İKİLİ KARAR DİYAGRAMLARI YARDIMIYLA BOOLE FONKSİYONLARININ ASAL BİLEŞENLERİNİN BELİRLENMESİ

Utku ÖZCAN
Elektronik ve Haberleşme Y. Müh.
ozcan@ieee.org

Ahmet DERVİŞOĞLU
Yeditepe Ü. EE Müh. Bölümü, Istanbul
adervisoglu@yeditepe.edu.tr

Anahtar sözcükler: ikili karar diyagramları, lojik tasarım, asal bileşen

ÖZET

İki seviyeli minimal gerçeklemeyi oluşturan iki adımdan birincisi olan asal bileşen bulma (prime computation), ikinci adım olan örtü problemini (covering problem) doğrudan etkilemektedir. Örtü probleminde olduğu gibi, asal bileşen bulmada da kısa sürede az bellek kullanarak çözüm bulma, problemin karmaşıklığı arttıkça önem kazanmaktadır.

Bu çalışmada, Boole fonksiyonların asal bileşenlerini bulmak için İkili Karar Diyagramları (Binary Decision Diagrams: BDD) adı verilen bir gösterim şeklinden yararlanılmıştır. BDD gösteriminin öteki gösterimlerden hız ve bellek bakımından daha iyi olduğu literatürde belirtilmiştir.

BDD ile Boole fonksiyonlarının asal bileşenlerinin bulunabileceğini göstermek için bir bilgisayar programı (IPCP) geliştirilmiştir. İki seviyeli minimal gerçeklemenin birinci adımı olan asal bileşenleri bulmada, bu çalışmada geliştirilmiş olan IPCP programı yardımıyla BDD'den yararlanılması, asal bileşen bulma problemi konusunda bir alternatif olmaktadır. Asal bileşen bulmada BDD kullanımı, örtü problemi için de yeni seçenekler sunmaktadır.

1. GİRİŞ

Bir Boole fonksiyonunun asal bileşen kümesi, iki seviyeli lojik indirgemenin ikinci adımı olan örtü problemine girdi olarak verilir. Günümüzde birçok asal bileşen kümesi oluşturma algoritmaları vardır. Karmaşık problemler için, bu algoritmalarda minimal zaman ve bellek kullanımı gereklidir.

Boole fonksiyonların ikili karar diyagramları (binary decision diagrams: BDD) adı verilen bir gösterimi vardır. Bu gösterimde gerçekleştirilen temel algoritmaların öteki gösterimlerden daha hızlı olduğu ve daha az bellek kullandığı bilinmektedir [1].

Temel algoritmaların BDD'de daha hızlı çalışması özelliği, Boole fonksiyonların asal bileşenlerinin de hızlı bulunması için alternatif olarak BDD kullanılabileceğini gösterir.

Bir Boole fonksiyonunun asal bileşen kümesi, Shannon kofaktörlerinin asal bileşen kümelerinden elde edilebilir [4]. Bu da, Boole fonksiyonunun küplerini hesaplamadan asal bileşen kümesini elde etmek anlamına gelir. Küplerin hesaplanmaması, çözüm süresini kısaltabilir.

BDD'de bir düğüm, Boole fonksiyonunu Shannon açınımı cinsinden gösterir. Bu sayede BDD gösteriminde bir Boole fonksiyonunun asal bileşen kümesi Shannon kofaktörlerinin asal bileşenleri cinsinden bulunabilir.

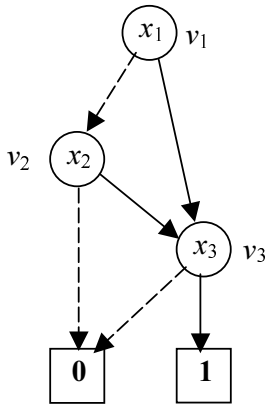
Bu çalışmada, MCNC bençmarkında tanımlanmış bir Boole fonksiyonunun BDD gösterimi yardımıyla asal bileşen kümesini bulan bir program geliştirilmiştir. IPCP (Implicit Prime Computation Program) adı verilen program, Coudert'in BDD'lerde asal bileşen bulan algoritmasını kullanmaktadır [2]. Program, BDD gösterimi için Kopenhag Enformasyon Teknolojisi Enstitüsü'nde bir doktora tezinde geliştirilmiş olan BuDDy adı verilen programdan yararlanmaktadır. IPCP, Boole fonksiyonunu MCNC formatında okur, BDD gösterimine çevirir ve BDD gösteriminde asal bileşen hesabı yapar.

BDD'de çalışan bu algoritma, gereğinden fazla asal bileşen bulmaktadır. Ancak bulunan küme elemanları birbirini kapsadığından, eleme işlemi sonucunda her zaman doğru sonuç elde edilmektedir.

IPCP, BDD'lerin asal bileşen bulma konusunda bir alternatif olabileceğini göstermektedir.

2. BDD GÖSTERİMİ

BDD, çevre içermeyen, köklü - yönlü bir graftır. BDD grafında bir düğüm Boole fonksiyonunu gösterir. BDD grafında herhangi bir *v* düğümünden çıkan eleman sayısı her zaman 2'dir. Kendisine eleman gelmeyen düğüm kök düğümdür ve BDD grafının Boole fonksiyonunu gösterir. Bir *v* düğümünden çıkan elemanlar, düğümün Boole fonksiyonunun kofaktörlerini gösterdiği düğümlere gider. Son düğüm, kendisinden çıkan elemanların 0 ve 1 değerlerine bağlı olduğu düğümdür. Kök düğümden son düğüme kadar olan herhangi bir yoldaki düğümler, değişken açısından sıralıysa BDD grafı OBDD (ordered BDD) adını alır. Kendisinden daha ufak BDD gösterimi olmayan BDD grafına ROBDD (reduced OBDD) denir. Literatürde ROBDD'ye çoğu kez kısaca BDD



Şekil 1 $f(x_1, x_2, x_3) = x_1x_3 + x_2x_3$ fonksiyonunun BDD grafı

denmektedir. Şekil 1'de, 3 değişkenli bir Boole fonksiyonunun BDD grafı görülmektedir.

BDD'lerin temel algoritmalarda Çarpımlar Toplamı ve Toplamlar Çarpımı'yla zaman karmaşıklığı açısından karşılaştırması Tablo 1'de gösterilmiştir:

Tablo 1 : ÇT, TÇ ve BDD'lerde karmaşıklık mertebeleri

Algoritma	*ÇT*	*TÇ*	*BDD*
Yeterlik	polinomsal	Üstel	Lineer
Totoloji	Üstel	Polinomsal	Sabit
tümleme	Üstel	Üstel	Sabit

Temel algoritmalarda BDD'nin üstünlüğü açıkça görülmektedir.

BDD'lerde tanımlanmış tüm işlemler, kullanılan BDD grafların büyüklüğü cinsinden polinomsal zamanda gerçekleştirilir:

Tablo 2 : BDD'deki temel işlemlerin karmaşıklık mertebeleri

İşlem	*İfade*	*Karmaşıklık*
İndirge (reduce)	BDD → ROBDD	O (|G|)
Uygula (apply)	G_1 <op> g_2	O (|G_1| * |G_2|)
Kısıtla (restrict)	$f(x_i = a)$	O (|G|)
Birleştir	$f(x_i = g)$	O (|F| * |G|)

BDD'deki tüm temel işlemlerin polinomsal zamanda olması, lojik devre teorisi ile ilgili belirli bir algoritmanın hızlı bir şekilde yapılabilmesine olanak vermektedir.

BDD'lerde değişken sıralaması grafın büyüklüğünü doğrudan etkilemektedir. Boole fonksiyonlarına en uygun değişken sıralamasını bulmak "NP-hard" sınıfındadır [3].

3. ASAL BİLEŞEN ALGORİTMASI

Bir Boole fonksiyonunun küpleri, bilgisayar programlamasına uygun olan geniş çarpım (metaproduct) adı verilen bir gösterimde tanımlanabilir. Bu gösterimdeki bir küpte, varlık değişkeni o_k (occurrence variable) ve işaret değişkeni s_k (sign variable) tanımlanır. Örneğin 3 değişkenli bir Boole fonksiyonunda $\bar{x}_1x_2x_3$ kübünün geniş çarpım ifadesi ([o_1 o_2 o_3], [s_1 s_2 s_3]) = ([1 1 1], [0 1 1]) olur.

Bir Boole fonksiyonunun asal bileşen kümesi $AB(f)$, Shannon kofaktörlerinin asal bileşen kümesi cinsinden ifade edilebilir [4]. Dolayısıyla, Boole fonksiyonunun asal bileşen kümesi $AB(f)$, rekürsif bir fonksiyonla, BDD gösteriminde, geniş çarpımlarla ifade edilebilir [2]:

$$\begin{aligned} AB(f) = &\left(\bar{o}_k \wedge AB\left(f_{\bar{x}_k} \wedge f_{x_k}\right)\right) \vee \\ &\left(o_k \wedge \bar{s}_k \wedge AB\left(f_{\bar{x}_k}\right) \wedge \neg AB\left(f_{\bar{x}_k} \wedge f_{x_k}\right)\right) \vee \\ &\left(o_k \wedge s_k \wedge AB\left(f_{x_k}\right) \wedge \neg AB\left(f_{\bar{x}_k} \wedge f_{x_k}\right)\right). \end{aligned}$$

Bu çalışmada yukarıdaki rekürsif algoritmaya bazı sınır değer koşulları uygulanmıştır. Her rekürsif adımda incelenen değişken, BDD grafının kök düğümünde tanımlı değilse rekürsif formül basitleşir. Bu sınır değer koşulunda, rekürsif formülün 2. ve 3. terimleri düşer. Bu da hesaplama süresini ve belleğini önemli ölçüde azaltır. Örneğin t481 bençmarkı sınır değer olmadan 47.84 saniyede çözülürken, sınır değer kullanılarak 2 saniyede sonuca ulaşılmaktadır.

Asal bileşenlerin BDD ile bulunabileceğini gösterebilmek için, bu çalışmada IPCP adlı program geliştirilmiştir. Program Boole fonksiyonunu MCNC formatında okur, BDD grafına çevirir ve BDD üstünde asal bileşen kümesini oluşturur.

Asal bileşenlerinin bulunması istenen n x m Boole fonksiyonu, önce m ayrı BDD grafına dönüştürülür ve m graf birbirleriyle çarpılarak 2^m tane BDD grafı elde edilir. BDD üstünde çarpma işlemi polinomsal zamandadır. Elde edilen 2^m BDD grafının ayrı ayrı asal bileşenleri oluşturulur. Gereksiz işlemden kurtulmak için, BDD grafının totoloji olup olmadığına bakan sınır değer koşulları kullanılır. Tüm bu işlemler sonucunda elde edilen asal bileşen kümesinde, ortak, birbirini kapsayan ve gereksiz asal bileşenler sıralı arama yöntemiyle elenir ve doğru sonuç elde edilir.

IPCP, BDD üstünde işlem yapabilmek için, BuDDy BDD yazılım paketini kullanmaktadır. BuDDy, Kopenhag Enformasyon Teknolojisi Üniversitesi'nden Joern Lind-Nielsen'in doktora çalışması sırasında geliştirilmiştir.

Program, dual 450 MHz UltraSPARC-II RISC işlemcili 2 GB RAM'li bir iş istasyonunda geliştirilmiştir.

Asal bileşen algoritmasının BDD'deki işlemi, gereğinden fazla asal bileşen üretmekte ve program bu gereksiz asal bileşenleri elemek için ekstra zaman harcamaktadır. Gereksiz asal bileşenler, rekürsif algoritmanın 2. ve 3. çarpım terimindeki tümleme işleminden kaynaklanmaktadır. Gereksiz asal bileşenler oluşturulmasını engelleyen, BDD gösteriminde çalıştırılabilen yöntemler geliştirilmiştir [5]. IPCP, gereksiz asal bileşenleri elemede BDD gösteriminden yararlanmamaktadır.

Algoritma, simetrik fonksiyonlarda gereksiz asal bileşen üretmemekte ve hızlı bir şekilde çözümü bulmaktadır.

4. SONUÇ

IPCP 90 bençmarkta test edilmiştir. Bu bençmarkların en önemlileri Tablo 3 ve 4'te verilmiştir.

Boole fonksiyonunun asal bileşen kümesini BDD gösteriminde üreten algoritma, gereksiz asal bileşenler oluşturmaktadır. Gereksiz asal bileşenler, gerçekte Boole fonksiyonunda tanımlı küptürler ve algoritmanın bulduğu diğer asal bileşenler tarafından her zaman kapsanmaktadır. Gereksiz asal bileşenler nedeniyle, asal bileşen bulma algoritması gereğinden fazla zaman ve bellek kullanarak çalışmaktadır.

Bazı bençmarklarda fazla küp oranı, doğru sonuçtan 1000 kat daha fazla olmaktadır. Bençmarkların büyük çoğunluğunda çözümden 10 kat daha fazla sayıda asal bileşen kümeler elde edilmektedir.

IPCP, Boole fonksiyonlarının asal bileşenlerini bulmada ve örtü problemi için BDD'nin bir alternatif olabileceğini göstermesi açısından geliştirilmiştir.

KAYNAKLAR

[1] Andersen, H.R., "An Introduction to Binary Decision Diagrams", Lecture Notes for 49285 Advanced Algorithms E97, Department of Information Technology, Technical University of Denmark, 1997.

[2] Coudert, O. and Madre, J.C., "Implicit and Incremental Computation of Primes and Essential Primes of Boolean Functions", 29th Design Automation Conference, Anaheim, CA, USA, pp 36 – 39.

[3] Wegener, I., "Branching Programs and Binary Decision Diagrams", SIAM Mon ographs on Discrete Mathematics and Applications, (SIAM, 2000).

[4] Brayton R.K., G.D. Hachtel, C.T. McMullen, A.L. Sangiovanni-Vincentelli, Logic Minimization Algorithms for VLSI Synthesis (Kluwer Academic Publishers, Dordrecht, 1984).

[5] Coudert, O. and Madre, J.C., "A New Graph Based Prime Computation Technique", Logic Synthesis and Optimization, Ed. Sasao, T., (Kluwer Academic Publishers, 1995).

Tablo 3 : Bazı zor bençmarkların IPCP programına uygulanması

Bm	I/O	NMI	NPI	RNPI	ENPI	T_{BDD}	T_{PIBDD}	T_{ELIM}	S_{BDD}	S_{RED}	S_{IRR}
t481	16/1	178496	481	481	481	0.04	2.01	0.03	640	19725	11552
Prom2	9/21	3027	2635	1.4M	2635	0.13	342.8	565.85	18720	64.6M	33M
Shift	19/16	4194304	165133	1.6M	165133	0.01	510.00	23760	1220	93.7M	36.3M

bm: bençmark
I/O: I: Giriş değişken sayısı, O: çıkış fonksiyon sayısı
NMI: Minterm sayısı
NPI: Gerçekteki asal bileşen sayısı
RNPI: BDD yardımıyla hesaplanan asal bileşen sayısı
ENPI: Eleme sonucunda elde edilen asal bileşen sayısı
T_{BDD}: BDD graflarını oluşturma süresi [saniye]
T_{PIBDD}: Asal bileşenlerin BDD graflarını oluşturma süresi [saniye]
T_{ELIM}: Gereksiz asal bileşenleri eleme süresi [saniye]
S_{BDD}: BDD grafının bellekte kapladığı alan [bayt, K, M]
S_{RED}: Elenmemiş asal bileşenlerin char formatında harcadığı bellek [bayt, K, M]
S_{IRR}: Elenmemiş asal bileşenlerin bit formatında harcadığı bellek [bayt, K, M]

Tablo 4 : Diğer bençmarkların IPCP programına uygulanması

bm	I/O	NMI	NPI	RNPI	ENPI	T_{BDD}	T_{PIBDD}	T_{ELIM}	S_{BDD}	S_{RED}	S_{IRR}
5xp1	7/10	784	390	2406	390	0.01	0.32	0.14	1760	76996	57752
9sym	9/1	696	1680	1680	1680	0.01	2.28	0.22	660	45364	40328
addm4	9/8	1310	1122	1806	1122	0.06	0.32	0.20	4600	61408	43352
alu2	10/8	1380	434	3736	434	0.02	8.78	0.21	4920	131.4K	87.6K
alu3	10/8	1076	540	2320	540	0.01	4.42	0.14	4940	81.3K	54.4K
apex4	9/19	2970	2336	40591	2336	0.12	18.94	11.34	20420	1.74M	0.93M
apla	10/12	157	201	90457	201	0.03	98.06	3.45	4560	3.45M	2.07M
dist	8/5	591	401	624	401	0.03	0.00	0.00	14984	18100	14984
f51m	8/8	1024	561	2077	561	0.04	0.00	0.00	1400	66468	49856
luc	8/27	2246	190	2.6M	190	0.02	756.08	2510	19342	102M	46M
m1	6/12	218	59	2307	59	0.01	0.00	1.00	1160	73700	55280
m2	8/12	831	243	49494	243	0.03	9.00	1.00	2840	1.89M	1.13M
m3	8/16	1105	344	39975	344	0.04	8.00	2.00	3100	1.52M	0.91M
m4	8/16	2134	670	56467	670	0.07	9.00	5.00	4260	2.15M	1.3M
max128	7/24	1616	469	596K	469	0.05	498.00	55.00	3320	26.2M	13.6M
max1024	10/6	3232	1278	3726	1278	0.13	2.00	0.00	6300	123.7K	87.3K
mlp4	8/8	678	606	1272	606	0.03	0.00	0.00	3140	40708	30536
rd53	5/3	48	51	62	51	0.00	0.00	0.00	460	1306	1496
rd84	8/4	411	633	825	633	0.02	1.00	0.00	1180	23104	19808
sao2	10/4	747	184	260	184	0.01	1.00	0.00	3080	8324	6248
sqr6	6/12	259	205	1541	205	0.01	0.00	0.00	1440	49316	36992
sym10	10/1	837	3150	3150	3150	0.04	14.00	1.00	760	91354	75608
tms	8/16	790	162	13695	162	0.01	3.00	1.00	2840	535K	321K
z5xp1	7/10	576	390	2406	390	0.03	1.00	0.00	1380	76996	57752
z9sym	9/1	420	1680	1680	1680	0.02	2.00	0.00	660	45364	40328
ex1010	10/10	1024	1333	2647	1333	0.11	1.00	0.00	21580	98.2K	62.1K