Hash Bit Selection: a Unified Solution for Selection Problems in Hashing

Xianglong Liu[†] Junfeng He[‡*] Bo Lang[†] Shih-Fu Chang[‡]
[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡]Department of Electrical Engineering, Columbia University, New York, NY, USA
[*]Facebook, 1601 Willow Rd, Menlo Park, CA, USA
{xlliu, langbo}@nlsde.buaa.edu.cn jh2700@columbia.edu sfchang@ee.columbia.edu

1. Proof of Theorem 1

To prove the theorem in the paper, we first prove the following proposition and lemmas.

The definition of the normalized dominant set can be characterized and represented in terms of determinants. For $S \subseteq V$, we denote by A_S the submatrix of A formed by the rows and the columns indexed by the elements of S. Then we define the matrix B_S:

$$B_S = \begin{pmatrix} 0 & (\pi^{-1})^T \\ \pi^{-1} & A_S \end{pmatrix}, \tag{1}$$

and the matrix ${}^jB_S$:

$${}^jB_S = \begin{pmatrix} 0 & & (\pi^{-1})^T & \\ \pi^{-1} & A_S^1 \ \ldots & A_S^{j-1} \ \mathbf{0} \ A_S^{j+1} & \ldots \ A_S^m \end{pmatrix}, \tag{2}$$

where $S = \{i_1, \ldots, i_m\}$ with $i_1 < \ldots < i_m$ and A_S^i is the i-th column of A_S.

Proposition 1 *Let $S = \{i_1, \ldots, i_m\} \subseteq V$ be a nonempty subset of vertices and assume $i_1 < \ldots < i_m$ without loss of generality. Then for any $i_h \in S$,*

$$w_S(i_h) = (-1)^m \det({}^hB_S), \tag{3}$$

and,

$$W_S = (-1)^m \det(B_S). \tag{4}$$

Proof First we prove (3) holds for $m \geq 1$.

1. For $m = 1$,

$${}^1B_{\{i\}} = \begin{pmatrix} 0 & \pi_i^{-1} \\ \pi_i^{-1} & 0 \end{pmatrix},$$

It is easy to verify that $w_{\{i\}}(i) = \frac{1}{\pi_i^2} = -\det({}^1B_{\{i\}})$.

2. For $m > 1$ and $S = \{i_1, \ldots, i_m\}$ with $i_1 < \ldots < i_m$,

$$\begin{aligned}
\det({}^hB_S) &= \begin{vmatrix} 0 & \pi_{i_1}^{-1} & \cdots & \pi_{i_h}^{-1} & \cdots & \pi_{i_m}^{-1} \\ \pi_{i_1}^{-1} & a_{i_1i_1} & \cdots & 0 & \cdots & a_{i_1i_m} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \pi_{i_h}^{-1} & a_{i_hi_1} & \cdots & 0 & \cdots & a_{i_hi_m} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \pi_{i_m}^{-1} & a_{i_mi_1} & \cdots & 0 & \cdots & a_{i_mi_m} \end{vmatrix} \\
&= \frac{(-1)^{h+2}}{\pi_{i_h}} \begin{vmatrix} \pi_{i_1}^{-1} & a_{i_1i_1} & \cdots & a_{i_1i_m} \\ \vdots & \vdots & \cdots & \vdots \\ \pi_{i_h}^{-1} & a_{i_hi_1} & \cdots & a_{i_hi_m} \\ \vdots & \vdots & \cdots & \vdots \\ \pi_{i_m}^{-1} & a_{i_mi_1} & \cdots & a_{i_mi_m} \end{vmatrix} \\
&= -\frac{1}{\pi_{i_h}}[\sum_{i_j \in S\setminus\{i_h\}} \pi_{i_j} a_{i_hi_j} \det({}^jB_{S\setminus\{i_h\}}) \\
&\quad + \frac{1}{\pi_{i_h}} \det(A_{S\setminus\{i_h\}})]
\end{aligned}$$

From the fact that

$$\begin{vmatrix} (\pi^{-1})^T\mathbf{1} & (A_S\mathbf{1})^T \\ \pi^{-1} & A_S \end{vmatrix} = 0,$$

we can obtain:

$$\sum_{i_j \in S} \frac{1}{\pi_{i_j}} \det(A_S) + \sum_{i_j \in S} (\sum_{i_k \in S} a_{i_ji_k}) \pi_{i_j} \det({}^jB_S) = 0,$$

namely

$$\det(A_S) = -\pi_{i_h} \sum_{i_j \in S} \pi_{i_j} f(S, i_j|i_h) \det({}^jB_S).$$

Therefore, we can rewrite $\det({}^hB_S)$ as

$$\begin{aligned}
&-\sum_{i_j \in S\setminus\{i_h\}} \frac{\pi_{i_j}}{\pi_{i_h}} (a_{i_hi_j} - f(S\setminus\{i_h\}, i_j|i_h)) \det({}^jB_{S\setminus\{i_h\}}) \\
&= -\sum_{i_j \in S\setminus\{i_h\}} \phi_{S\setminus\{i_h\}}(i_j, i_h) \det({}^jB_{S\setminus\{i_h\}}).
\end{aligned}$$

According to the recursive definition of $w_S(i_h)$, we can conclude that (3) holds for $m \geq 1$.

For (4), since $\det(B_S) = \sum_{i_h \in S} \det({}^hB_S)$, then $W(S) = (-1)^m \det(B_S)$. □

When using the following fact

$$\begin{vmatrix} \pi_h^{-1} & (A_S^h)^T \\ \pi^{-1} & A_S \end{vmatrix} = 0,$$

we give an alternative way to compute $w_S(i)$:

$$w_S(i) = \sum_{j \in S \backslash \{i\}} \frac{1}{\pi_i^2} (\pi_i \pi_j a_{ij} - \pi_h \pi_j a_{hj}) w_{S \backslash \{i\}}(j) \quad (5)$$

with h as an arbitrary element of $S \backslash \{i\}$, and $|S| > 1$.

Lemma 1 *With $\hat{A} = \Pi A \Pi$, where $\Pi = diag(\pi)$, the KKT equality conditions in (7) in the paper hold, if and only if*

$$B_\sigma [-\lambda, \pi_{i_1} x_{i_1}, \ldots, \pi_{i_h} x_{i_h}]^T = [1, 0, \ldots, 0]^T, \quad (6)$$

where λ is a real constant number, $\sigma = \sigma(\mathbf{x}) = \{i_1, \ldots, i_h\}$ with $i_1 < \ldots < i_h$, and the matrix B_σ:

$$B_\sigma = \begin{pmatrix} 0 & \pi_{i_1}^{-1} & \ldots & \pi_{i_h}^{-1} \\ \pi_{i_1}^{-1} & a_{i_1 i_1} & \ldots & a_{i_1 i_h} \\ \vdots & \vdots & \ldots & \vdots \\ \pi_{i_h}^{-1} & a_{i_h i_1} & \ldots & a_{i_h i_h} \end{pmatrix}.$$

Proof $\mathbf{x} \in \Delta$ satisfies the Karush-Kuhn-Tucker (KKT) conditions for problem (7), if there exist $n+1$ real constants (Lagrange multipliers) $\mu_1, \ldots, \mu_L$ and λ, with $\mu_i \geq 0$ for all $i = 1 \ldots L$, such that for all $i = 1 \ldots L$:

$$\begin{aligned} (\hat{A}\mathbf{x})_i - \lambda + \mu_i &= 0; \\ x_i \mu_i &= 0. \end{aligned}$$

Because of the nonnegativity of both x_i and μ_i, it can be restated as follows:

$$(\hat{A}\mathbf{x})_i \begin{cases} = \lambda, & \text{if } i \in \sigma(x); \\ \leq \lambda, & \text{otherwise} \end{cases}$$

with some real constant $\lambda = x^T \hat{A} x$.

For $\sigma = \sigma(\mathbf{x}) = \{i_1, \ldots, i_m\}$ with $i_1 < \ldots < i_m$.

$$B_\sigma = \begin{pmatrix} 0 & \pi_{i_1}^{-1} & \ldots & \pi_{i_m}^{-1} \\ \pi_{i_1}^{-1} & & & \\ \vdots & & A_\sigma & \\ \pi_{i_m}^{-1} & & & \end{pmatrix}.$$

Then $B_\sigma [-\lambda, \pi_{i_1} x_{i_1}, \ldots, \pi_{i_m} x_{i_m}]^T = [1, 0, \ldots, 0]^T$ is equivalent to:

$$\begin{cases} \sum_{h=1}^m x_{i_h} &= 1; \\ A_\sigma [\pi_{i_1} x_{i_1}, \ldots, \pi_{i_m} x_{i_m}]^T &= \lambda [\pi_{i_1}^{-1}, \ldots, \pi_{i_m}^{-1}] \end{cases}$$

namely, using $\Pi = \text{diag}(\pi)$

$$\begin{cases} \mathbf{1}^T \mathbf{x} &= 1; \\ (\Pi A \Pi \mathbf{x})_i &= \lambda, \ i \in \sigma(x) \end{cases}$$

By setting $\hat{A} = \Pi A \Pi$, we prove Lemma 1. □

Lemma 2 *Let $\sigma = \sigma(\mathbf{x})$ be the support of a vector $\mathbf{x} \in \Delta$. Then, $\mathbf{x}$ satisfies the KKT equality conditions in (7) in the paper if and only if*

$$x_i = \begin{cases} \frac{w_\sigma(i)}{W(\sigma)}, & \text{if } i \in \sigma; \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

Moreover,

$$\frac{w_{\sigma \cup \{j\}}(j)}{W(\sigma)} = \frac{1}{\pi_j^2} [(\hat{A}\mathbf{x})_j - (\hat{A}\mathbf{x})_i] = -\frac{1}{\pi_j^2} \mu_j \quad (8)$$

for all $i \in \sigma$ and $j \notin \sigma$, where the μ_j are the (nonnegative) Lagrange multipliers of program (7).

Proof For (6) which is equivalent to the KKT conditions $\hat{A}\mathbf{x})_i = \lambda$, if $i \in \sigma(x)$, it can be treated as a linear equation problem with unknowns λ and $x_i, i \in \sigma$. Since $\det(B_\sigma) \neq 0$, the problem has an unique solution whose support denoted by $\sigma = \{i_1, \ldots, i_m\}$ without loss of generality. Using Cramer's rule, we can get

$$\pi_{i_h} x_{i_h} = \frac{\pi_{i_h} \det({}^hB_\sigma)}{\det(B_\sigma)},$$

Then according to Lemma 1, we have

$$x_{i_h} = \frac{(-1)^m w_\sigma(i_h)}{(-1)^m W(\sigma)} = \frac{w_\sigma(i_h)}{W(\sigma)}.$$

for any $1 \leq h \leq m$. Therefore, $x = x^\sigma$.

Using Equation (5) in the paper, we obtain:

$$\begin{aligned} \frac{w_{\sigma \cup \{j\}}(j)}{W(\sigma)} &= \frac{\sum_{i_h \in \sigma} \frac{1}{\pi_j^2} (\pi_j \pi_{i_h} a_{j i_h} - \pi_{i_k} \pi_{i_h} a_{i_k i_h}) w_\sigma(i_h)}{W(\sigma)} \\ &= \frac{1}{\pi_j^2} \sum_{i_h \in \sigma} (\pi_j \pi_{i_h} a_{j i_h} - \pi_{i_k} \pi_{i_h} a_{i_k i_h}) x_{i_h}^\sigma \\ &= \frac{1}{\pi_j^2} [(\hat{A}x)_j - (\hat{A}x)_{i_k}]. \end{aligned}$$

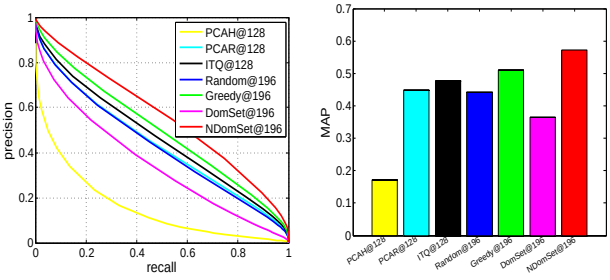
We have the fact $(\hat{A}x)_j - (\hat{A}x)_i = -\mu_j$ for all $i \in \sigma$ and $j \notin \sigma$, and $\pi_j > 0$ for all j. Then we can conclude that

$$\frac{w_{\sigma \cup \{j\}}(j)}{W(\sigma)} = \frac{1}{\pi_j^2} [(\hat{A}x)_j - (\hat{A}x)_i] = -\frac{1}{\pi_j^2} \mu_j \leq 0.$$

for all $i \in \sigma$ and $j \notin \sigma$, where the μ_j are the (nonnegative) Lagrange multipliers of quadratic programming problem in the paper. □

Table 1. MAP (%) of bit selection over different hashing algorithms using 32 - 128 bits on GIST-1M.

500 BITS		32 BITS	64 BITS	128 BITS
RMMH	LRH [6]	5.85	10.86	16.78
	RANDOM	5.81	10.75	16.53
	GREEDY	5.74	9.56	15.65
	DOMSET	3.78	7.72	13.97
	NDOMSET	**7.06**	**11.78**	**17.56**



(a) PR (b) MAP

Figure 1. Performance comparison of bit selection methods over multiple hashing methods on SIFT-1M.

Theorem 1 *If* $\mathbf{x}^*$ *is a strict local solution of program (7) with* $\hat{A} = \Pi A \Pi$*, where* $\Pi = diag(\pi)$*, then its support* $\sigma = \sigma(\mathbf{x})$ *is the normalized dominant set of graph* $G = (V, E, A, \pi)$*, provided that* $w_{\sigma\cup\{i\}}(i) \neq 0$ *for all* $i \notin \sigma$*.*

Proof Using Lemma 1 and 2, the proof can be completed following that of [7]. □

2. More experimental results

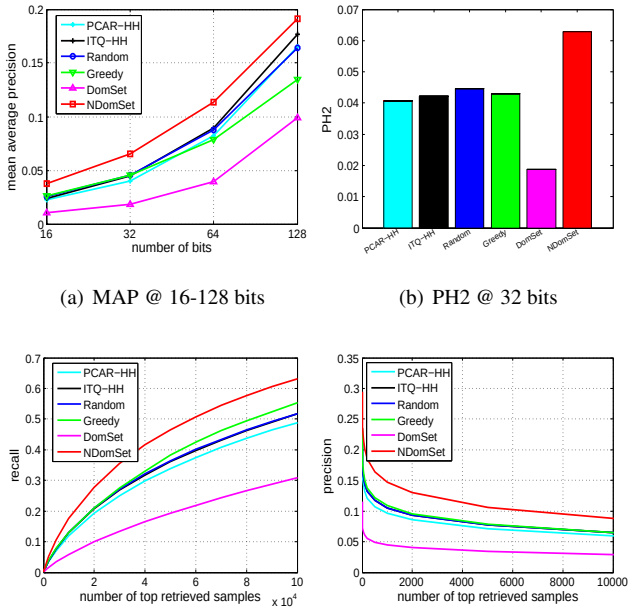
2.1. Bit selection over basic hashing method

Baselines. We still use the learned reconfigurable hashing (**LRH**) [6] and other naive selection methods as our baselines. However, here we report more results over the state-of-the-art hashing method RMMH [3].

The results are shown in Table 1. As we can observe, the proposed selection (NDomset) attains the best performance in all cases in terms of MAP with 32, 64 and 128 bits. Furthermore, RMMH generates each hash bit independently, and thus randomly selecting l bits acts like generating l bits using RMMH directly. By comparing results using Random and other bit selection methods, we can conclude that our bit selection method improves the performance of RMMH most. Although LRH can obtain performance gains when using long hash bits, it fails to compete with NDomSet in terms of both accuracy and speeds (Roughly, NDomSet is more than five times faster than it).

2.2. Bit selection over multiple hashing methods

Baselines. We consider the scenario using multiple hashing methods, where our baselines include some hashing methods with the longest codes they can generate.



(a) MAP @ 16-128 bits (b) PH2 @ 32 bits

(c) R @ 32 bits (d) P @ 32 bits

Figure 2. Performance comparison of bit selection methods over multiple bit hashing on GIST-1M.

The scenario derives from the fact that a number of hashing methods can only generate hash codes of limited length, due to the limited number of the feature dimension [1, 8] or the landmark number [5]. Using multiple hashing methods, we can generate any desired number of hash bits, and meanwhile our bit selection can select the most desired ones from the pool of bits generated by multiple hashing methods. With 128-D SIFT features [2], hashing methods like PCAH, PCAR and ITQ can generate at most 128 bits. Therefore to use longer hash codes, say 196 in our experiment, we build a large bit pool with 384 bits, of which each 128 bits are generated respectively by PCAH, PCAR, and ITQ [1]. Then we compare all selection methods picking 196 bits and basic hashing methods with 128 bits.

Figure 1 show the comparison performances in terms of P-R curves and MAP. As we can see, bit selection methods like Greedy and NDomSet using 196 bits outperform the three basic hashing methods using their longest hash codes individually (*i.e.*, 128 bits), and moreover they achieve better performances than the native selection methods including Random and DomSet using the same number of hash bits. This observation demonstrates the benefits of a good bit selection under this specific scenario. Note that our bit selection obtains the most significant performance gains in all cases.

2.3. Bit selection over multiple bit hashing

Baselines. We employ another double bit method using hierarchical hashing (HH) proposed in [4]. Similar to DB in

the paper, we generate bits by HH over PCAR (**PCAR-HH**) and ITQ (**ITQ-HH**).

A 500 bit pool is first built with 250 bits generated by PCAR-HH and the rest bits by ITQ-HH on GIST-1M. Figure 2 shows the results comparing PCAR-HH, ITQ-HH, and different bit selection methods. In Figure 2 (a) the MAP of NDomSet increases dramatically when using more bits, and is consistently superior to both the double bit hashing and selection baselines. Figure 2 (c)-(d) plot the recall, PH2 and P-R curves using 32 bits. In all cases, significant performance improvements are observed, which certainly supports our conclusion in the paper that our bit selection method can recognize the bits of good quality and further improve performances over multi-bit hashing algorithms.

References

[1] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, 2011. 3

[2] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 33(1):117–128, 2011. 3

[3] A. Joly and O. Buisson. Random Maximum Margin Hashing. In *CVPR*. IEEE, 2011. 3

[4] H. Liu and S. Yan. Robust graph mode seeking by graph shift. In *ICML*, 2010. 3

[5] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *ICML*, 2011. 3

[6] Y. Mu, X. Chen, X. Liu, T.-S. Chua, and S. Yan. Multimedia semantics-aware query-adaptive hashing with bits reconfigurability. *IJMIR*, pages 1–12, 2012. 3

[7] M. Pavan and M. Pelillo. Dominant sets and pairwise clustering. *IEEE TPAMI*, 29(1):167–172, 2007. 3

[8] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, 2010. 3