하나의 n 차 정사각 불리언 행렬과 모든 n 차 정사각 불리언 행렬 사이의 연속곱셈에 관한 연구

A Study on Multiplying an $n \times n$ Boolean Matrix by All $n \times n$ Boolean Matrices Successively

한재일
국민대학교

Han Jae-Il
Kookmin Univ.

요약

모든 n차 정사각 불리언 행렬 사이의 연속곱셈은 D-클래스 계산과 같은 응용에서 기본적으로 요구되는 연산이다. 그러나 불리언 행렬에 대한 많은 연구에도 불구하고 이에 대한 연구는 아직 보이지 않고 있다. 본 논문은 하나의 n차 정사각 불리언 행렬과 모든 n차 정사각 불리언 행렬 사이의 이중 연속곱셈을 효율적으로 할 수 있는 이론을 제시하고, 이를 모든 n차 정사각 불리언 행렬 사이의 연속곱셈에 적용한 실행결과를 보인다.

Abstract

The successive multiplication of all $n \times n$ boolean matrices is necessary for applications such as D-class computation. But, no research has been performed on it despite many researches dealing with boolean matrices. The paper suggests a theory with which successively multiplying a $n \times n$ boolean matrix by all $n \times n$ boolean matrices can be done efficiently, applies it to the successive multiplication of all $n \times n$ boolean matrices and shows its execution results.

I. 서 론

불리언 행렬에 대한 많은 연구가 수행되었으나[1-6] 모든 n 차 정사각 불리언 행렬 사이의 곱셈을 다루는 연구는 극히 소수[8-11]가 보이고 있다. 그러나 이 연구들도 불리언 행렬 쌍에 대한 곱셈만을 다루고 있으며 모든 n 차 정사각 불리언 행렬 사이의 연속된 곱셈에 대한 연구는 아직 보이지 않고 있다. D-클래스 계산과 같은 응용에서는 최악의 경우 모든 n 차 정사각 불리언 행렬 사이의 사중 연속곱셈을 요구한다[7].

본 논문은 하나의 n 차 정사각 불리언 행렬과 모든 n 차 정사각 불리언 행렬 사이의 이중 연속곱셈을 효율적으로 할 수 있는 이론을 제시하고, 이를 모든 n 차 정사각 불리언 행렬 사이의 이중 연속곱셈에 적용한 실행결과에 대하여 기술한다. 본 논문의 구성은 다음과 같다. 2장은 기존에 제시된 불리언 행렬 곱셈에 대한 연구와 문제점을 논하고, 본 논문에서 사용할 용어 및 기호를 정의한다. 3장은 하나의 n 차 정사각 불리언 행렬과 모든 n 차 정사각 불리언 행렬 사이의 이중 연속곱셈을 효율적으로 할 수 있는 이론을 기술하며, 4장은 이를 모든 n 차 정사각 불리언 행렬 사이의 이중 연속곱셈에 적용한 실행결과에 대하여 기술한다. 5장은 결론 및 향후 연구방향에 대하여 논한다.

II. 관련 연구 및 문제점

불리언 행렬은 원소가 0(거짓)이나 1(참) 값을 갖는 행렬로 정의되며 단순하고 논리적인 특성으로 인해 여러 분야에 응용되어 유용하게 사용되고 있다[4-6]. 그러나 불리언 행렬에 대한 연구는 대부분 두 불리언 행렬곱셈의 최적화에 초점을 두고 있으며[1-9], 극히 소수의 연구[8-11]만이 모든 불리언 행렬 사이의 곱셈을 다루고 있다.

모든 불리언 행렬의 곱셈에 대한 연구 중 [10, 11]은 순환문 개선에 초점을 두었으나 메모리 공간이나 실행시간 등의 개선 정도가 매우 미약하다. 반면 행렬 연산을 벡터 기반으로 계산할 것을 제안한 [8-9]는 동시에 메모리 공간과 실행시간을 개선하였으나, 모든 불리언 행렬의 곱셈은 근본적으로 NP-완전 계산복잡도를 가지므로 보다 큰 크기의 행렬에 적용하려면 더 많은 개선이 필요하다.

본 논문은 $F=\{0,1\}$, $M_n(F)$를 모든 $n \times n$ 불리언 행렬의 집합, A_i 와 A^i 는 각각 A 행렬의 i 행과 열, A^T는 A의 전치(transpose)행렬이라 할 때 다음과 같이 기호를 정의한다.

$$v = (b_0 b_1 \cdots b_{n-1})$$
$$where\ b_i \in F, 0 \le i \le n-1$$

$$v^T = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{pmatrix} where\ v = (b_0 b_1 \cdots b_{n-1})$$

$$V = \{\ (b_0 b_1 \cdots b_{n-1}) \mid b_i \in F \text{for } 0 \le i \le n-1\ \}$$

$$V^T = \{ v^T \mid v \in V \}$$

$$(V)_k = \{ \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{k-1} \end{pmatrix} \mid v_i \in V, 0 \le i \le k-1 \}$$

$$(V^T)^k = \{\ (v_0^T v_1^T \cdots v_{k-1}^T) \mid v_i \in V, 0 \le i \le k-1\ \}$$

$$Av^T = \begin{pmatrix} A_0 v^T \\ A_1 v^T \\ \vdots \\ A_{n-1} v^T \end{pmatrix} where\ A \in M_n(F), v \in V$$

$$vB = (vB^0\ vB^1 \cdots vB^{n-1}) \quad where\ B \in M_n(F), v \in V$$

III. $n \times n$ 불리언 행렬의 연속곱셈

본 논문은 하나의 $n \times n$ 불리언 행렬과 모든 $n \times n$ 불리언 행렬의 연속곱셈을 보다 효율적으로 하기 위해서 불리언 행렬 연산의 특성을 이용하여 연속곱셈 시간을 개선할 수 있는 네 개의 정리를 제시하며, 공간의 제약으로 [정리 4]에 대한 증명만을 기술한다.

[정리 1]
$M_n(F)$에 속한 임의의 $n \times n$ 불리언 행렬 A에 대해
$R_A = \{\ AB \mid B \in M_n(F)\ \}$,
$C_A = \{\ (A_0 v^T\ A_1 v^T \cdots A_{n-1} v^T)^T \mid v \in V\ \}$
로 정의하면 $(C_A)^n = R_A$ 이다.

[정리 2]
$M_n(F)$에 속한 $n \times n$ 불리언 행렬 A에 대해
$L_A = \{\ BA \mid B \in M_n(F)\ \}$,
$T_A = \{\ (vA^0\ vA^1 \cdots vA^{n-1}) \mid v \in V\ \}$
로 정의하면 $L_A = (T_A)_n$이다.

[정리 3]
A와 Z를 $M_n(F)$에 속한 임의의 두 불리언 행렬이라 할 때
$S_{AX} = \{\ AX \mid X \in M_n(F)\ \}$,
$S_{UAX} = \{\ UAX \mid U, X \in M_n(F)\ \}$,
$T_Z = \{\ (vZ^0\ vZ^1 \cdots vZ^{n-1}) \mid v \in V\ \}$
로 정의하면 $S_{UAX} = \bigcup_{Z \in S_{AX}} (T_Z)_n$ 이다.

위의 [정리 1]-[정리 3]을 적용하면 주어진 $n \times n$ 불리언 행렬과 모든 $n \times n$ 불리언 행렬 사이의 연속곱셈은 행렬 사이의 연산대신 행렬과 벡터의 연산으로 수행할 수 있다. 그러나 이 경우 각 불리언 행렬에 대하여 모든 벡터와의 첫 번째 곱셈에서 얻은 벡터 집합으로부터 불리언 행렬을 하나씩 생성하여야 하는 오버헤드를 포함하며 많은 계산이 중복된다. [정리 4]는 $M_n(F)$의 어떤 불리언 행렬에 모든 $n \times n$ 불리언 행렬을 곱하여 얻는 행렬 집합이 이미 존재하는 경우 모든 $n \times n$ 불리언 행렬에 대한 이중 연속곱셈을 수행하여 행렬 집합을 계산할 필요 없이 이전의 연속곱셈 결과를 사용하면 되므로 앞의 정리만 적용하였을 경우 나타나는 오버헤드와 많은 중복계산을 상당부분 개선할 수 있다.

[정리 4]
$M_n(F)$에 속한 임의의 두 불리언 행렬 A, B 에 대해
$S_A = \{\ AX \mid X \in M_n(F)\ \}$,
$S_B = \{\ BY \mid Y \in M_n(F)\ \}$,
$T_A = \{\ UAX \mid U, X \in M_n(F)\ \}$,
$T_B = \{\ VBY \mid V, Y \in M_n(F)\ \}$
로 정의할 때 $S_A = S_B$이면 $T_A = T_B$이다.

(증명)
$$\begin{aligned} T_A &= \{\ UAX \mid U, X \in M_n(F)\ \} \\ &= \{\ UW \mid U \in M_n(F), W \in S_A\ \} \\ &= \{\ UW \mid U \in M_n(F), W \in S_B\ \} \\ &= \{\ UBY \mid U, Y \in M_n(F)\ \} \\ &= T_B \ \blacksquare \end{aligned}$$

IV. 실행결과

위에 기술한 정리를 적용하였을 경우 실제 연속곱셈에 대한 실행시간 개선정도를 알아보기 위하여 행렬 사이의 연속곱셈, [정리 1]-[정리 3]을 적용한 연속곱셈, 모든 정리를 적용한 연

속곱셈에 대한 알고리즘을 작성하여 실행하였다. 알고리즘은 Java 언어로 구현하였으며 2개의 Zeon CPU, 1GB RAM, 250GB HDD, Fedora 9.0 환경에서 실행하였다. <표 1>은 세 알고리즘의 실행 결과이며, 정리를 적용할 경우 상당한 실행시간 개선 결과를 보이고 있다.

[표 1] 실행시간 비교

행렬크기	직접 연속곱셈 알고리즘	정리 1-3 적용 연속곱셈 알고리즘	정리 1-4 적용 연속곱셈 알고리즘
2×2	0.006 초	0.02 초	0.015 초
3×3	9.682 초	0.041 초	0.016 초
4×4	29312169 초	54.056 초	2.125 초
5×5	5125731454508070 초 이상	2728587 초	26267 초

V. 결론 및 향후 연구방향

모든 n 차 정사각 불리언 행렬 사이의 연속곱셈은 D-클래스 계산과 같은 응용에서 기본적으로 요구되는 연산이다. 그러나 불리언 행렬에 대한 많은 연구에도 불구하고 이에 대한 연구는 아직 보이지 않고 있다.

본 논문은 하나의 n 차 정사각 불리언 행렬과 모든 n 차 정사각 불리언 행렬 사이의 이중 연속곱셈을 효율적으로 할 수 있는 네 개의 정리를 제시하고, 정리를 적용한 알고리즘의 실행결과를 보임으로써, 정리를 적용할 경우 모든 n 차 정사각 불리언 행렬 사이의 연속곱셈 실행시간이 크게 개선됨을 보였다.

그러나 모든 $n \times n$ 불리언 행렬 사이의 곱셈은 근본적으로 NP-완전문제이기 때문에 실행결과에서 보듯이 행렬 크기가 커짐에 따라 실행시간이 급격히 증가한다. 따라서 본 논문의 결과는 단지 시작일 뿐이며 앞으로 모든 불리언 행렬에 대한 곱셈을 다차원 함수 계산복잡도로 수행할 수 있는 이론과 알고리즘에 대한 연구가 필요하며, 이와 더불어 알고리즘의 최적화 방법, 병렬 컴퓨팅 적용 등에 대한 연구도 필요하다.

▌참 고 문 헌▐

[1] Macii, E., "A Discussion of Explicit Methods for Transitive Closure Computation Based on Matrix Multiplication", 29th Asilom Conference on Signals, Systems and Computers, Vol.2, pp.799-801, 1995.

[2] Angluin, D., "The four Russians'algorithm for boolean matrix multiplication is optimal in its class", ACM SIGACT News, Vol.8, No.1, pp.29-33, 1976.

[3] Booth, K. S., "Boolean matrix multiplication using only $O(n^{\log_2 7}\log n)$ bit operations", ACM SIGACT News, Vol.9, No.3, pp.23, 1977.

[4] Lee, L., "Fast context-free grammar parsing require fast Boolean matrix multiplication," JACM, Vol.49, No.1, pp.1-15, 2002.

[5] Yi, X., et al., "Fast Encryption for Multimedia," IEEE Transactions on Consumer Electronics, Vol.47, No.1, pp.101-107, 2001.

[6] Martin, D. F., "A Boolean matrix method for the computation of linear precedence functions," CACM, Vol.15, No.6, pp.448-454, 1972.

[7] Rim, D. S., and Kim, J. B., "Tables of D-Classes in the semigroup B of the binary relations on a set X with n-elements", Bull. Korea Math. Soc., Vol.20, No.1, pp.9-13, 1983.

[8] 한재일, "모든 m x k 불리언 행렬과의 효율적 곱셈에 관한 연구", 한국콘텐츠학회 논문지, 제6권, 제2호, pp.27-33, 2006.

[9] 한재일, "n x m 불리언 행렬과 모든 l x n, m x k 불리언 행렬과의 중첩곱셈에 요구되는 공간 및 시간 복잡도 개선에 대한 연구", 국민대학교 기초과학연구소 논문집, 제25권, pp.223-230, 2006.

[10] 신철규, 한재일, "내부 순환문 개선을 통한 Linux 기반의 D-클래스 계산 고효율 순차 알고리즘", 한국SI학회 춘계학술대회 논문집, pp.526-531, 2005.

[11] 신철규, 한재일, "공유 메모리 기반의 고성능 D-클래스 계산 병렬 알고리즘", 한국컴퓨터종합학술대회 논문집, 제32권, 제1호, pp.10-12, 2005.