JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 21, No. 6 : 1113~1120 Jun. 2017

병렬 구조의 블룸필터 설계

장영달[1] · 김지홍[2*]

The Construction of A Parallel type Bloom Filter

Young-dal Jang[1] · Ji-hong Kim[2*]

Department of Information and Communication Engineering, Semyung University, Jecheon 27136, Korea

요 약

최근 정보통신 기술의 발달로 인하여 데이터의 양이 점차 증가하고 있으며, 이에 대한 처리와 관련된 연구가 활발히 진행되고 있다. 주어진 집합 내에 특정 개체의 존재여부를 알기위해 사용되고 있는 블룸필터는 데이터의 공간 활용에 매우 유용한 구조이다. 본 논문에서는 블룸필터의 긍정오류확률에 대한 요인분석과 함께, 긍정오류를 최소화시키기 위한 방안으로 병렬구조 방식의 블룸필터를 제안한다. 일반 블룸필터의 최소 긍정오류확률값을 가질 수 있도록 구현된 병렬 불룸필터 방식은 일반 블룸필터 크기의 메모리와 유사한 크기를 사용하지만, 해쉬함수별로 병렬 처리함으로서, 속도를 높일 수 있다는 장점을 가진다. 또한 완전 해쉬함수를 사용하는 경우에는 삽입뿐 아니라, 삭제도 가능하다는 장점을 가진다.

ABSTRACT

As the size of the data is getting larger and larger due to improvement of the telecommunication techniques, it would be main issues to develop and process the database. The bloom filter used to lookup a particular element under the given set is very useful structure because of the space efficiency. In this paper, we analyse the main factor of the false positive and propose the new parallel type bloom filter in order to minimize the false positive which is caused by other hash functions. The proposed method uses the memory as large as the conventional bloom filter use, but it can improve the processing speed using parallel processing. In addition, if we use the perfect hash function, the insertion and deletion function in the proposed bloom filter would be possible.

키워드 : 블룸필터, 긍정오류확률, 병렬블룸필터, 해쉬함수

Key word : Bloom filter, False Positive Rate, Parallel Bloom filter, Hash Function

Received 11 February 2017, **Revised** 21 February 2017, **Accepted** 19 April 2017

* **Corresponding Author** Ji-hong Kim(E-mail: jhkim@semyung.ac.kr, Tel:+82-43-649-1310)
Department of Information and Communication, Semyung University, Jecheon 27136, Korea

Open Access https://doi.org/10.6109/jkiice.2017.21.6.1113 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

시스템 내에 저장된 캐쉬 메모리, 라우팅 테이블 등의 데이터 증가로 인하여, 데이터의 존재여부를 확인할 수 있는 인덱스로서 블룸필터를 자주 사용한다. 블룸필터는 n 개의 입력요소에 대하여 k 개의 해시함수 결과값을 사이즈가 m 인 배열의 해당 비트에 '1'로 설정하는 것으로 데이터의 공간 활용에 매우 유용한 구조이다. 하지만 블룸필터의 특성상 존재하지 않는 데이터를 존재한다고 판단하는 긍정오류 확률(false positive rate) 발생할 수 있다. 반면에 부정오류 확률(false negative rate)은 존재하지 않는 다는 장점이 있다.

블룸필터[1]가 제안된 이래로, 블룸필터에 대한 긍정오류 확률에 대한 다양한 연구와 긍정오류 확률에 대한 최소값 및 하한경계에 대한 분석[1,2]이 이루어 졌다. 이후에도 블룸필터의 응용에 관한 연구로서, 일반화된 블룸필터, 압축 블룸필터[3], 스펙트럼 블룸필터, 캐쉬 메모리에 적용 가능한 카운팅 블룸필터[4,5], 메모리 최적화된 카운팅 블룸필터[6,7], 삭제 가능한 블룸필터[8-10] 등에 대한 연구가 이루어졌다.

분산시스템의 각종 응용에서 사용되고 있는 블룸필터는 입력요소에 대한 존재유무를 확인하기 위해 사용되며, 검색할 때 긍정오류가 발생할 수 있다. 본 논문에서는 블룸필터의 긍정오류에 대한 발생원인을 분석하고, 이를 최소화할 수 있는 병렬 구조의 블룸필터를 제안한다. 제안된 방식은 처리속도의 향상과 더불어 완전한 해쉬함수를 사용하는 경우에는 삭제기능도 부가할 수 있다. 본 논문은 1장은 서론, 2장 본론에서는 블룸필터의 개념, 긍정오류확률의 개념을 설명하고, 3장에서는 제안된 방식을 설명하고, 4장에서는 기존의 방식과 제안된 방식을 비교분석하고, 결론을 맺는다.

Ⅱ. 본 론

2.1. 블룸필터의 긍정오류 확률

2.1.1. 블룸필터 개념

블룸필터는 1970년 Burton Howard Bloom에 의해 고안된 것으로 원소가 집합에 속하는지 여부를 검사하는데 사용되는 확률적 자료 구조이다[1]. 그림 1과 같이 초기에 모두 "0"으로 설정된 m(=16) 비트의 블룸필터를 이용하여, 개체에 대한 해쉬함수 결과값에 따라 블룸필터의 내용이 "1"로 세트된다.

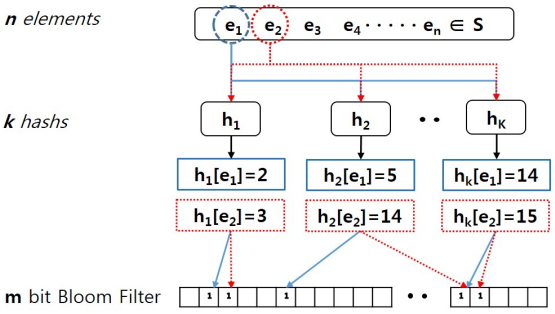

Fig. 1 The element Insertion process

예를 들어 개체 e1 에 대한 세 개의 해쉬함수 결과 값이 각각 2, 5, 14 이라면, 이에 해당되는 비트가 "1"로 설정된다. 마찬가지로 개체 e2에 대해서 3, 14, 15 번째 비트가 "1"로 설정된다. 이러한 블룸필터에는 개체 e1, e2가 존재함을 표시하고 있다. 블룸필터를 이용하여 개체의 존재여부를 확인하기 위한 룩업(look up) 과정에서 그림 2와 같이 정상적으로 입력된 e1 의 경우에는 실제 존재하는 개체이므로 정상적으로 판정된다. 반면에 입력되지 않은 개체 ex에 대해서는 실제로 존재하지 않지만, 블룸필터 룩업과정을 통하여 존재하는 것으로 표현되기 때문에 이를 긍정오류(false positive)라 한다.

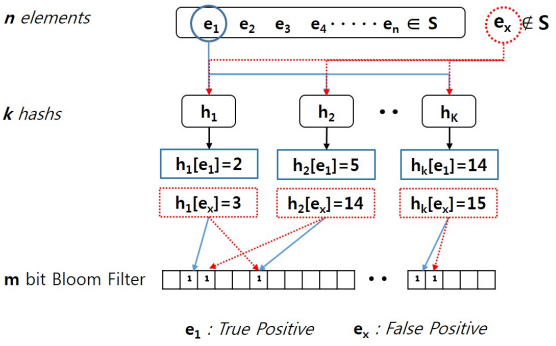

Fig. 2 The element look up process

2.1.2. 블룸필터의 긍정오류 확률

긍정오류란 긍정으로 판정된 것 중에서의 실제로 존재하지 않는 오류를 말한다. 이와 같이 블룸필터는 주

어진 공간을 이용하여 요소의 존재여부를 표현하기 때문에, 이러한 긍정오류가 발생할 수 있는 확률을 항시 가지고 있다. 그러나 이러한 긍정오류확률은 m, n, k을 적당한 값으로 조절하여 제어할 수 있다.

1970년 Bloom에 의해 제안된 이래, 지금까지 블룸필터의 긍정오류 확률 P_{fp_org}는 아래의 식으로 정의되어 지금까지도 많은 논문에서 이를 인용하고 있다.

$$P_{fp\_org} = [1-(1-\frac{1}{m})^{nk}]^k \approx (1-e^{-nk/m})^k \quad (1)$$

블룸필터는 초기에 모두 "0"으로 설정된 상태에서 해쉬함수의 결과 값에 의해 블룸필터를 설정한 것이다. 즉, n개의 입력요소에 대하여 존재유무를 확인할 수 있도록 k 개의 해쉬함수 결과값을 m 비트의 블룸필터의 각 비트에 세트시키고, 검색과정에서 이를 이용하는 방식이다.

위의 긍정오류 확률식의 내부 수식을 P_{nk}이라면, $P_{nk} = 1-(1-\frac{1}{m})^{nk}$은 nk번 시행에 대하여 임의의 비트가 "1"로 세트될 확률을 의미하며, 이를 다시 k개의 해쉬함수에 대해 적용하면,

$$P_{fp-org} = (1-(1-\frac{1}{m})^{nk})^k = (1-e^{-nk/m})^k$$

검색과정에서 특정 요소에 대한 k개의 해쉬함수의 결과값에 해당되는 모든 비트들이 "1"로 설정된 경우에는 존재한다고 판단한다. 그러나 실제로 존재하지 않는 것에 대해서 존재한다고 판단하는 긍정오류가 발생할 수 있다는 것을 의미한다.

긍정오류 확률을 최소화하기 위한 해쉬함수의 개수 k 값은 $k = \ln 2 \times (m/n)$으로 알려져 있으며, 이때 최소 긍정오류확률은 $P_{org} = (1/2)^k \approx (0.6185)^{m/n}$이 된다.

2.1.3. 긍정오류에 대한 원인분석

블룸필터에서 긍정오류가 발생하는 이유는 크게 두 가지 요인으로 초래된다. 첫 번째 요인은 특정 해쉬함수에 두 가지 입력값을 적용하였을 때, 동일한 결과값을 가지는 해쉬함수의 충돌이라고 볼 수 있다.

전체 집합을 U 라하고 전체 요소 갯수를 u개로 정의하고, 이들 중 S집합에 속하는 요소 n개를 블룸필터에 입력하였을 경우에, 블룸필터의 긍정오류확률을 ϵ 이라 하자. 블룸필터가 m 비트로 구성되어있는 경우에, 블룸필터의 모든 비트들로 u를 표현할 수 있는 $2^m \geq u$인 경우에는 완전한 해쉬함수에 의한 충돌현상을 제거할 수 있다. 또한 $2^m \geq n+\epsilon(u-n)$인 경우도 완전한 해쉬함수를 사용하는 경우에는 제거할 수 있으며, 이러한 경우에는 긍정오류가 포함된 $n+\epsilon(u-n)$ 개를 블룸필터에 표현된다고 볼 수 있다.

두 번째 요인은 블룸필터에서 사용되는 여러 개의 해쉬함수들 간의 충돌에 의해 발생된다.

첫 번째 해쉬함수의 결과값과 두 번째 해쉬함수의 결과값이 동일한 경우에 충돌이 발생되며, 빈번히 발생될 수 있다. 이론적으로 해쉬함수의 긍정오류확률을 줄이기 위해서 해쉬함수의 개수를 늘리거나, m/n 값을 늘리는 방법이 사용될 수 있다. 전 절에서 언급된 최소 긍정오류확률을 얻기 위한 식 $P_{org} = (1/2)^k \approx (0.6185)^{m/n}$을 통해서 알 수 있다.

본 연구에서는 두 번째 요인에 해당되는 해쉬함수 간의 충돌에 의해 발생되는 긍정오류를 제거하기 위한 방법을 제안한다.

2.2. 병렬블룸필터 설계

본 논문에서 제안하는 병렬구조 블룸필터란 다 수개의 블룸필터들로 구성한 방식이다.

그림 3은 본 논문에서 제안한 병렬구조 블룸필터이다. n개의 입력요소에 대하여 k 개의 해쉬함수 결과값을 k개의 블룸필터들에 각각 적용하였을 경우의 긍정오류확률은 다음과 같다.

$$P_{fp\_pro} = [1-(1-\frac{1}{m})^n]^k \approx (1-e^{-n/m})^k$$

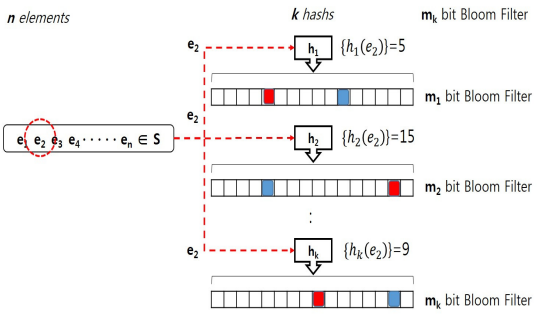

Fig. 3 Proposed parallel Bloom Filter

표 1은 n, k, m 값이 주어졌을 경우에 이에 대한 긍정오류 확률 P_{fp_org}와 제안된 방식에 의해, 동일한 n, m 값이 주어졌을 때, k개로 구성된 병렬구조 블룸필터 방식(제안방식)에서의 긍정오류 확률P_{fp_pro}을 비교한 것이다.

Table. 1 Comparison of False Positive Rate

n	k	m	P_{fp_org}	P_{fp_pro}
100	2	400	0.1548	0.0489
100	4	400	0.1596	0.0023
200	2	1600	0.0489	0.0138
200	4	1600	0.0239	0.0001

본 논문에서는 긍정오류확률의 오류의 확률을 최소화하기 위하여 병렬구조블룸필터를 제안하여 해쉬함수간의 발생되는 긍정오류를 제거하고, 기존의 방식에서 발생되는 긍정오류확률과 동일한 값을 갖는 다층구조의 블룸필터에서의 구조를 찾기 위하여 다음과 같은 방식을 제안한다.

① 일반 블룸필터와 동일한 긍정오류확률을 갖는 병렬 블룸필터 구조를 찾는다. 이때 병렬블룸필터에 필요한 해쉬함수의 갯수 k_{opt} 를 구한다.
② k_{opt} 개의 병렬구조의 블룸필터를 구성한다. 각 단은 서로 다른 해쉬함수별로 구성된 블룸필터이다.
③ 검색 과정은 각각의 해쉬함수의 결과에 따른 블룸필터값이 모두 “1”이 되면 존재로 확인하고, 하나라도 “0”이면 존재하지 않는 것으로 설정한다.

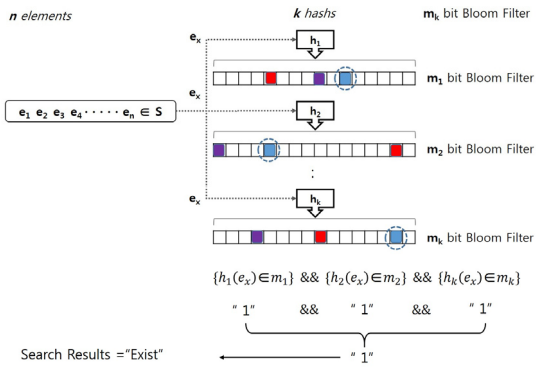

Fig. 4 Look up process of parallel Bloom Filter (1)

그림 4와 5는 제안한 병렬구조 블룸필터를 이용하여 요소의 존재여부를 검색하는 그림이다. n개의 입력요소에 대하여 k 개의 해쉬함수 결과값을 k개의 블룸필터들에 각각 적용하였을 때 그림 4와 같이 모두 “1”이 나타나면 존재하는 것으로, 그림 5와 같이 하나라도 “0”이 나타나면 존재하지 않는 것으로 판정한다.

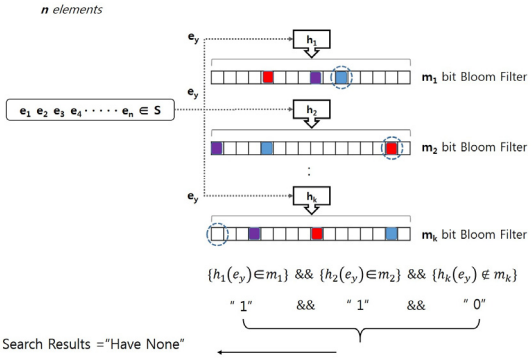

Fig. 5 Look up process of parallel Bloom Filter (2)

2.3. 성능비교

본 절에서는 일반 블룸필터와 제안된 방식의 병렬구조블룸필터간의 비교를 수행한다.

(1) $m/n=c$ 인 경우의 긍정오류확률 비교

예를 들어 $m/n=8$로 가정하였을 때, 원래 블룸필터의 긍정오류확률을 최소화 하기 위한 값은 다음과 같다.

$$k=\frac{m}{n}\ln 2 = 8\ln 2 = 5.545177$$

$$P_{fp-org} = (\frac{1}{2})^k = (0.6185)^{\frac{m}{n}} = 0.021415$$

k값을 이용하여 k단으로 구성된 병렬구조블룸필터에 대한 긍정오류확률은 다음과 같다.

$$P_{fp-pro} = (1-e^{-n/m})^k = (1-e^{-1/8})^{5.54517} = 0.000006$$

결국 $m/n=8$ 인 경우, 위와 동일한 긍정오류확률인 $P_{fp-org}=0.021415$을 얻기 위하여 요구되는 단수인 한 k_{opt} 값을 구하면,

$$(1-e^{-1/8})^k = 0.021415, k\ln(1-e^{-1/8}) = \ln(0.021415)$$

$$k_{opt} = \frac{\ln(0.021415)}{\ln(1-e^{-1/8})} = 1.795022$$

이러한 결과를 정리하면, 일반 블룸필터의 최소긍정오류확률을 갖기 위한 조건인 $m/n=8$, $k=5.545177$를 이용한 결과 얻어진 긍정오류확률은 $P_{fp-org}=0.021415$이다. 제안된 병렬구조블룸필터를 이용하여 이와 동등한 긍정오류확률을 얻기 위하여 동등한 크기의 블롬필터를 $k_{opt}=1.795022$ 단으로 연결하여 병렬처리방식으로 얻을 수 있음을 보인다.

Table. 2 Comparison of FP based on m/n

	$m/n=4$	$m/n=8$	$m/n=16$	$m/n=32$
k	2.772589	5.545177	11.090355	22.180710
P_{fp_org}	0.146338	0.021414	0.000459	0.000000
P_{fp_pro}	0.015253	0.000006	0.000000	0.000000
k_{opt}	1.273840	1.795022	2.741875	4.416330

표 2는 m/n 값을 일정한 상수로 유지하였을 경우에 대하여 일반 블룸필터에서 얻은 최소긍정오류확률 P_{fp-org} 와 동일한 단수(k)로 구성하여 제안된 병렬블룸필터 방식으로 얻은 긍정오류확률은 P_{fp-pro}이다. 마지막 k_{opt} 는 최소긍정오류확률과 동일한 값을 얻기 위하여 제안된 병렬방식으로 구성하였을 경우에 필요한 단수(the number of stage)를 의미한다.

(2) $k=c$ 인 경우의 긍정오류확률 비교

이번에는 긍정오류확률을 최소화하기 위한 해쉬함수의 갯수 k를 상수로 두고 고려한다.

예를 들어 $k=5$로 가정하였을 때, 일반 블룸필터의 긍정오류확률을 최소화 하기 위한 값은 다음과 같다.

$$\frac{m}{n} = \frac{k}{\ln 2} = \frac{5}{\ln 2} = 7.213475$$

$$P_{fp-org} = (\frac{1}{2})^k = (0.6185)^{\frac{m}{n}} = 0.031249$$

k값을 이용하여 k단으로 구성된 병렬구조블룸필터에 대한 긍정오류확률은 다음과 같다.

$$P_{fp-pro} = (1-e^{-n/m})^k = (1-e^{-1/7.213475})^5 = 0.000036$$

결국 $m/n=7.213475$인 경우, 위와 동일한 긍정오류확률인 $P_{fp-org}=0.031249$을 얻기 위하여 요구되는 단수인 한 k_{opt} 값을 구하면,

$$(1-e^{-1/7.213475})^{k_{opt}} = 0.031249,$$

$$k_{opt} = \frac{\ln(0.031249)}{\ln(1-e^{-1/7.213475})} = 1.695198$$

이러한 결과를 정리하면, 블룸필터의 최소긍정오류확률을 갖기 위한 조건인 $k=5$, $m/n=7.213475$를 이용한 결과 얻어진 긍정오류확률은 $P_{fp-org}=0.031249$이다. 제안된 병렬구조블룸필터를 이용하여 이와 동등한 긍정오류확률을 얻기 위하여 동등한 크기의 블롬필터를 $k_{opt}=1.695198$ 단으로 연결하여 병렬처리방식으로 얻을 수 있음을 보인다.

Table. 3 Comparison of FP rates based on k

	$k=3$	$k=5$	$k=7$	$k=9$
m/n	4.328085	7.213475	10.098865	12.984255
P_{fp_org}	0.124997	0.031249	0.007812	0.001953
P_{fp_pro}	0.008780	0.000036	0.000000	0.000000
k_{opt}	1.317428	1.695198	2.054639	2.397538

표 3은 k 값을 일정한 상수로 유지하였을 경우에 대하여 일반 블룸필터에서 얻은 최소긍정오류확률 P_{fp_org}와 동일한 단수(k)로 구성하여 제안된 병렬블룸필터 방식으로 얻은 긍정오류확률은 P_{fp_pro}이다. 마지막 k_{opt} 는 기존의 최소긍정오류와 동일한 값을 얻기 위하여 제안된 병렬방식으로 구성하였을 경우에 필요한 단수를 의미한다.

(3) 실제 적용 가능한 경우의 긍정오류확률 비교

이론적으로 m/n 값이 일정한 상수일 때, 최소긍정오류확률을 얻을 수 있는 k 값은 소수점으로 나타나기 때문에 이를 실제 현실에 적용할 수 없다.

그러므로 본 절에서는 실제 적용할 수 있도록 블룸필터의 크기는 2의 지수 승 크기로 하고, 또한 해쉬함수의 개수를 상수를 사용하여 현실에 적용할 수 있는 병렬구조블룸필터를 알아본다.

제안된 블룸필터에서는 표 2에서 나타난 k 값을 반

올림하여 해쉬함수의 개수로 정의하고 블룸필터의 크기를 $m/2$, $m/4$ 로 정의하였을 경우에 병렬 블룸필터의 단수를 계산하였다.

Table. 4 Comparison of FP rates based on k and m/n

	$m/n=4$	$m/n=8$	$m/n=16$	$m/n=32$
k	3	6	11	22
P_{fp-org}	0.146892	0.021577	0.000459	0.000000
k_{opt}	1.271340	1.791499	2.741790	4.416193
$k_{opt-m/2}$	2.056345	2.542681	3.589933	5.483580
$k_{opt-m/4}$	4.181740	4.112690	5.095204	7.179867

표 4는 m/n 값과 k 값을 일정한 상수로 유지하였을 경우에 대하여 일반 블룸필터에서 얻은 긍정오류값 P_{fp_org} 를 구한다. k_{opt} 는 동일한 긍정오류값을 얻기 위해 필요한 단수이다. $k_{opt-m/2}$ 는 제안된 병렬방식에서 블룸필터의 크기를 1/2로, $k_{opt-m/4}$ 는 블룸필터의 크기를 1/4로 줄였을 때, 필요한 단수를 의미한다.

$k=5.545177 \Rightarrow 6$으로 설정하고, 이에 대한 긍정오류확률은

$$P_{fp-org}=(1-(1-\frac{1}{m})^{nk})^k=(1-e^{-nk/m})^k$$

$$P_{fp-org}=(1-e^{-6/8})^6=0.021577$$

제안된 블룸필터에서는

$P_{fp-pro}=(1-e^{-n/m})^k=(1-e^{-1/8})^6=2.63\times10^{-6}$ 가 된다. 동일한 긍정오류확률인 $f=0.021577$을 얻기 위한 k_{opt} 값을 구하면

$$k_{opt}=\frac{\ln(0.021577)}{\ln(1-e^{-1/8})}=1.791499$$

$k_{opt-m/2}$를 구하기 위하여 동일한 긍정오류확률과 함께 블룸필터의 크기 1/2로 하여 단수를 구하면 다음과 같다.

$$k_{opt-m/2}=\frac{\ln(0.021577)}{\ln(1-e^{-1/4})}=2.542681$$

$k_{opt-m/4}$를 구하기 위하여 동일한 긍정오류확률과 함께 블룸필터의 크기 1/4로 하여 단수를 구하면 다음과 같다.

$$k_{opt-m/4}=\frac{\ln(0.021577)}{\ln(1-e^{-1/2})}=4.112690$$

즉, $m/n=8$, $k=6$인 블룸필터와 동일한 긍정오류확률을 얻기 위해서 제안된 병렬 구조 블룸필터를 적용하면 다음과 같다.

m 비트의 블룸필터로 적용하면 1.791499 단, $m/2$ 비트의 블룸필터로 적용하면 2.542681 단, $m/4$ 비트의 블룸필터로 적용하면 4.112690 단으로 동일한 긍정오류확률을 얻을 수 있다는 것을 알 수 있다.

(4) 하드웨어 구성

표 4에서 m/n=4 인 경우, m= 1024 bits의 일반블룸필터로 구성하면 그림 6과 같다. 반면에 m/4= 256 bits 크기의 블룸필터를 각각 해쉬함수 별로 5단 병렬형태로 구성하면 그림 7과 같이 구성된다.

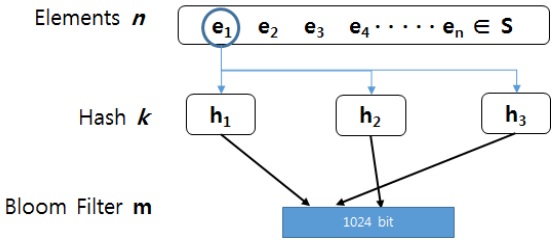

Fig. 6 H/W structure of General Bloom Filter

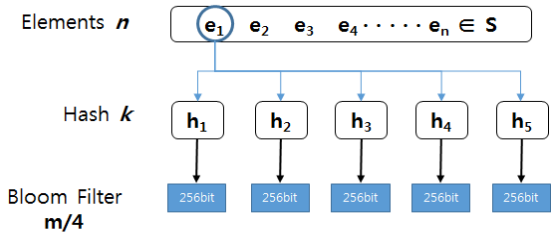

Fig. 7 H/W structure of Parallel Bloom Filter

일반 블룸필터는 그림 6과 같이 1024 bits의 메모리를 사용하고 있으며, 3개의 해쉬함수 결과값이 한 개의 블룸필터에 적용되어야 하며, 이때 긍정오류확률은 0.146892이다.

$$P_{fp-org}=(1-e^{-nk/m})^k=(1-e^{-3/4})^3=0.146892$$

제안된 방식은 그림 7과 같이 5×256 bits = 1280 bits의 메모리를 사용하고 있으며, 이때 긍정오류확률은 0.100925이다.

$$P_{fp-pro} = (1-e^{-n/m})^k = (1-e^{-1/1})^5 = 0.100925$$

5개의 해쉬함수값이 병렬로 블룸필터에 적용되기 때문에 속도와 성능 면에서 매우 빨라질 수 있다.

또한 그림 7과 같이 5개의 완전 해쉬함수 별로 병렬형태로 블룸필터를 구성하고, 특정 요소에 대한 블룸필터들의 비트 값이 모두 "1" 이 될 경우에는 요소에 대한 삭제가 가능하다.

Ⅲ. 결 론

블룸필터는 공간효율성을 가진 메모리 구조로서 지금까지 다양한 분야에 많이 사용되어 왔다. 또한 부정오류(false negative)가 전혀 발생하지 않고 없고, 단지 긍정오류가 존재하기 때문에 긍정오류를 최소화하기 위한 많은 노력이 시행되어왔다.

블룸필터에서 긍정오류확률은 동일한 해쉬함수에 의해 발생되는 결과값의 충돌에 의해 나타나는 유형과 다른 해쉬함수 결과값의 의해 충돌되는 경우로 구분할 수 있다. 여기서 첫 번째 유형의 긍정오류는 충돌이 없는 완전해쉬함수를 적용함으로서 제거할 수 있다. 두 번째 유형의 긍정오류는 타 해쉬함수의 결과값에 의해 충돌되는 긍정오류로 볼 수 있으며, 본 연구는 두 번째 유형의 긍정오류를 제거하기 위하여 해쉬함수별로 블룸필터를 병렬형으로 구성하는 방식을 제시하였다.

본 연구를 통하여 제안된 병렬구조 블룸필터는 기존의 일반블룸필터에 비해 약간 큰 메모리를 활용함으로서, 기존방식의 블룸필터의 긍정오류확률와 유사한 값을 얻을 수 있음을 보였다. 실제로 제안된 병렬구조의 블룸필터는 병렬처리 방식의 하드웨어를 이용하여 구현하면, 기존의 방식에 비해 처리속도도 빨라질 뿐 아니라, 선택한 해쉬함수가 충돌을 발생시키지 않는 완전 해쉬함수 라면 삽입과 삭제가 가능할 뿐 아니라, 긍정오류가 대부분 제거될 수 있다는 것을 알 수 있다.

또한 추후에는 제안된 병렬구조 블룸필터 방식을 기존의 삭제가 가능한 카운팅 블룸필터등과 비교하고, 이를 보완하여 각종 응용에 적용할 예정이다.

ACKNOWLEDGMENTS

This paper was supported by the Semyung University Research Grant of 2016.

REFERENCES

[1] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters : A Survey," *Internet Mathematics*, vol.1, no. 4, pp. 485 -509, Jan. 2004.

[2] S. Tarkoma, C. E. Lagerspetz, and E. Lagerspetz, "Theory and Practice of Bloom Filters for Distributed Systems," *IEEE Communications Surveys and Tutorials*, vol.14, no.1, pp.131-155 , Mar. 2012.

[3] M. Michael Mitzenmacher, "Compressed Bloom Filters," *IEEE/ACM Transactions on Networking*, vol.10, no.5, pp.604-612, Oct. 2002.

[4] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no.3, pp. 281-293, Aug. 2002.

[5] R. P. Laufer, P. B. Velloso, and O. C. M. B. Duarte, L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no.3, pp. 281-293, Aug. 2002.

[6] R. P. Laufer, P. B. Velloso, and O. C. M. B. Duarte "A Generalized Bloom Filter to Secure Distributed Network Applications," *Computer Networks*, vol. 55, no. 8, pp. 1804-1819, June 2011.

[7] F. Bonomi, M. Mitzenmacher, R. Panigrahy, S. Singh, and G. Varghese, "An improved Construction for Counting Bloom Filters," *in 14th Annual European Symposium an Algorithms, LNCS 4168*, pp. 684-695, Sep. 2006.

[8] C. E. Rothenberg, C. A. B. Macapuna, F. L. Verdi, and M.Magalhaes, "The deletable Bloom Filters: a new member of the Bloom family," *IEEE Communications*

Letters, vol. 14, no. 6, pp. 557-559, June 2010.

[9] J. Qian, Q. Zhu, and Y. Wang, “Bloom Filter Based Associative Deletion,” I*EEE transactions on parallel and distributed syst*, vol.25, no.8, pp. 1986-1998, Aug. 2014.

[10] H. Lim, J. Lee, and C. Yim, “Complement Bloom Filter for Identifying True Positiveness of a Bloom Filter,” *IEEE communications letters*, vol.19, no.11, pp. 1905-1908, Sep. 2015.

장영달 (Young-dal Jang)

1998년 세명대학교 전자공학과 학사
2001년 세명대학교 전기전자공학부 석사
2016년 세명대학교 정보통신학과 박사수료
※관심분야 : 개인정보보호, 데이터베이스 보안, 네트워크보안

김지홍(Ji-hong Kim)

1982년 한양대학교 전자공학과 학사
1984년 한양대학교 전자통신학과 석사
1996년 한양대학교 전자통신학과 박사
1991년 ~ 세명대학교 정보통신학부 교수
※ 관심분야 : 네트워크 보안, 데이터베이스 보안