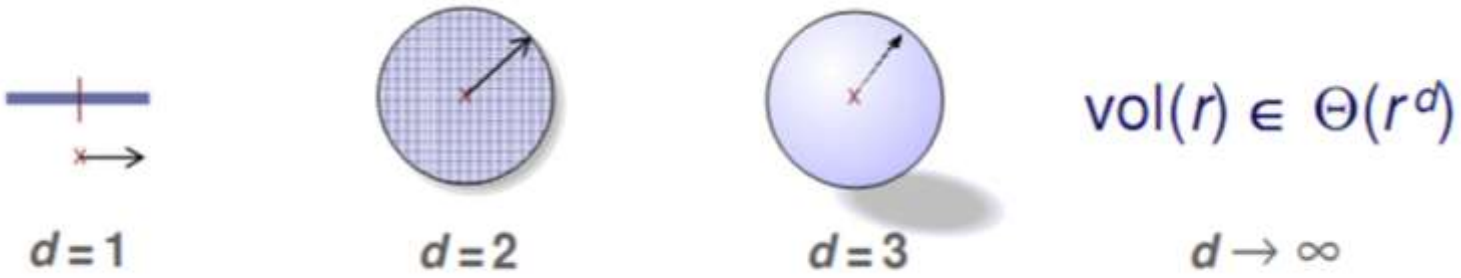
سپیده آقاملائی

مشکلات ابعاد بالا

- d همسایه ی مستقل در هر جهت

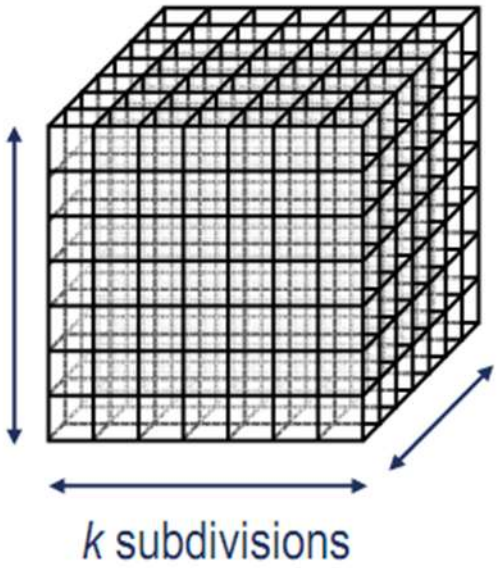


- سرعت رشد بالای همسایگی با افزایش شعاع



توری کمکی نمی کند.

- تکنیک های تقسیم معمولی کار نمی کنند.
  - توری های $k \times k$ دارای k^d خانه هستند.
- توری های adaptive هم جواب نمی دهند.



نفرین ابعاد

- دورترین همسایه ~ نزدیک ترین همسایه
- لم JL

JL-Lemma: [Dasgupta et al. 99]

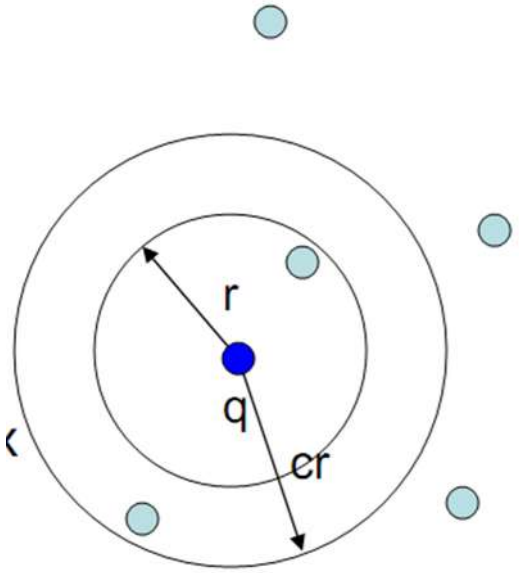
- Point set P in $\mathbb{R}^d$, $n := \#P$
- There is $f: \mathbb{R}^d \to \mathbb{R}^k$, $k \in O(\varepsilon^{-2} \ln n)$
  $(k \geq 4(\varepsilon^2/2 - \varepsilon^3/3)^{-1} \ln n)$
- …that preserves all inter-point distances up to a factor of $(1+\varepsilon)$

کاربرد لم JL

- می توانیم به صورت تصادفی جهت هایی را انتخاب کنیم و روی آنها تصویر عمودی نقاط را به دست بیاوریم.
  - با احتمال بیشتر از $1-1/n$ جواب می دهد.
- لم JL به ما می گوید فاصله ی زوج نقاط در مجموعه کوچکی از نقاط (نمایی بر حسب تعداد ابعاد) می تواند در ابعاد پایین حفظ شود.
- لم JL نمی گوید که خود نقاط هم حفظ می شوند، فقط فاصله ی زوج نقاط حفظ می شود.

تقریب همسایه نزدیک

- با ضریب تقریب c همسایگی به شعاع r حول یک نقطه
- ساختمان داده ای بدهید که برای هر نقطه کوئری q
  - اگر نقطه ای مثل p هست که ||p-q|| <= r باشد، نقطه ی p’ را برگرداند که ||p’-q|| <= c.r



کاهش از مساله های دیگر

- نزدیک ترین همسایه c-تقریبی به همسایه نزدیک c-تقریبی
  - با اضافه شدن یک $\log$
- با پیدا کردن همه ی همسایه های نزدیک می توان نزدیک ترین همسایه را دقیق حل کرد.
- کاربردها:
  - درخت پوشای کمینه c-تقریبی
  - خوشه بندی
  - ...

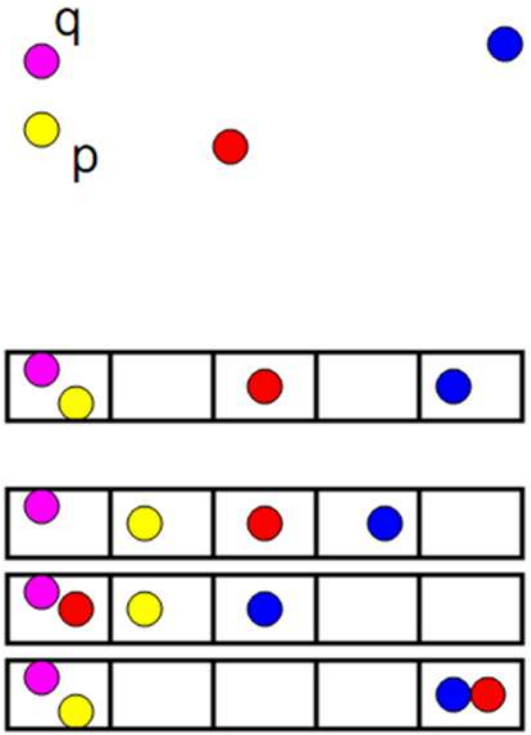
نتایج همسایه نزدیک تقریبی

	Query time	Space used	Preprocessing time
Vornoi	$O(2^d \log n)$	$O(n^{d/2})$	$O(n^{d/2})$
Kd-tree	$O(2^d \log n)$	$O(n)$	$O(n \log n)$
LSH	$O(n^{\rho} \log n)$	$O(n^{1+\rho})$	$O(n^{1+\rho} \log n)$

کارهای گذشته

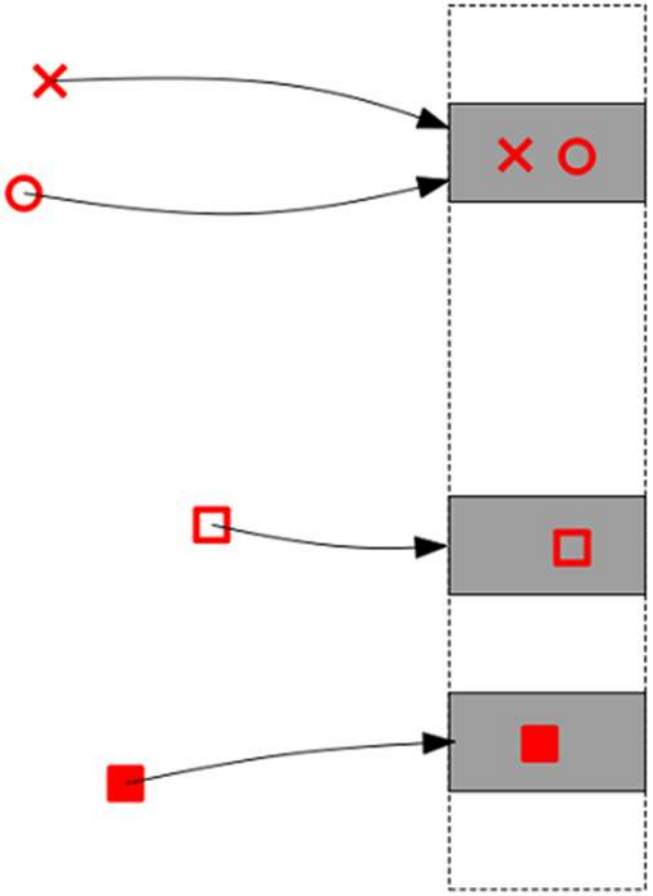
Space	Time	Comment	Norm	Ref
$dn+n^{4/\varepsilon^2}$	$d * \log n /\varepsilon^2$ or 1	$c=1+\varepsilon$	Hamm, l_2	[KOR'98, IM'98]
$n^{\Omega(1/\varepsilon^2)}$	$O(1)$			[AIP'0?]
$dn+n^{1+\rho(c)}$	$dn^{\rho(c)}$	$\rho(c)=1/c$	Hamm, l_2	[IM'98], [Cha'02]
		$\rho(c)<1/c$	l_2	[DIIM'04]
$dn * \log s$	$dn^{\sigma(c)}$	$\sigma(c)=O(\log c/c)$	Hamm, l_2	[Ind'01]
		$\sigma(c)=O(1/c)$	l_2	[Pan'06]
$dn+n^{1+\rho(c)}$	$dn^{\rho(c)}$	$\rho(c)=1/c^2 + o(1)$	l_2	[AI'06]
$dn * \log s$	$dn^{\sigma(c)}$	$\sigma(c)=O(1/c^2)$	l_2	[AI'06]

هشینگ حساس به محل (LSH)



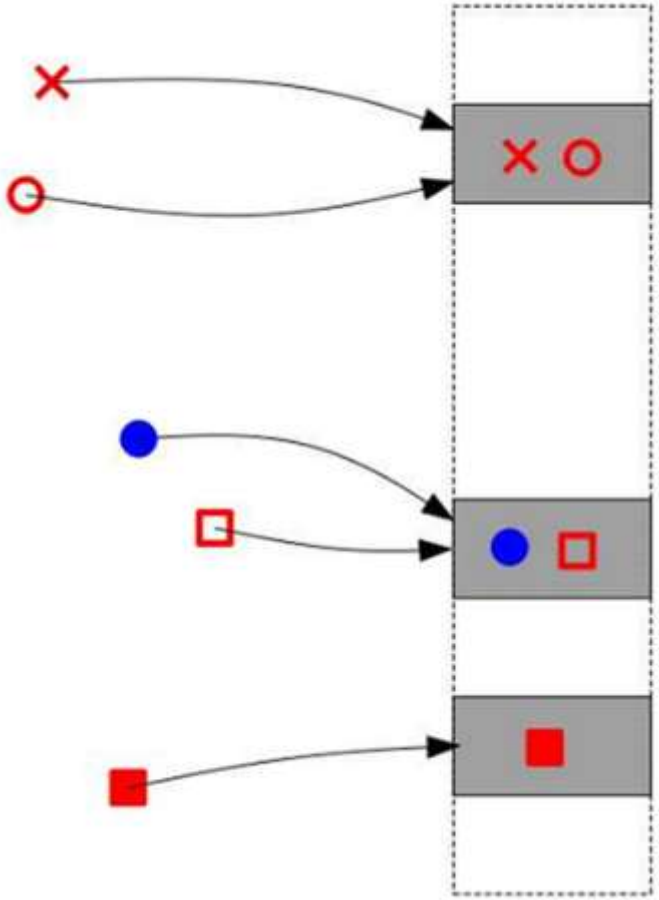
- ایده: توابع هش $g: R^d \rightarrow U$ را برای هر زوج نقاط p و q تعریف می کنیم:
  - اگر $||p-q|| <= r$ آنگاه $Pr(g(p)=g(q))$ خیلی کوچک نباشد.
  - اگر $||p-q|| > cr$ آنگاه $Pr(g(p)=g(q))$ کوچک باشد.
- در این صورت با هش کردن مساله را می توان حل کرد.

تصویر کلی عملکرد LSH



- پیش پردازش
- خانواده ای از توابع هش که نقاط نزدیک به یک سطل و نقاط دور به سطل های مجزا بروند.
- یک تابع تصادفی انتخاب کن و نقاط ورودی را هش کن.
- فقط سطل های ناتهی را نگهداری کن.

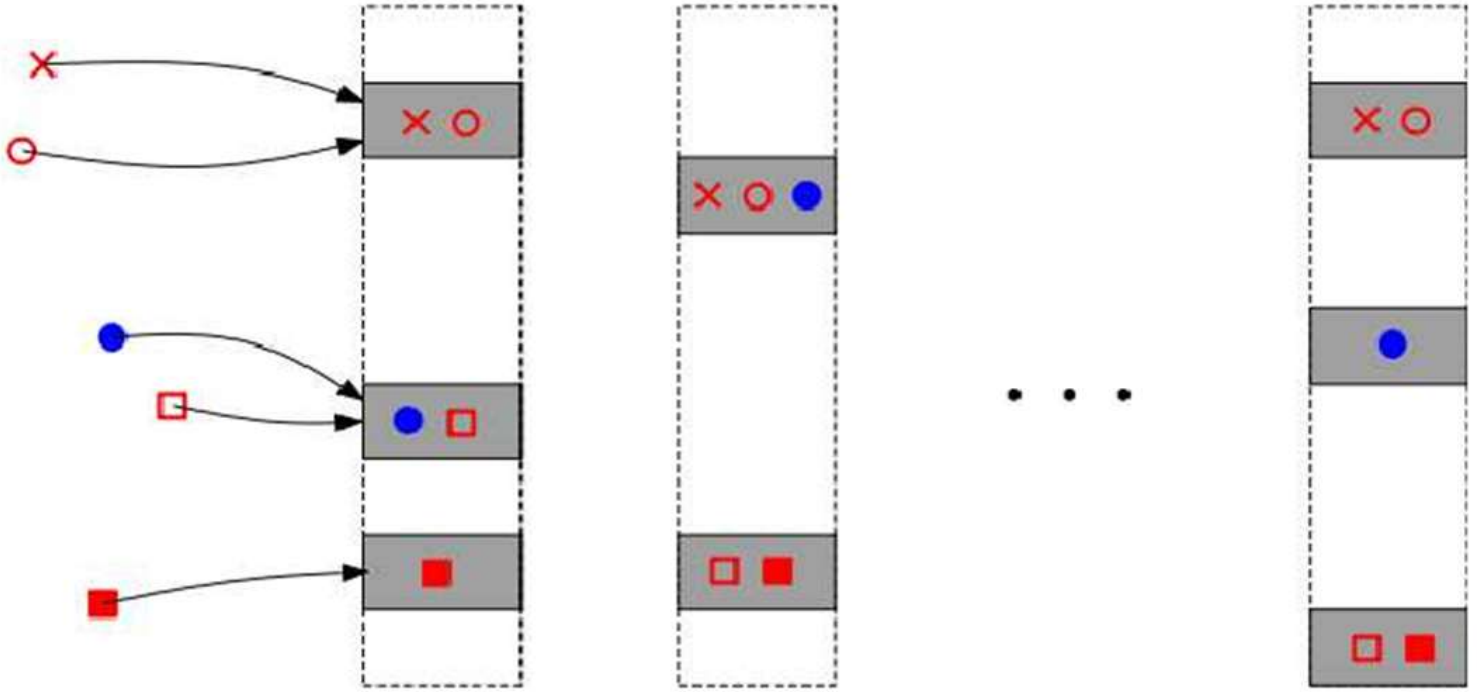
تصویر کلی عملکرد LSH



- کوئری
- سطل متناظر هر نقطه q را برای همسایه تقریبی نزدیکش بگرد.
- ایراد: ممکن است سطل q خالی باشد.

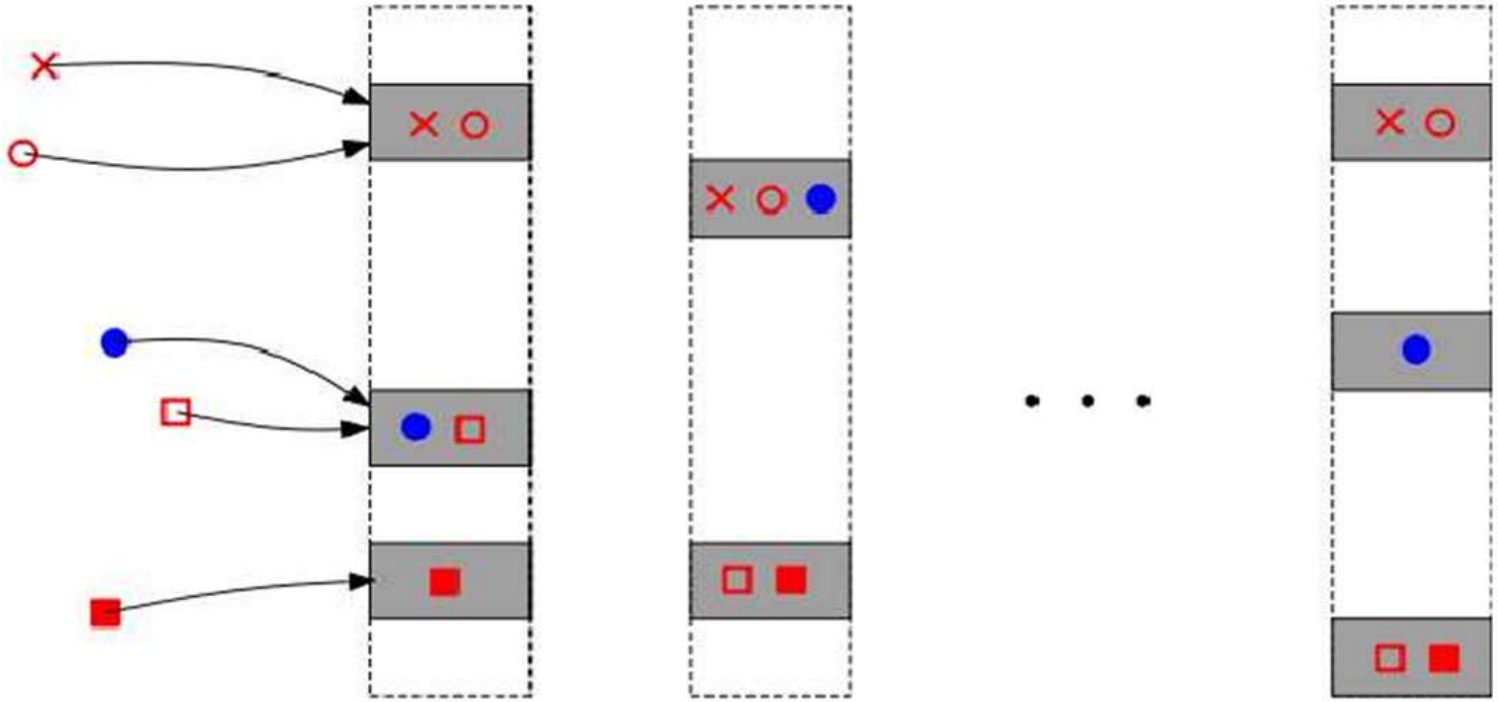
تصویر کلی عملکرد LSH

- راه حل: از چند جدول هش استفاده کن.
- اگر هر همسایگی نزدیک تقریبی پیدا شد، کار تمام است.



تصویر کلی عملکرد LSH

- ایراد: دقت کافی ندارد، چون کاندیداهای زیادی پیدا می شوند.
- راه حل: بعد از رسیدن به احتمال کوچک مشخصی متوقف شود.



تصویر کلی عملکرد LSH

- پیدا کردن تابع هش

$$\text{If } u \in B(q,r) \text{ then } \Pr[h(u)=h(q)] \geq \alpha$$
$$\text{If } u \notin B(q,R) \text{ then } \Pr[h(u)=h(q)] \leq \beta$$
$$r < R, \quad \alpha >> \beta$$

- که در آن
  - h به صورت تصادفی از یک خانواده تابع هش انتخاب شده است.
  - R به صورت روبرو به دست آمده است:

$$R = r(1+\varepsilon)$$

خانواده توابع هش

- A family H of functions $h: R^d \rightarrow U$ is called (P_1,P_2,r,cr)-sensitive, if for any p,q:
  - if $||p-q|| < r$ then $Pr[h(p)=h(q)] > P_1$
  - if $||p-q|| > cr$ then $Pr[h(p)=h(q)] < P_2$

- Example: Hamming distance
  - LSH functions: $h(p)=p_i$, i.e., the i-th bit of p
  - Probabilities: $Pr[h(p)=h(q)] = 1-D(p,q)/d$

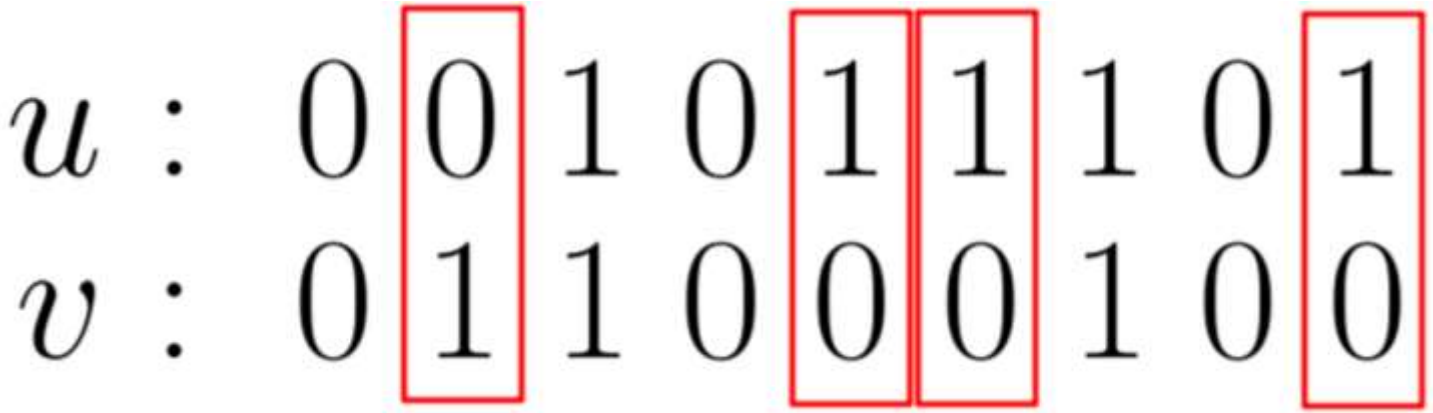
p=10010010
q=11010110

الگوریتم LSH

- توابعی به شکل $g(p) = <h1(p),h2(p),...,hk(p)>$
- پیش پردازش:
  - $g1,...,gL$ را انتخاب کنید.
  - به ازای هر نقطه p، آن را به سطلهای $g1(p),...,gL(p)$ هش کنید.
- کوئری:
  - از سطلهای $g1(p),g2(p),...$ نقاط را بازیابی کنید تا
    - همه ی نقاط از L سطل بازیابی شده باشند.
    - تعداد نقاط بازیابی شده $2L$ شود.
  - زمان کوئری به تعداد نقاط بازیابی شده بستگی دارد.
- زمان کل $O(dL)$

فاصله همینگ و مکعب d-بعدی یکه

- نقاط درون $H^d=\{0,1\}^d$ قرار می گیرند.
- هر نقطه یک رشته باینری است.
- فاصله همینگ (r): تعداد مختصات متفاوت



توابع هش ابرمکعب یکه

- **Define family *F*:**

$$\text{Given}: \text{Hypercube } H^d, \text{ point } b = (b_1, \ldots, b_d)$$

$$h \in F : \left\{ h_i(b) = b_i \,\middle|\, b = (b_1, \ldots, b_d) \in H^d, \text{ for } i = 1, \ldots, d \right\}$$

$$\alpha = 1 - \frac{r}{d}, \quad \beta = 1 - \frac{r(1+\varepsilon)}{d}$$

- Intuition: compare a random coordinate
- Called: $(r, r(1+\varepsilon), \alpha, \beta)$-sensitive family

توابع هش ابرمکعب یکه

- **Define family *G*:**

$$\text{Given}: b \in H^d, F$$

$$g \in G:$$

$$\left\{ g:\{0,1\}^d \to \{0,1\}^k \,\middle|\, g(b)=\left(h^1(b),\ldots,h^k(b) \right), \text{for } h^i \in F \right\}$$

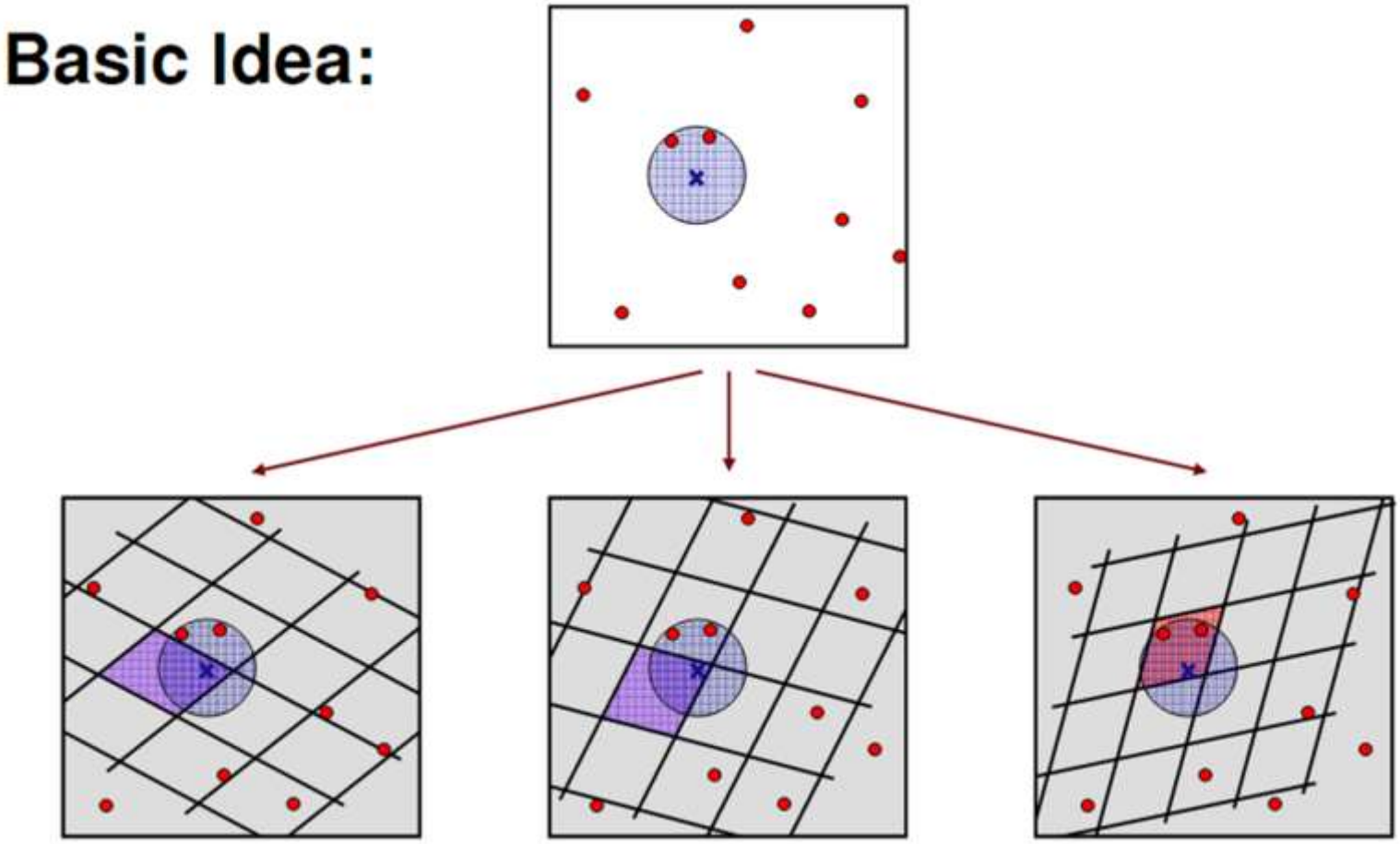
$$\alpha' = \left(1-\frac{r}{d}\right)^k = \alpha^k, \quad \beta' = \left(1-\frac{r(1+\varepsilon)}{d}\right)^k = \beta^k$$

- Intuition: Compare *k* random coordinates
- Choose *k* later – logarithmic in *n* J-L lemma

تحليل

- LSH solves c-approximate NN with:
  - Number of hash fun: $L=n^{\rho}$, $\rho=\log(1/P1)/\log(1/P2)$
  - E.g., for the Hamming distance we have $\rho=1/c$
  - Constant success probability per query q
- Questions:
  - Can we extend this beyond Hamming distance ?
    - Yes:
      - embed l_2 into l_1 (random projections)
      - l_1 into Hamming (discretization)
  - Can we reduce the exponent ρ ?

LSH با روش تصویر کردن تصادفی

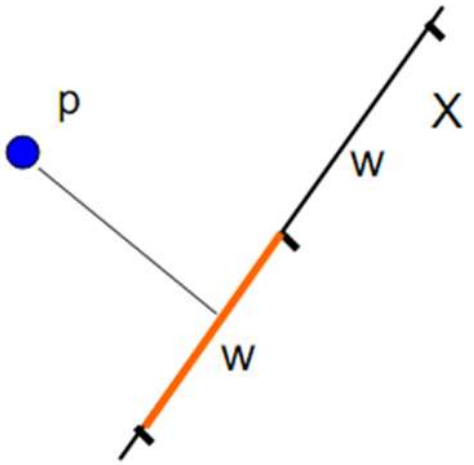


تصویر کردن تصادفی

- از چه توزیع هایی باید برای انتخاب بردارهای تصویر کردن استفاده کنیم؟
- اندازه سطل مناسب چیست؟
- حساسیت به محل
  - چه تعداد خط در هر توری باشد؟
  - چه تعداد خانه در کل باشد؟
  - به حساسیت بستگی دارد.
- کارایی این روش چقدر است؟

LSH بر مبنای تصویر کردن

- Define $h_{X,b}(p)=\lfloor (p*X+b)/w \rfloor$:
  - $w \approx r$
  - $X=(X_1 \ldots X_d)$, where X_i is chosen from:
    - Gaussian distribution (for l_2 norm)
    - "s-stable" distribution* (for l_s norm)
  - b is a scalar
- Similar to the $l_2 \rightarrow l_1 \rightarrow$Hamming route



* I.e., $p*X$ has same distribution as $||p||_s$ Z, where Z is s-stable

توزیع های p-stable

A prob. distribution D is called p-stable $:\Leftrightarrow$

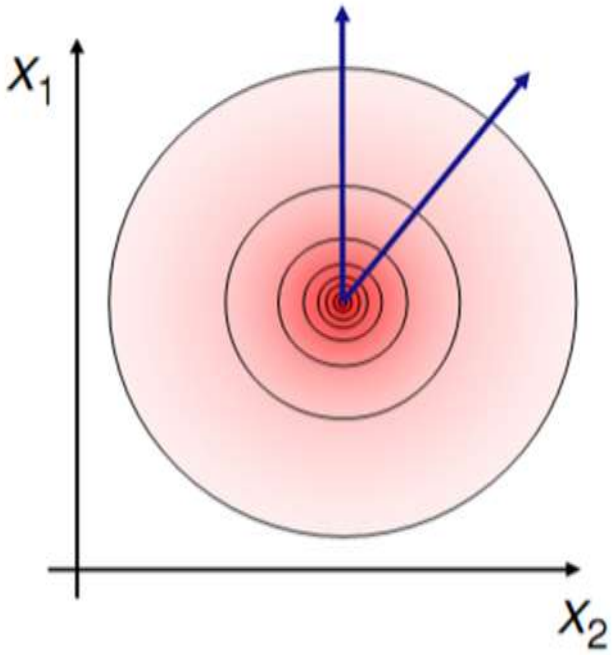
- For any $v_1, \ldots, v_n \in \mathbb{R}$
- And i.i.d. random variables $X_1, \ldots, X_n \sim D$

$\sum_i v_i X_i$ has the same distribution as $\left[\sum_i |v_i|^p\right]^{1/p} X$

where $X \sim D$

توزیع گاوسی

- توزیع نرمال 2-stable است.



الگوریتم تصویر کردن

- Chose p according to metric of the space l_p
- Compute vector with entries according to a p-stable distribution
  [for example: Gaussian noise entries]
- Each vector v_i yields a hash function h_i
- Compute: $h_i(x) = \left\lfloor \frac{\langle v_i, x \rangle + b}{r} \right\rfloor$

  b: random value $\in [0 \ldots r]$

  r: bucket size

محاسبه حساسیت به محل برای تصویر کردن

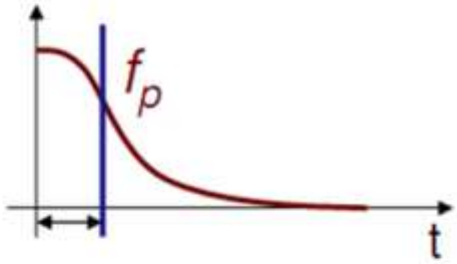
Computing the Locality "Sensitivity"

Distance $c = ||v_1 - v_2||_p$

cX-distributed, X from p-stable distr.

$$\left[h_i(x) = \left\lfloor \frac{\langle v_i, x \rangle + b}{r} \right\rfloor \right]$$

$$\underbrace{\Pr(collision)}_{=:p(c)} = \int_0^r \frac{1}{c} \underbrace{f_p\left(\frac{t}{c}\right)}_{abs.\ density} \underbrace{\left(1 - \frac{t}{r}\right)}_{hit\ bucket} dt$$

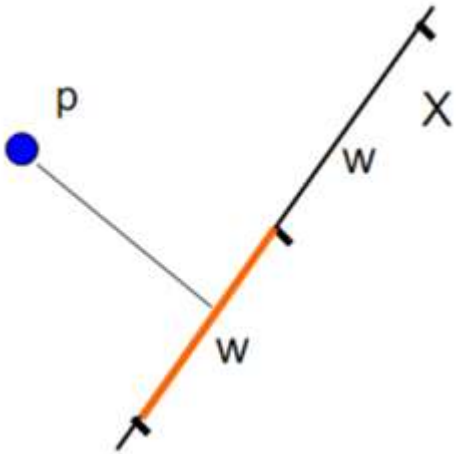


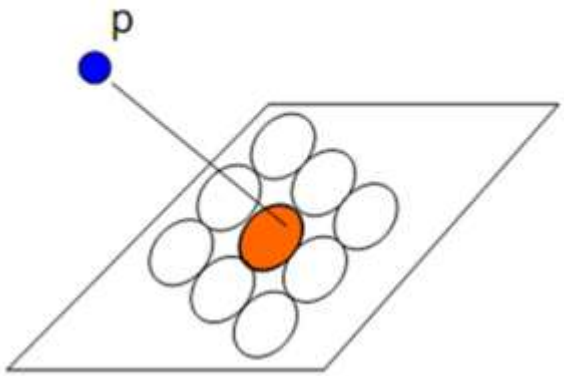
The constructed family of hash functions is $(r_1, r_2, \alpha, \beta)$-sensitive for

$$\alpha = p(1),\ \beta = p(c),\ r_2/r_1 = c$$

روش جدید LSH

- Instead of projecting onto R^1, project onto R^t, for constant t
- Intervals → lattice of balls
  - Can hit empty space, so hash until a ball is hit
- Analysis:
  - $\rho = 1/c^2 + O(\log t / t^{1/2})$
  - Time to hash is $t^{O(t)}$
  - Total query time: $dn^{1/c^2+o(1)}$
- [Motwani-Naor-Panigrahy'06]: LSH in l_2 must have $\rho \geq 0.45/c^2$





MinHash

- =Min-wise independent permutations
- Let h be a hash function that maps the members of A and B to distinct integers, and for any set S define hmin(S) to be the member x of S with the minimum value of h(x).
- Then hmin(A) = hmin(B) exactly when the minimum hash value of the union $A \cup B$ lies in the intersection $A \cap B$. Therefore,
- Pr[hmin(A) = hmin(B)] = J(A,B).

Min-Hashing

- Consider
  - $S_A, S_B \subset U$
  - Pick – random permutation π of U
  - Define $\alpha = \pi^{-1}(\min\{\pi(S_A)\})$ and $\beta = \pi^{-1}(\min\{\pi(S_B)\})$
  - Meaning? – minimal element under permutation π
- Lemma:

$$P[\alpha = \beta] = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$$

  - Let $\delta = \min\{ \pi(S_A \cup S_B) \}$
  - Claim: $\alpha = \beta \Leftrightarrow \pi^{-1}(\delta) \in S_A \cap S_B$
  - Clearly

$$P[\pi^{-1}(\delta) \in S_A \cap S_B] = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$$

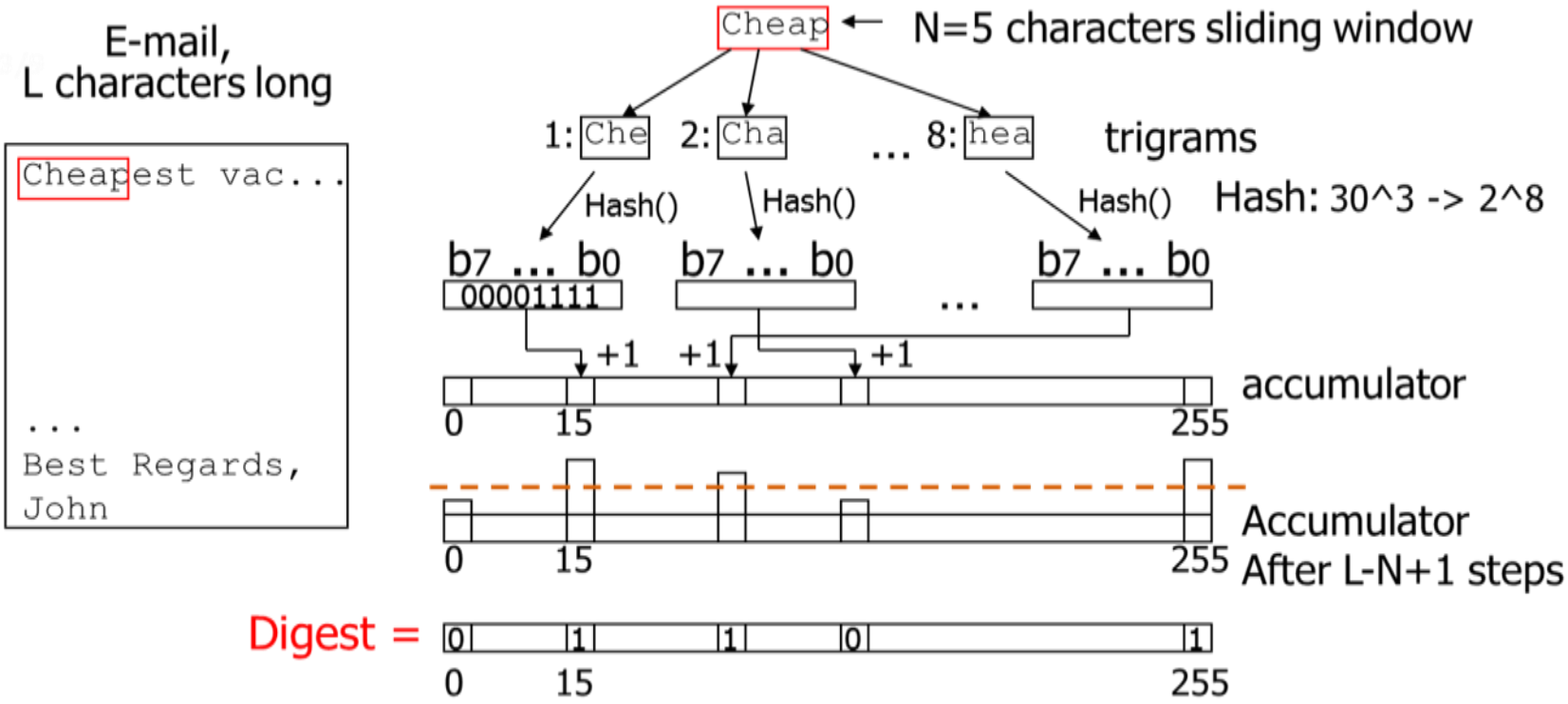
Min-Wise Indep Permutations

- Problem
  - Truly-random π over U = [0 … N-1] is infeasible
  - But – do we really need true randomness?
- Solution
  - Poly-size family of permutations F⊂SN over U
  - Choosing/representing random π∈F is easy
- Min-Wise Independence (MWI) Property:
  - For all sets X⊂U, for all x∈F,

$$P_{\pi \in F}\left[\min(\pi(X)) = x\right] = \frac{1}{|X|}$$

Nilsimsa similarity hashing

- Definition: Nilsimsa Compare Value (NCV) between two digests is equal to the number of bits at corresponding positions that are equal, minus 128.



Methods

- Bit sampling for Hamming distance
- Min-wise independent permutations
  - Jaccard index
- Nilsimsa Hash
- Random projection
  - Cosine distance
- Stable distributions

Conclusion

- Hamming: 1 dimension from d dimensions
- Jaccard: k dimensions from d dimensions
  - Binary values only
- Random Projection (Cosine): k dimensions (optional)
- Nilsimsa: k consecutive dimensions
  - Alphabet
- Stable Distributions: k dimensions