itüdergisi/***d***
mühendislik
Cilt:9, Sayı:1, 45-56
Şubat 2010

Yüksek hızlı sayısal FIR filtrelerinin tasarımında alan optimizasyonu algoritmaları

Levent AKSOY*, Ece Olcay GÜNEŞ

İTÜ Fen Bilimleri Enstitüsü, Elektronik Mühendisliği Programı, 34469, Ayazağa, İstanbul

Özet

Son on yıl içinde, birden fazla katsayının çarpımı (MCM) problemi, bir başka deyişle, bir değişkenin birden fazla katsayı ile çarpımının en az sayıda toplama/çıkarma işlemleri kullanılarak tasarımı, için etkili algoritmalar önerilmiştir. Bu algoritmalarda, toplama/çıkarma işlemi iki girişli bir işlem olarak kabul edilmekte ve genellikle, hesaplama süresi fazla olan elde ötelemeli toplayıcılar ile gerçeklenmektedir. Bunun yanında, elde korumalı toplayıcılar (CSA) çok girişli toplama işlemlerinin yüksek hızlı tasarımında sıklıkla kullanılmaktadır. Yine de, CSA blok sayısının optimizasyonu için önerilen algoritmalar sezgisellerdir ve minimum sonucu garanti edemezler. Bu makalede, MCM işlemi için gereken minimum sayıdaki CSA bloklarını bulan bir kesin ortak alt ifade eliminasyonu (CSE) algoritması önerilmektedir. Önerilen kesin yöntem gerçek boyutlu örnekler üzerinde uygulanabilse de, MCM probleminin bir NP-bütün problem olmasından dolayı, doğal olarak, kesin algoritmanın ele alamayacağı örnekler mevcuttur. Bu yüzden, bu makalede, büyük boyutlu örnekleri ele alabilen bir yaklaşık CSE yöntemi de önerilmekedir. CSE algoritmaları ile elde edilen sonuçlar katsayıların gerçeklenmesi için kullanılan sayı gösterimlerine bağlı olduğundan bu makalede, ayrıca, katsayıların genel sayı gösterimindeki ifadelerini ele alabilen bir yaklaşık yöntem de sunulmaktadır. Önerilen algoritmalar, rastgele üretilmiş örnekler ve FIR filtreleri üzerinde test edilmiş ve daha önceden önerilmiş CSE ve graf tabanlı sezgisel yöntemler ile karşılaştırılmıştır. Deneysel sonuçlardan kesin CSE algoritmanın sezgisel CSE yönteme göre oldukça iyi sonuçlar elde ettiği ve genel sayı gösterimi altında yaklaşık algoritmanın graf tabanlı sezgisel yöntemle rekabet edebilecek düzeyde ve daha iyi sonuçlar bulduğu gözlenmektedir.

Anahtar Kelimeler: *Birden fazla katsayının çarpımı problemi, ortak alt ifade eliminasyonu, elde ötelemeli toplama, elde korumalı toplama, 0-1 tamsayı lineer programlama.*

*Yazışmaların yapılacağı yazar: Levent AKSOY. aksoyl@itu.edu.tr; Tel: (212) 285 67 33.
Bu makale, birinci yazar tarafından İTÜ Fen Bilimleri Enstitüsü, Elektronik Mühendisliği Programı'nda tamamlanmış olan "Optimization algorithms for the multiple constant multiplications problem" adlı doktora tezinden hazırlanmıştır. Makale metni 11.03.2009 tarihinde dergiye ulaşmış, 25.03.2009 tarihinde basım kararı alınmıştır. Makale ile ilgili tartışmalar 31.05.2010 tarihine kadar dergiye gönderilmelidir.

Area optimization algorithms in high-speed digital FIR filter synthesis

Extended abstract

Digital Finite Impulse Response (FIR) filters are frequently used in Digital Signal Processing (DSP) by virtue of stability and easy implementation. The problem of designing the multiplier block of a digital FIR filter has received a significant amount of attention during the last decade, as the filters require a large number of multiplications, leading to excessive area, delay, and power consumption even if implemented in a full custom integrated circuit. Previous works have focused on the design of filters with minimum area by replacing the multiplication operations with constant coefficients by addition, subtraction, and shifting operations. Since shifts can be implemented with only wires in hardware, the design problem can be defined as the minimization of the number of addition/subtraction operations to implement the coefficient multiplications. This problem is generally known as the MCM problem.

In the last decade, many efficient algorithms have been proposed for the MCM problem. These algorithms can be categorized in two classes: Common Subexpression Elimination (CSE) and graph-based algorithms. While the CSE algorithms basically find the common non-zero digit patterns on the representations of the constants, the graph-based algorithms are not restricted to a particular number representation and synthesize the constants iteratively by building a graph.

However, in the algorithms proposed for the MCM problem, an addition/subtraction operation is assumed to be a two-input operation that is generally implemented using a Ripple Carry Adder (RCA) block yielding great latency in the implementation of MCM. In high-speed applications, particularly in DSP systems, Carry-Save Adder (CSA) blocks are preferred to RCA blocks taking into account the increase in area.

Despite the large number of algorithms designed for the minimization of addition/subtraction operations based on RCAs, there are only a few algorithms proposed for the optimization of the number of CSA blocks. Although these algorithms give good results, they are based on heuristics, i.e., provide no indication on how far from the minimum their solutions are. To the best of our knowledge, there is no exact algorithm proposed for the optimization of the number of CSA blocks in MCM.

In this paper, an exact CSE algorithm designed for the minimization of the number of CSA blocks is introduced. In the exact CSE algorithm, initially, all possible implementations of filter coefficients and partial terms using CSAs are found when filter coefficients are defined under a number representation, and a combinational network that represents the implementations of constants is constructed. Then, the minimization of the number of CSA blocks problem is defined as a 0-1 Integer Linear Programming (ILP) problem with a cost function to be minimized and constraints to be satisfied. Finally, a generic 0-1 ILP solver is used to obtain the minimum solution.

Due to the NP-completeness of the MCM problem, naturally, there are instances that the exact algorithm cannot handle. Hence, in this paper, we also introduce an approximate CSE algorithm based on the exact CSE algorithm that considers limited implementations of the coefficients reducing the size of the search space to be explored significantly. It is also shown that the approximate CSE algorithm obtains similar results with the exact CSE algorithm.

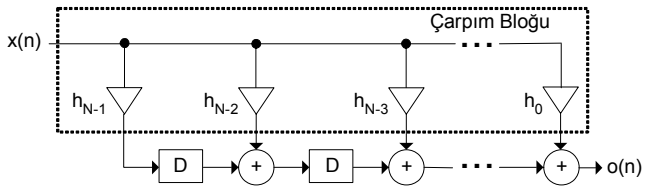
Since the solutions obtained by the proposed CSE algorithms depend on the number representation, in the approximate algorithm, the number of possible implementations of filter coefficients is increased using a general number representation allowing the approximate algorithm to be more effective in area optimization. It is shown that the approximate algorithm under general number representation obtains more promising results than the approximate CSE algorithm.

In this paper, the results of the proposed exact and approximate algorithms on a comprehensive set of instances including randomly generated and FIR filter instances are presented. It is shown that the proposed algorithms can be applied on real size instances. Also, the results of the exact and approximate algorithms are compared with those of the previously proposed CSE and graph-based heuristics. As observed from the experimental results, the exact and approximate CSE algorithms find significantly better solutions than the CSE heuristic and the approximate algorithm under general number representation obtains competitive and better results than the graph-based heuristic.

Keywords: *Multiple constant multiplications, common subexpression elimination, ripple carry adders, carry save adders, 0-1 integer linear programming.*

Giriş

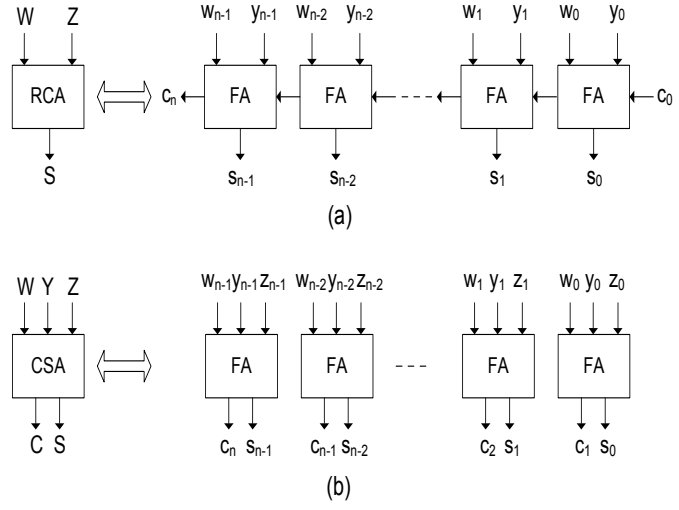
Sonlu impuls cevaplı (Finite Impulse Response, FIR) filtrelerde, Şekil 1'de gösterildiği gibi, aynı giriş birden fazla katsayı ile çarpılmaktadır. Bu işlem, genel olarak birden fazla katsayının çarpımı (Multiple Constant Multiplications, MCM) olarak bilinir. Bu işlemler sayısal işaret işleme (Digital Signal Processing, DSP) uygulamalarında sıklıkla kullanılır ve donanım tabanlı mimariler maksimum başarım ve minimum güç tüketimi için en iyi seçenektir. Sayısal filtreler birçok sayıda çarpma işlemi gerektirdiğinden ve bu çarpma işlemleri, tümleşik devrelerde gerçeklense bile, alan, gecikme ve güç tüketimi açısından oldukça maliyetli olduklarından, sayısal FIR filtrenin çarpma bloğunun tasarımı problemi son on yılda oldukça ilgi çekmiştir. Önceki çalışmalar FIR filtreleri minimum alan içinde tasarlarken sabit katsayılar ile çarpma işlemini toplama, çıkarma ve öteleme işlemleri ile gerçeklemeye odaklanmışlardır (Nguyen ve Chatterjee, 2000). Donanım içinde öteleme işlemleri sadece bağlantılar ile gerçeklenebileceğinden dolayı, bu problem, katsayı çarpımlarını gerçeklemek için gereken toplama/çıkarma işlem sayısının minimize edilmesi olarak tanımlanabilir. Bu problem, genel olarak MCM problemi olarak adlandırılır ve belirleyici olmayan polinom bütün (Nondeterministic Polynomial time, NP, complete) bir problem olduğu Cappello ve Steiglitz (1984)'te ispatlanmıştır.



Şekil 1. Devrik formda sayısal FIR filtre gerçeklemesi

MCM içindeki işlem sayısının optimizasyonu için tasarlanan algoritmalar iki kısma ayrılabilir: Ortak alt ifade eliminasyonu (Common Subexpression Elimination, CSE) yöntemleri, Hartley (1996) ve Pasko ve diğerleri (1999), ve graf tabanlı katsayı gerçekleme teknikleri, (Dempster ve MacLeod, 1995) ve (Voronenko ve Püschel, 2007). CSE algoritmaları temel olarak katsayıların gösterimindeki sıfırdan farklı ortak basamak örüntülerini bulurlar. Kesin CSE algoritmaları, Gustafsson ve Wanhammar (2002) ve Flores ve diğerleri (2005), MCM problemini bir 0-1 tamsayı lineer programlama (Integer Linear Programming, ILP) problemi olarak formülleştirirler ve MCM probleminin minimum sayıda toplama ve çıkarma işlemi içeren çözümünü ara terimlerin paylaşımını maksimize ederek bulurlar. Graf tabanlı algoritmalar ise bir sayı gösterimine bağlı değillerdir ve katsayıları bir graf inşa ederek gerçeklerler.

MCM problemi için önerilmiş yöntemlerde, bir toplama/çıkarma işlemi iki girişli bir işlemdir ve genel olarak, MCM tasarımındaki gecikmeyi oldukça arttıran elde ötelemeli toplama (Ripple Carry Adder, RCA) blokları ile gerçeklenir (Johansson vd, 2005), (Aksoy vd, 2007a). Yüksek hızlı uygulamalarda, özellikle DSP sistemlerde, elde korumalı toplama (Carry Save Adder, CSA) blokları, alan içinde artışı göz önüne alarak, RCA bloklarına göre tercih edilir. Bir CSA bloğun üç adet girişi ve iki adet çıkışı, toplam (Sum, S) ve elde (Carry, C), vardır. Bu iki çıkış, toplama sonucunu oluşturur. Bir *n*-bitlik CSA bloğu *n* adet tam toplayıcı (Full Adder, FA) içerir. Eldenin RCA bloğunda olduğu gibi ötelenmesi gerekmediğinden, Şekil 2(a), toplama işleminin gecikmesi bir tam toplayıcının kapı gecikmesine eşittir ve girişlerin boyutundan bağımsızdır, Şekil 2(b).



Şekil 2. (a) Elde ötelemeli toplama bloğu (b) Elde korumalı toplama bloğu

RCA blokları ile gerçeklenecek toplama veya çıkarma işlem sayısının minimize edilmesi için tasarlanmış çok sayıda yöntem olmasına rağmen, CSA blok sayısının optimizasyonu için sadece birkaç algoritma, Hosangadi ve diğerleri (2006) ve Gustafsson ve diğerleri (2004), önerilmiştir. Bu algoritmalar iyi sonuçlar elde etmelerine rağmen, sezgisel yöntemlere dayanmaktadırlar ve bu yüzden, sonuçlarının minimumdan ne kadar uzak olduklarını gösteren bir bilgi sağlamamaktadırlar. Bildiğimiz kadarıyla bugüne kadar CSA blok sayısının optimizasyounu için kesin bir algoritma önerilmemiştir.

Bu makalede, CSA blok sayısını minimize eden bir kesin CSE algoritması sunulmaktadır. Bu kesin CSE algoritmada, ilk olarak, filtre katsayıları bir sayı gösteriminde ifade edilir ve filtre katsayılarının ve ara terimlerin bütün olası CSA blok gerçeklemeleri bulunur. Daha sonra, katsayıların bütün gerçeklemelerini gösteren bir kombinezonsal Boolean devresi inşa edilir. Bundan sonra, CSA blok sayısının minimize edilmesi problemi, minimize edilecek hedef fonksiyonu ve sağlanması gerekecek kısıtları ile bir 0-1 ILP problemi olarak tanımlanır. Son olarak, minimum sonuç genel bir 0-1 ILP çözücü kullanılarak bulunur.

Ayrıca, bu makalede, büyük boyutlu örnekleri ele alabilen kesin CSE algoritma tabanlı bir yaklaşık CSE algoritma sunulmaktadır. Önerilen CSE algoritmaların sonuçları, kullanılan sayı gösterimine bağlı kaldığından, katsayıların genel sayı gösterimindeki gerçeklemelerini ele alabilen ve böylelikle alan optimizasyonunda oldukça etkili olan bir yaklaşık yöntem de sunulmaktadır.

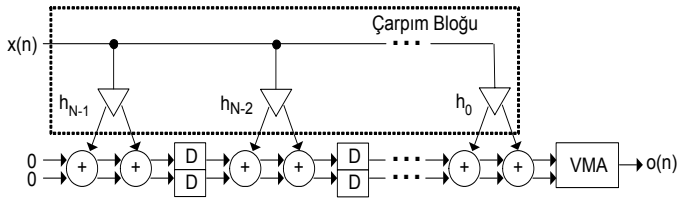
Bu makalede önerilen yöntemler rastgele üretilmiş örnekler ve FIR filtreler üzerinde uygulanmış ve daha önceden önerilmiş CSE ve graf tabanlı yöntemler ile karşılaştırılmıştır. Deneysel sonuçlar, kesin ve yaklaşık CSE algoritmaların sezgisel CSE yöntemden oldukça iyi sonuçlar bulduğunu ve genel sayı gösterimi altındaki yaklaşık algoritmanın graf tabanlı sezgisel yöntem ile rekabet edecek düzeyde ve daha iyi sonuçlar elde ettiğini göstermektedir.

Tanımlar

Bu bölümde, ilk olarak, problem tanımı verilmekte ve sayı gösterimleri tanıtılmaktadır. Daha sonra, bu konu üzerinde yapılan çalışmalar özetlenmektedir.

Problem tanımı

Bu makalede önerilen yöntemler MCM içeren herhangi bir sisteme uygulanabilse de, bu algoritmalar Şekil 3'te gösterilen yüksek hızlı sayısal FIR filtrenin çarpım bloğunun gerçeklenmesi üzerinde gösterilecektir. Şekil 3'te, her bir toplama işlemi bir CSA bloğunu gösterirken filtre çıkışı bir vektör birleştirmeli toplama (Vector Merging Adder, VMA) bloğu ile elde edilmektedir. Böylece, *CSA blok sayısının minimize edilmesi problemi,* verilen filtre katsayıları ile sayısal FIR filtrenin çarpım bloğundaki katsayı çarpımlarının gerçeklenmesi için gereken minimum sayıdaki CSA bloğun bulunması olarak ifade edilebilir.



Şekil 3. Yüksek hızlı sayısal FIR filtrenin devrik formu

Sayı gösterimleri

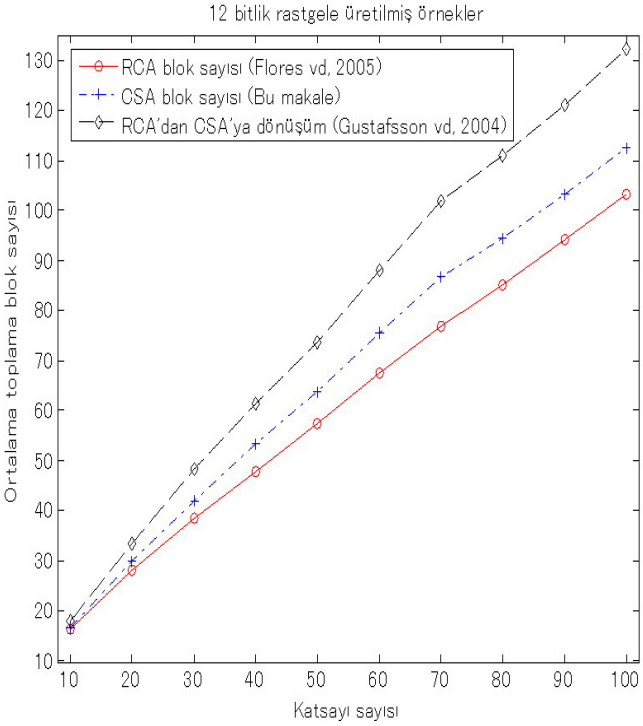
Kanonik işaretli basamak (Canonical Signed Digit, CSD) gösterimi, -1'in $\bar{1}$ ile gösterildiği $\{1,0,\bar{1}\}$ basamak kümesi ile bir işaretli sayı sistemidir. Her bir tamsayının CSD gösterimi tektir ve iki adet temel özelliğe sahiptir: i) Sıfırdan farklı basamak sayısı minimumdur. ii) Sıfırdan farklı basamaklar birbirine komşu değildir. CSD gösterimde bir katsayı minimum sayıda sıfırdan farklı basamak ile ifade edildiğinden, bu gösterim çarpma işlemleri kullanılmadan gerçeklenen tasarımlarda sıklıkla tercih edilir. Minimal işaretli basamak (Minimal Signed Digit, MSD) gösterimi, CSD gösteriminin ikinci özelliğinin göz önüne alınmaması ile elde edilir. Bundan dolayı, bir katsayı birden fazla MSD gösterimine sahip olabilir, fakat bütün hepsi minimum sayıda sıfırdan farklı basamak içerir. Örnek ola-

rak, altı bit ile tanımlanan 23 tamsayısını ele alalım. 23'ün ikili tabanda gösterimi, 010111, dört adet sıfırdan farklı basamak içerir. Bunun yanında, 23, CSD gösterimde $10\bar{1}00\bar{1}$ olarak ifade edilir ve her iki, $10\bar{1}00\bar{1}$ ve $01100\bar{1}$, gösterimleri MSD içinde 23'ü minimum sayıda sıfırdan farklı basamak kullanarak ifade eder.

İlgili çalışmalar

Kim ve diğerleri (1998)'de, bir işlem kümesini toplama işlemlerine ve bu toplama işlemlerini bir CSA ağaç yapısına dönüştüren bir yöntem önerilmiştir. Verna ve Ienne (2004)'te, bir kompleks aritmetik devresinin akış grafını yeniden biçimlendirip, bu grafı, CSA blokları içeren yüksek kaliteli aritmetik devre olarak sentezleyen bir algoritma sunulmuştur. Yine de, bu yöntemler CSA blok sayısının minimize edilmesine odaklanmazlar. Bir sayısal FIR filtre içindeki MCM işlemlerini gerçeklemek için kullanılan RCA bloklarını CSA bloklarına dönüştüren benzer bir yaklaşım ise Gustafsson ve diğerleri (2004)'te verilmiştir. Buna göre, üç farklı durum ile karşılaşılabilir: i) Eğer bir RCA bloğun tüm girişleri filtre girişi ise, CSA blok kullanmaya gerek yoktur. Çünkü, bu iki giriş toplam ve elde girişleri olarak gösterilebilir. ii) Eğer bir RCA bloğun sadece bir adet girişi filtre girişi ise, sadece bir adet CSA bloğuna gerek vardır. iii) Eğer bir RCA bloğun iki girişi de filtre girişi değil ise, iki adet CSA bloğuna ihtiyaç vardır.

Şekil 4'te, bu makalede önerilen kesin CSE algoritması ile elde edilen sonuçlar, Gustafsson ve diğerleri (2004)'te verilen RCA'dan CSA'ya dönüşüm tekniği ile elde edilen sonuçlar ile karşılaştırılmaktadır. Minimum sayıda RCA blok içeren çözümler ise Flores ve diğerleri (2005)'te önerilen kesin CSE algoritması ile elde edilmiştir. Bu deneyde, 12 bitlik katsayılar rastgele üretilmiş ve CSD gösterimi altında tanımlanmışlardır. Katsayı sayıları 10 ile 100 arasında değişmekte ve her biri 30 adet örnek içermektedir. Şekil 3'ten görülebildiği gibi, RCA'dan CSA'ya dönüşüm, minimum sayıdaki CSA blok sonuçlarından oldukça uzak sonuçlar vermektedir. Ayrıca, katsayı sayısı arttıkça bu dönüşüm ile kesin CSE algoritmanın sonuçları arasındaki farkın da arttığı gözlemlenmektedir. Bu deney, optimizasyon sırasında CSA blok mimarisini göz önüne almanın minimum alan sonuçlarını elde etmek için kaçınılmaz olduğunu açıkça göstermektedir.



Şekil 4. RCA ve CSA blok sayısı çözümleri ile RCA'dan CSA'ya dönüşüm yöntemi ile elde edilen sonuçların karşılaştırılması

CSA blok sayısını optimize etmek için önerilen yöntemler ise iki kısma ayrılabilir: CSE ve graf tabanlı algoritmalar. Sezgisel CSE algoritması, Hosangadi ve diğerleri (2006), ilk olarak, katsayı çarpımlarını ifadeler içinde tanımlar. Sonra, tekrarlamalı olarak, bu ifadelerden bütün olası üç-terimli bölenleri çıkartır, bu bölenler arasından en iyi böleni, yani en fazla ortak olanı, bulur. Bu böleni iki terimle, yani bir CSA bloğunun toplam ve elde çıkışları, ile tanımlayıp bu iki terimi ifadeler içine yerleştirerek ifadeleri tekrar biçimlendirir. Graf tabanlı sezgisel algoritma, Gustafsson ve diğerleri (2004), ise optimal ve sezgisel olarak iki kısma ayrılır. Optimal kısımda, sadece bir adet CSA blok ile gerçeklenebilen katsayılar gerçeklenir. Optimal kısımdan sonra hala gerçeklenmemiş katsayılar kalmış ise algoritma sezgisel aşamaya geçer. Bu aşamada ise gerçeklenmemiş bir katsayı, iki adet CSA bloğu ile sentezlenir veya Gustafsson ve diğerleri

(2001)'de önerilen yöntem ile belirlenmiş minimum sayıda CSA blok ile gerçeklenir. Her iki durumda da ara terimler gerçeklenmek üzere ilave edilir. Gustafsson ve diğerleri (2004)'te, graf tabanlı yöntemin sezgisel CSE yöntemlerden daha iyi sonuçlar verdiği gösterilmiştir. Bunun nedeni, graf tabanlı yöntemde bir katsayı için ele alınan gerçekleme sayısının bir CSE yönteme göre daha fazla olması olarak açıklanmıştır.

Kesin CSE algoritması

Bu bölümde, katsayıları ikili, CSD ve MSD sayı gösterimlerinde ele alabilen bir kesin CSE algoritması sunulmaktadır. Algoritma üç ana kısma ayrılmaktadır: i) Filtre katsayılarının ve ara terimlerin CSA blokları kullanılarak bütün olası gerçeklemelerinin bulunması. ii) Katsayıların gerçeklemelerini gösteren Boolean devresinin inşası. iii) CSA blok sayısının minimize edilmesi probleminin bir 0-1 ILP problemi olarak formülleştirilmesi.

İşlemlerin bulunması

Algoritmanın ön hazırlık aşamasında, ilk olarak, *Cset* olarak adlandırılan bir boş küme oluşturulur, 1 ile gösterilen filtre girişi *Cset* kümesine eklenir ve gerçeklenmiş olarak etiketlenir. Daha sonra, filtre katsayıları pozitif ve tek sayıya dönüştürülür. Elde edilen katsayılar tekrarlanmadan *Cset* içine atılır ve gerçeklenmemiş olarak etiketlenir. Bu aşamada, *Cset*, filtre girişini ve gerçeklenecek tekrarsız pozitif ve tek katsayıları içermektedir. Aşağıda verilen tekrarlamalı çevrimde, *Cset* içindeki gerçeklenmemiş olarak etiketlenen her bir eleman için, bu katsayıyı gerçekleyen işlemler bulunur ve bu işlemlerin maliyet değerleri gerekli olan CSA blok sayısı olarak belirlenir.

1. *Cset* kümesinden gerçeklenmemiş olarak etiketlenmiş bir eleman, $Cset_i$, al ve verilen sayı gösterimi altında onun gösterimlerini bul. $Oset_i$ boş kümesini oluştur. Bu küme, $Cset_i$'nin gerçeklenmesi için gerekli olan işlemleri, bu işlemlerin pozitif ve tek girişleri cinsinden içerir.
2. Eğer $Cset_i$'nin gösterimi 2 adet sıfırdan farklı basamak içeriyorsa, $Cset_i$'yi gerçeklenmiş olarak etiketle ve 6. adıma git. Bu durumda $Cset_i$'nin gerçeklenmesi için bir işleme gerek duyulmaz.
3. Eğer $Cset_i$'nin gösterimi 3 adet sıfırdan farklı basamak içeriyorsa, bu işlemin pozitif ve tek girişlerini $Oset_i$ içinde sakla ve işlemin maliyet değerini 1 olarak belirle. $Cset_i$'yi gerçeklenmiş olarak etiketle ve 6. adıma git. Bu durumda, $Cset_i$ sadece bir adet CSA blok ile gerçeklenebilir ve bu da minimum maliyetli gerçeklemedir.
4. Aksi halde, $Cset_i$'nin her bir gösteriminde, $Cset_i$'yi gerçekleyen bütün olası işlemleri bul. Bu işlemlerin pozitif ve tek girişlerini $Oset_i$ içinde sakla ve işlemlerin maliyet değerlerini belirle.
5. $Cset_i$'yi gerçeklenmiş olarak etiketle, $Oset_i$'nin bütün elemanlarını *Cset*'e tekrarlamadan ekle ve gerçeklenmemiş olarak etiketle.
6. *Cset* kümesindeki bütün elemanlar gerçeklenmiş olarak etiketlenene kadar 1. adımı tekrarla.

Bu tekrarlamalı çevrimin başlangıcında, *Cset*, filtre girişini ve pozitif ve tek filtre katsayılarını içerirken, ileriki aşamalarda, aynı zamanda katsayıların gerçeklenmesi için gerekli olan ara terimleri de içerecektir. Üçten fazla sayıda sıfırdan farklı basamak içeren katsayıları gerçekleştiren işlemlerin bulunmasında, yani tekrarlı çevrimin 4. adımında ise sıfırdan farklı katsayılar iki kısma ayrılır. Bunlardan biri, ikiden fazla sıfırdan farklı basamak içerirken, diğeri, bir veya ikiden fazla sıfırdan farklı basamak içerir. İkiden fazla sıfırdan farklı basamak içeren bölmeleme bir ara terimi oluşturur. Ara terimler bir CSA bloğu ile gerçekleneceklerinden dolayı toplam ve elde çıkışları ile gösterilirler. Eğer, bir işlemin girişlerinden biri filtre girişi ise bu işlemin maliyet değeri 1 CSA blok olarak belirlenir. Eğer işlemin tüm girişleri de filtre girişi değil ise, bu işlemin maliyeti değeri 2 CSA bloktur.

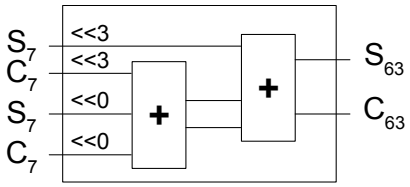
Örnek olarak, 51'in bir filtre katsayısı olarak verildiğini ve CSD gösteriminde tanımlandığını, $10\bar{1}010\bar{1}$, varsayalım. 51'in gerçeklemeleri Şekil 4'te verilmektedir.

$$I_1 : S\&C_{51} = 10\bar{1}0100 + 000000\bar{1} = 13_{<<2} - 1 = S\&C_{13<<2} - 1$$
$$I_2 : S\&C_{51} = 10\bar{1}000\bar{1} + 0000100 = 47 + 1_{<<2} = S\&C_{47} + 1_{<<2}$$
$$I_3 : S\&C_{51} = 100010\bar{1} + 00\bar{1}0000 = 67 - 1_{<<4} = S\&C_{67} - 1_{<<4}$$
$$I_4 : S\&C_{51} = 00\bar{1}010\bar{1} + 1000000 = -13 + 1_{<<6} = -S\&C_{13} + 1_{<<6}$$

Şekil 4. CSD gösterimde 51'in CSA blokları kullanılarak gerçeklemeleri

Şekil 4'te, ara terimler, 13, 47 ve 67, üç adet sıfırdan farklı basamak içerdiklerinden toplam ve elde çıkışları ile gösterilmişlerdir. Bütün işlemlerin girişlerinden biri filtre girişi, yani 1, olduğundan, her bir işlemin maliyet değeri 1 CSA bloğudur. Bunun yanında, I_4 gerçeklemesi fazlalıktır, çünkü I_1 gerçeklemesi gibi aynı pozitif ve tek girişleri içerir ve her ikisinin de maliyet değeri 1 CSA bloğudur.

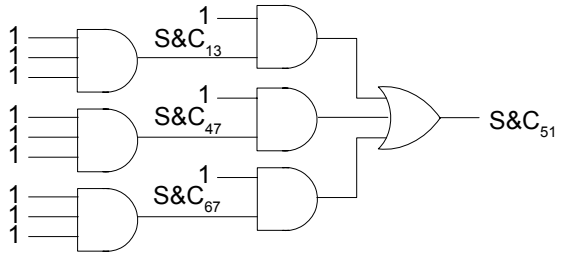
2 adet CSA bloğu içeren bir işleme örnek olarak, 63 katsayının ikili tabanda gösterimini, 111111, ele alalım. $S\&C_{63}$'ü gerçekleştiren işlemlerden biri, $S\&C_{7<<3}+S\&C_7$, Şekil 5'te gösterildiği gibi 2 adet CSA bloğu içermektedir.



Şekil 5. 63'ün ikili gösterimde 2 adet CSA bloğu kullanılarak gerçeklenmesi

Boolean devresi

Filtre katsayılarının ve ara terimlerin bütün olası gerçeklemeleri bulunduktan sonra AND ve OR kapıları içeren bir Boolean devresi inşa edilir. Devrenin asal girişleri filtre girişidir. Devre içindeki bir AND kapısı CSA blok(ları) içeren bir işlemi gösterir. Bir AND kapısının çıkışı ise katsayıyı gerçekleyen CSA bloğunun toplam ve elde çıkışlarını gösterir. Filtre katsayısı veya ara terimler ile ilişkilendirilmiş OR kapıları, bu katsayıları gerçekleyen bütün işlemleri bir araya getirir. Bir OR kapısının çıkışı, katsayıyı ifade eden toplam ve elde çıkışlarını gösterir. Devrenin asal çıkışları filtre katsayıları ile ilişkilendirilmiş OR kapılarının çıkışlarıdır. CSD gösterimi altında 51 katsayısı için oluşturulmuş Boolean devresi Şekil 6'da verilmektedir. Bu devreye 13, 47 ve 67 ara terimleri için gereken 1 girişli OR kapıları ve Şekil 4'te gösterilen I_4 fazlalık işlemi dahil edilmemiştir. Şekil 6'da görüldüğü gibi, 51, CSD gösterimde iki adet CSA bloğu kullanılarak gerçeklenir ve oluşturulan Boolean devresi 51'in CSD gösteriminden çıkartılabilecek bütün olası işlemleri, bütün olası ara terimler ile gösterir.



Şekil 6. CSD gösteriminde 51 için inşa edilmiş Boolean devresi

CSA blok sayısının minimize edilmesi problemini bir 0-1 ILP problemine dönüştürürken, minimize edilecek hedef fonksiyonunu oluşturabilmek için devre içerisine optimizasyon değişkenlerini de eklemek gerekir. Bunu yapabilmek için, optimizasyon değişkenleri işlemlerle ilişkilendirilmiştir (Aksoy vd, 2008). Böylece, devre içinde her bir AND kapısına optimizasyon değişkenini ifade eden bir ek giriş eklenmiştir. Optimizasyon değişkenlerinin işlemler ile ilişkilendirilmesi ile 0-1 ILP çözücünün elde edeceği minimum sonuç, doğrudan gerçeklenmesi gereken işlemi gösterecektir.

0-1 ILP problemine dönüştürme

Boolean devresi elde edildikten sonra CSA blok sayısının minimize edilmesi probleminin bir 0-1 ILP problemine dönüştürülmesi doğrudan gerçeklenir. 0-1 ILP probleminin hedef fonksiyonu, işlemler ile ilişkilendirilmiş optimizayon değişkenlerinin bir lineer fonksiyonudur, öyle ki, her bir optimizasyon değişkenin katsayısı, her bir işlem için gerekli olan CSA blok sayısıdır. 0-1 ILP probleminin kısıtları ise, devre içindeki her bir kapının toplamlar çarpımı formundaki (Conjunctive Normal Form, CNF) formüllerin bulunması ve CNF formüllerindeki her bir çarpımın Barth (1995)'te verildiği gibi bir lineer eşitsizlik olarak ifade edilmesi ile belirlenir. Bu

kısıtlara ek olarak, amacımız filtre katsayılarını gerçekleştirmek olduğundan, filtre katsayıları ile ilişkilendirilmiş OR kapılarının çıkışlarına 1 değeri atanır. Böylelikle, elde edilen model bir genel 0-1 ILP çözücüsüne giriş olarak verilebilir ve minimum sonuç elde edilebilir.

Yaklaşık algoritmalar

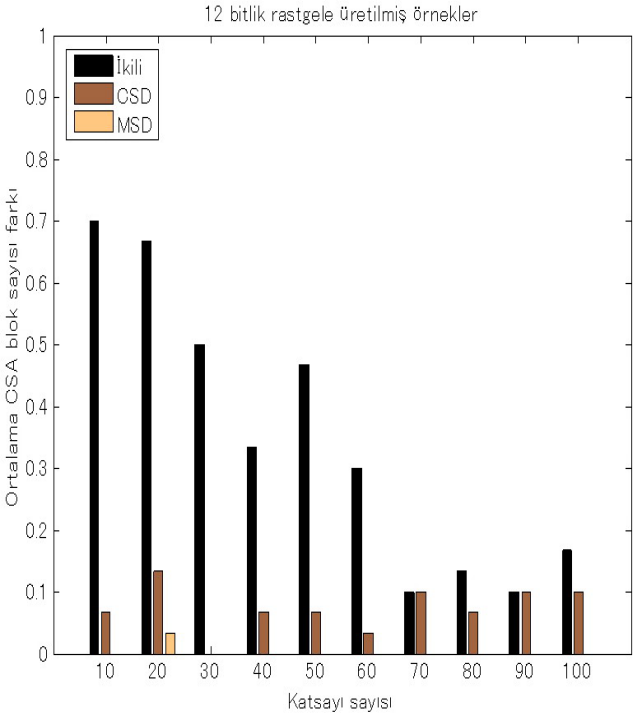
Bu bölümde, ilk olarak, yaklaşık CSE algoritması sunulmakta ve sonra, katsayıları genel sayı gösteminde ele alabilen bir yaklaşık algoritma verilmektedir.

Yaklaşık CSE algoritması

Bir önceki bölümde verilen kesin CSE algoritması deneysel sonuçlar bölümünde de gösterildiği gibi gerçek boyutlu örnekler üzerinde uygulanabilse de, kesin algoritmanın ele alamayacağı çok daha karmaşık örnekler doğal olarak mevcuttur. Çünkü, 0-1 ILP çözücünün minimum sonucu bulması için gereken süre, 0-1 ILP probleminin boyutunun büyümesi ile artmaktadır. Yine de, 0-1 ILP probleminin boyutu, katsayıların gerçeklemelerini bulurken sadece 1 adet CSA bloğu gerektiren işlemler, yani girişlerinden birinin filtre girişi olduğu işlemler, göz önüne alınarak oldukça azaltılabilir. Önerilen bu yaklaşık CSE algoritması, kesin CSE algoritmasından sadece katsayı gerçeklemelerinin bulunması aşamasında farklılık göstermektedir.

Şekil 7'de, 12 bitlik rastgele üretilmiş örnekler üzerinde yaklaşık ile kesin CSE algoritma sonuçları arasındaki ortalama CSA blok sayısı farkı gösterilmiştir. Bu deneyde, katsayı sayıları 10 ile 100 arasında değişmekte ve her biri 30 adet örnek içermektedir.

Şekil 7'den görülebildiği gibi yaklaşık algoritma ile elde edilen sonuçlar kesin algoritma ile bulunan sonuçlara oldukça yakındır (ortalama 0.7 CSA bloktan daha az). Bunun nedeni ise, kesin algoritmada 2 adet CSA blok gerektiren işlemler nadiren gerçeklenmektedir. Katsayı sayısı arttığında yaklaşık ile kesin algoritma sonuçları arasındaki fark da azalmaktadır. Bunun nedeni ise, katsayı sayısının artması ile ara terimlerin paylaşımı da artmaktadır. Ayrıca, MSD gösterimi altında yaklaşık algoritma, toplam 300 örnekte, kesin algoritmadan sadece bir adet fazla CSA bloğu kullanmaktadır. Bunun nedeni ise, MSD gösteriminde, ikili ve CSD sayı gösterimlerine göre, bir katsayı birden fazla gösterimle ifade edilebildiğinden dolayı, olası gerçekleme ve ara terim sayısı da artmaktadır.



Şekil 7. Rastgele üretilmiş örnekler üzerinde yaklaşık ile kesin CSE algoritma sonuçlarının farkı

Genel sayı gösterimi altında yaklaşık algoritma

CSE algoritmaların sonuçları sayı gösterimlerine bağlı olduğundan Aksoy ve diğerleri, (2007b)'de gösterildiği gibi minimum sayıdaki işlem sonuçları bir kesin CSE algoritması kullanılarak elde edilemeyebilir. Katsayılar için nümerik değerler kullanılarak arama uzayı oldukça genişletilebilir ve önerilen yaklaşık algoritma alan optimizasyonu için daha etkili olabilir.

Yaklaşık CSE algoritması, katsayıları genel sayı gösteriminde ele alacak şekilde Aksoy ve diğerleri (2007b)'de verildiği gibi genişletilmiştir. Önerilen bu algoritmada, ilk olarak, 1 ve 2^{bw+1}-1 arasındaki pozitif ve tek sayılar, *Nset* adlı küme içinde CSD gösterimi altında içerdikleri sıfırdan farklı basamak sayısına göre sıralanmıştır. Bu-

rada, *bw*, filtre katsayılarının maksimum bit genişliğini göstermektedir. Daha sonra, her bir katsayı için bu katsayıyı gerçekleyen işlemler, işlemin birinci girişine filtre girişinin ve onun ötelenmiş hallerinin pozitif ve negatif işaretli olarak atanması ve katsayının gerçeklenmesi için gereken ara terimlerin bulunması ile elde edilir. Ara terimi gerçekleyen CSA bloğunun toplam ve elde çıkışları ise işlemin diğer iki girişine atanır. Bir işlem, eğer ara terimleri bu katsayının *Nset* içindeki konumundan daha önce yer alıyorsa, geçerli bir işlem olarak belirlenir. Bunun nedeni, Aksoy ve diğerleri (2007b)'de açıklandığı gibi elde edilecek çözümün yönlü çevrimsiz graf olarak garanti edilmesi gerektiğindendir. Örnek olarak, yine 51'i filtre katsayısı olarak ele alalım. Genel sayı gösterimi altında 51'in bütün gerçeklemeleri Şekil 8'de verilmektedir.

$$I_1 : S \,\&\, C_{51} = 1 + 25_{<<1} = 1 + S \,\&\, C_{25<<1}$$
$$I_2 : S \,\&\, C_{51} = -1 + 13_{<<2} = -1 + S \,\&\, C_{13<<2}$$
$$I_3 : S \,\&\, C_{51} = 1_{<<1} + 49 = 1_{<<1} + S \,\&\, C_{49}$$
$$I_4 : S \,\&\, C_{51} = -1_{<<1} + 53 = -1_{<<1} + S \,\&\, C_{53}$$
$$I_5 : S \,\&\, C_{51} = 1_{<<2} + 47 = 1_{<<2} + S \,\&\, C_{47}$$
$$I_6 : S \,\&\, C_{51} = -1_{<<2} + 55 = -1_{<<2} + S \,\&\, C_{55}$$
$$I_7 : S \,\&\, C_{51} = 1_{<<3} + 43 = 1_{<<3} + S \,\&\, C_{43}$$
$$I_8 : S \,\&\, C_{51} = -1_{<<3} + 59 = -1_{<<3} + S \,\&\, C_{59}$$
$$I_9 : S \,\&\, C_{51} = 1_{<<4} + 35 = 1_{<<4} + S \,\&\, C_{35}$$
$$I_{10} : S \,\&\, C_{51} = -1_{<<4} + 67 = -1_{<<4} + S \,\&\, C_{67}$$
$$I_{11} : S \,\&\, C_{51} = 1_{<<5} + 19 = 1_{<<5} + S \,\&\, C_{19}$$
$$I_{12} : S \,\&\, C_{51} = -1_{<<5} + 83 = -1_{<<5} + S \,\&\, C_{83}$$
$$I_{13} : S \,\&\, C_{51} = 1_{<<6} - 13 = 1_{<<6} - S \,\&\, C_{13}$$
$$I_{14} : S \,\&\, C_{51} = -1_{<<6} + 115 = -1_{<<6} + S \,\&\, C_{115}$$

Şekil 8. Genel sayı gösterimi altında 51'in CSA blokları kullanılarak gerçeklemeleri

Şekil 8'de verilen işlemlerden I_4, I_{12} ve I_{14}, 51'in gerçeklenmesi için kabul edilmemiştir. Çünkü, bu işlemlerdeki ara terimler, 53, 83 ve 115 *Nset* içinde 51'in konumundan ötede bulunmaktadır. Ayrıca, I_{13} işlemi fazlalıktır, çünkü I_2 işlemi gibi aynı pozitif ve tek girişlere sahiptir. Böylece, 51, genel sayı gösteriminde 10 farklı şekilde gerçeklenebilir ve bu farklı ara terimler MCM içinde ara terimlerin paylaşımını arttırabilir. Bu örnekten, genel sayı gösteriminin kullanılması ile herhangi bir sayı gösterimi altında yaklaşık CSE algoritması ile elde edilen bütün gerçeklemelerin göz önüne alındığı ve bunun ötesinde, katsayıların bir sayı gösterimi altında sıfırdan farklı basamak kombinasyonları ile elde edilemeyen işlemlerin bulunabildiği kolaylıkla gözlenebilir.

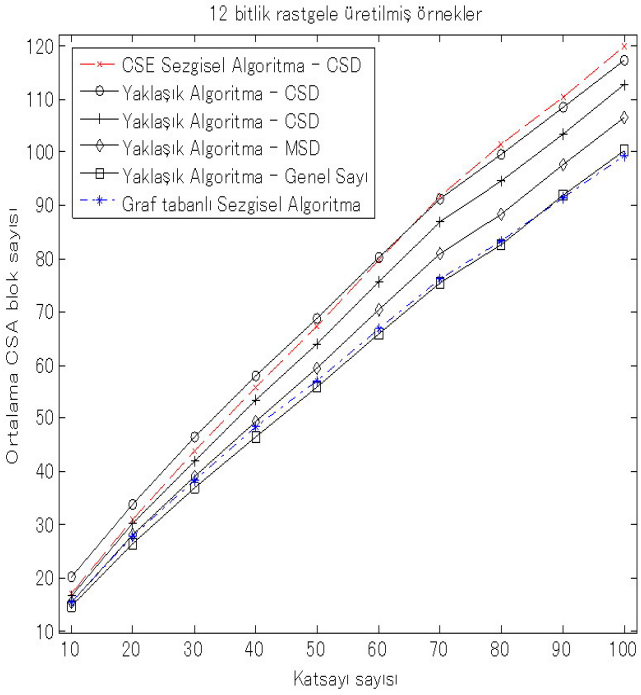
Kesin CSE algoritmasında olduğu gibi yaklaşık algoritmalarda da filtre katsayıları ve ara terimlerin bütün gerçeklemeleri bulunduktan sonra Boolean devresi inşa edilir ve CSA blok sayısının minimize edilmesi problemi bir 0-1 ILP problemine dönüştürülür. Minimum sonuç, bir genel 0-1 ILP çözücü ile bulunur.

Deneysel sonuçlar

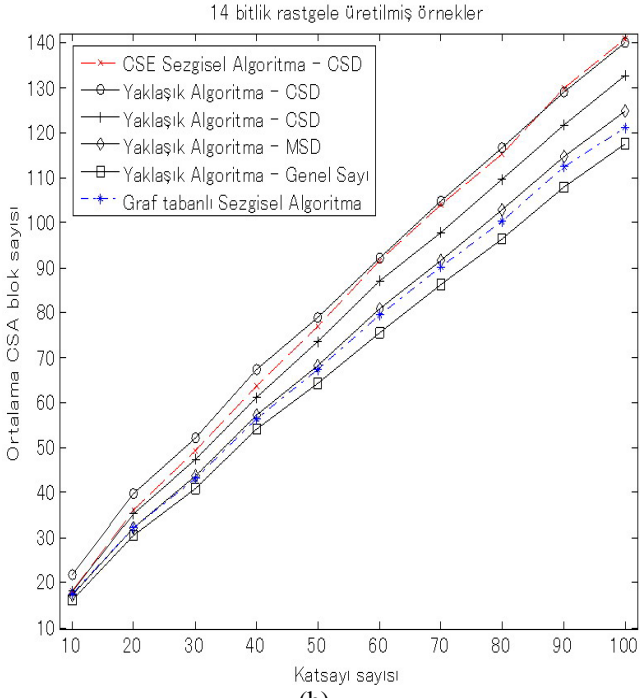
Bu bölümde, kesin ve yaklaşık algoritmaların rastgele üretilmiş örnekler ve FIR filtreleri üzerindeki sonuçları verilmekte ve daha önceden önerilen sezgisel yöntemlerin sonuçları ile karşılaştırılmaktadır. CSE sezgisel yöntemi, Hosangadi ve diğerleri (2006), ayrıca gerçeklenmiş ve graf tabanlı sezgisel yöntem, Gustafsson ve diğerleri (2004), Oscar Gustafsson tarafından sağlanmıştır.

İlk deney kümesi olarak 12 ve 14 bitlik katsayıların içerildiği rastgele üretilmiş örnekler kullanılmıştır. Katsayı sayısı 10 ile 100 arasında değişirken her bir küme için 30 adet örnek, toplamda 600 örnek, üretilmiştir. Yaklaşık algoritmaların ve daha önceden önerilmiş sezgisel yöntemlerin 12 ve 14 bitlik örnekler üzerindeki sonuçları sırasıyla Şekil 9(a) ve (b)'de verilmektedir.

Şekil 9'dan gözlenebildiği gibi katsayıların MSD gösterimde ifade edilmeleri, ikili ve CSD gösterimi ile elde edilen sonuçlardan daha iyi sonuçlar vermektedir. Bunun nedeni, bir katsayının MSD sayı sisteminde birden fazla gösterime sahip olmasındandır. Ayrıca, yaklaşık CSE algoritması sezgisel CSE yönteminden, Hosangadi ve diğerleri (2006), daha iyi sonuçlar vermektedir. CSD gösterimi altında, 12 ve 14 bitlik 100 adet katsayı içeren örneklerde sezgisel CSE yöntem ile yaklaşık CSE algoritmasının sonuçları arasındaki ortalama CSA blok sayısı farkı sırasıyla 7.2 ve 8.2'dir. Graf tabanlı sezgisel



(a)



(b)

Şekil 9. Algoritmaların rastgele üretilmiş örnekler üzerinde karşılaştırılması (a) 12 bitlik örnekler (b) 14 bitlik örnekler

yöntem, Gustafsson ve diğerleri (2004), bir sayı gösterimine bağlı kalmadığından dolayı, CSE algoritmalardan daha iyi sonuçlar elde etmektedir. Yine de, az sayıda katsayı içeren örneklerde MSD gösterimi altında yaklaşık CSE algoritmanın graf tabanlı sezgisel yöntem ile rekabet edecek düzeyde sonuçlar verdiği gözlemlenebilir. Öte yandan, genel sayı gösterimi altında yaklaşık algoritmanın 12 bitlik 90 ve 100 adet katsayı içeren örnekler hariç, graf tabanlı sezgisel yöntemden daha iyi sonuçlar bulmaktadır. 14 bitlik 90 adet katsayı içeren örnekler üzerinde graf tabanlı sezgisel yöntem ile genel sayı gösterimi altında yaklaşık algoritmanın sonuçları arasındaki ortalama CSA blok sayısı farkı 4.5'tir.

İkinci deney kümesi olarak filtre katsayıları MATLAB'de yer alan *remez* algoritması ile belirlenmiş FIR filtreleri kullanılmıştır. Filtrelerin özellikleri Tablo 1'de verilmektedir. Burada, *pass* ve *stop*, normalize geçirme ve söndürme frekanslarını, *#tap*, filtre katsayı sayısını ve *width* ise katsayıların bit uzunluğunu göstermektedir. Bunun yanında, Tablo 1'de kesin ve yaklaşık algoritmalar ile elde edilen 0-1 ILP problem boyutları da verilmektedir. Burada, *vars*, *cons* ve *optvars* sırasıyla 0-1 ILP problemin değişken sayısını, kısıt sayısını ve optimizasyon değişkeni sayısını göstermektedir.

Tablo 1'den görülebildiği gibi yaklaşık CSE algoritması ile kesin CSE algoritmaya göre daha küçük boyutlu 0-1 ILP problemler elde edilmektedir. Ayrıca, katsayılar genel sayı gösteriminde ifade edildiğinde oldukça büyük boyutlu 0-1 ILP problemler elde edilir. Bunun sebebi, genel sayı gösterimi altında bir katsayıyı, herhangi bir sayı gösterimine göre gerçekleyen olası işlem sayısının daha fazla olmasıdır. Ancak, günümüzün 0-1 ILP çözücüleri bu büyüklükteki problemleri rahatlıkla ele alabilmektedirler.

Tablo 2, algoritmaların sonuçlarını vermektedir. Bu tabloda, *CSA*, CSA blok sayısını, *step* ise MCM işlemini geçekleştiren seri olarak bağlanmış maksimum CSA blok sayısını göstermektedir. Ayrıca, *CPU*, *glpPB* adlı 0-1 ILP çözücünün 3.16GHz'lik Intel Xeon işlemciye ve 8GB'lık hafızaya sahip bir bilgisayarda minimum sonucu bulmak için gerekli olan CPU süresini saniye cinsinden göstermektedir.

Tablo 2'den görülebildiği gibi kesin CSE algoritması, sezgisel CSE yöntemi, Hosangadi ve diğerleri (2006), ile benzer veya ondan daha iyi

Tablo 1. Filtre özellikleri ve 0-1 ILP problem boyutları

Filtre	Filtre Özellikleri				Kesin CSE Algoritması			Yaklaşık Algoritma					
					MSD			MSD			Genel Sayı		
	pass	stop	#tap	width	vars	cons	optvars	vars	cons	optvars	vars	cons	optvars
1	0.15	0.25	40	12	506	809	271	126	165	96	5583	11138	2777
2	0.20	0.25	80	12	823	1345	426	84	74	74	3102	5882	1630
3	0.24	0.25	120	12	515	795	275	65	47	63	6638	12948	3398
4	0.15	0.25	60	14	1744	3108	871	141	205	105	9145	18357	4538
5	0.15	0.20	60	14	851	1362	442	199	275	145	24623	49916	12116
6	0.10	0.15	60	14	1581	2678	791	264	401	186	17479	35337	8615
7	0.10	0.15	100	16	11742	22726	5658	1768	4689	876	426650	876949	207350
8	0.15	0.25	120	16	8854	17091	4272	1338	3244	696	138176	283099	67358
9	0.10	0.15	160	16	27261	57079	13164	3117	8777	1397	539405	1106400	262733

Tablo 2. Algoritmaların FIR filtreler üzerindeki sonuçları

Filtre	CSE Sezgisel	Kesin CSE Algoritması						Yaklaşık Algoritma								Graf tabanlı Sezgisel
	CSD	CSD		MSD				CSD		MSD			Genel Sayı			
	CSA	CSA	step	CSA	step	CPU		CSA	step	CSA	step	CPU	CSA	step	CPU	CSA
1	16	16	3	16	4	0.1		16	3	16	3	0.1	15	4	2.7	16
2	30	28	3	27	4	0.2		28	3	27	3	0.1	25	4	0.3	25
3	31	31	3	31	4	0.1		31	3	31	3	0.1	31	4	0.6	31
4	25	25	3	21	5	0.3		25	4	21	4	0.1	21	5	3.9	22
5	36	35	3	34	4	0.3		35	3	34	3	0.1	34	3	13.5	35
6	36	34	3	32	4	0.5		34	3	32	3	0.1	32	4	6.8	34
7	60	60	5	55	6	52.1		60	5	55	5	0.5	53	6	4652.1	54
8	62	61	4	57	5	6	47.6	63	5	57	5	0.5	53	6	937.2	55
9	88	85	6	82	6	754.9		86	6	83	6	6.0	78	8	27165.9	81
Toplam	384	375	33	355	42	43	856.1	378	35	356	35	7.3	342	42	32783.0	353

sonuçlar elde etmektedir. CSD gösterimi altında sezgisel CSE yöntem ile kesin CSE algoritmasının sonuçları arasındaki fark ortalama 1 CSA bloğudur. Ayrıca, kesin CSE algoritması, katsayılar MSD gösteriminde tanımlandığında graf tabanlı sezgisel yöntemden, Gustafsson ve diğerleri (2004), 4., 5. ve 6. filtre örneklerinde daha iyi sonuçlar bulmaktadır. Bunun yanında, MSD gösterimi altında yaklaşık algoritma kesin CSE algoritmanın sonuçlarına benzer çözümleri daha kısa zamanda bulmaktadır. Genel sayı gösterimi altında yaklaşık algoritma ise graf tabanlı sezgisel yöntem ile benzer veya ondan daha iyi sonuçlar bulmaktadır. Graf tabanlı sezgisel yöntem ile genel sayı gösterimi altında yaklaşık algoritmanın sonuçları arasındaki fark ortalama 1.2 CSA bloğudur.

Sonuçlar

Bu makalede, yüksek hızlı bir sayısal FIR filtrenin çarpım bloğunun gerçeklenmesinde gerekli olan CSA blok sayısını minimize etmek için tasarlanmış bir kesin CSE algoritması sunulmuştur. Önerilen kesin CSE yöntemi gerçek boyutlu örnekler üzerinde uygulanabilse bile, doğal olarak, daha karmaşık problemleri ele alamayabilir. Bu yüzden, kesin CSE algoritmaya dayandırılan ve katsayıların sınırlı gerçeklemelerini ele alabilen bir yaklaşık CSE algoritması önerilmiştir. Bunun yanında, yaklaşık CSE algoritma, katsayıları genel sayı gösteriminde ele alacak şekilde genişletilmiştir. Böylece, CSE algoritmalardan çok daha fazla katsayı gerçeklemelerini göz önüne alabilen ve böylelikle, daha iyi sonuçlar bulabilen bir yaklaşık algoritma sunulmuştur.

Bu makalede önerilen yöntemler, rastgele üretilmiş örnekler ve FIR filtreler üzerinde test edilmiştir. Deneysel sonuçlardan, önerilen algoritmaların gerçek boyutlu örnekler üzerinde uygulanabildiği gözlemlenmiştir. Ayrıca, bu algoritmalar daha önceden önerilmiş sezgisel yöntemler ile karşılaştırılmıştır. Deneysel sonuçlardan kesin ve yaklaşık CSE algoritmaların sezgisel CSE yöntemden oldukça iyi sonuçlar elde ettiği ve genel sayı gösterimi altında yaklaşık algoritmanın graf tabanlı sezgisel yöntem ile rekabet edecek düzeyde ve daha iyi sonuçlar bulduğu gözlemlenmiştir.

Kaynaklar

Aksoy, L., Costa, E., Flores, P. ve Monteiro, J., (2007a). Optimization of area in digital FIR filters using gate-level metrics, *Proceedings*, Design Automation Conference, 420-423.

Aksoy, L., Costa, E., Flores, P. ve Monteiro, J., (2007b). Minimum number of operations under a general number representation for digital filter synthesis, *Proceedings*, European Conference on Circuit Theory and Design, 252-255.

Aksoy, L., Costa, E., Flores, P. ve Monteiro, J., (2008). Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **27**, 6, 1013-1026.

Barth, P., (1995). A Davis-Putnam based enumeration algorithm for linear pseudo-Boolean optimization, Technical Report, Max-Planck-Institut Für Informatik.

Cappello, P. ve Steiglitz, K., (1984). Some complexity issues in digital signal processing, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **32**, 5, 1037-1041.

Dempster, A. ve Macleod, M., (1995). Use of minimum-adder multiplier blocks in FIR digital filters, *IEEE Transactions on Circuits and Systems II*, **42**, 9, 569-577.

Flores, P., Monteiro, J. ve Costa, E., (2005). An exact algorithm for the maximal sharing of partial terms in multiple constant multiplications, *Proceedings*, International Conference on Computer-Aided Design, 13-16.

Gustafsson, O., Ohlsson, H. ve Wanhammar, L., (2001). Minimum-adder integer multipliers using carry-save adders, *Proceedings,* International Symposium on Circuits and Systems, 709-712.

Gustafsson, O. ve Wanhammar, L., (2002). ILP modelling of the common subexpression sharing problem, *Proceedings,* International Conference on Electronics, Circuits and Systems, 1171-1174.

Gustafsson, O., Dempster, A. ve Wanhammar, L., (2004). Multiplier blocks using carry-save adders, *Proceedings*, International Symposium on Circuits and Systems, 473-476.

Hartley, R., (1996). Subexpression sharing in filters using canonic signed digit multipliers, *IEEE Transactions on Circuits and Systems II*, **43**, 10, 677-688.

Hawley, R., Wong, B., Lin, T.-J., Laskowski, J. ve Samueli, H., (1996). Design techniques for silicon compiler implementations of high-speed FIR digital filters, *IEEE Journal of Solid-State Circuits*, **31**, 5, 656-667.

Hosangadi, A., Fallah, F. ve Kastner, R., (2006). Optimizing high speed arithmetic circuits using three-term extraction, *Proceedings*, Design, Automation and Test in Europe Conference, 1294-1299.

Johansson, K., Gustafsson, O. ve Wanhammar, L., (2005). A detailed complexity model for multiple constant multiplication and an algorithm to minimize the complexity, *Proceedings*, European Conference on Circuit Theory and Design, 465-468.

Kim, T., Jao, W. ve Tjiang, S., (1998). Circuit optimization using carry-save-adder cells, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **17**, 10, 974-984.

Nguyen, H. ve Chatterjee, A., (2000). Number-splitting with shift-and-add decomposition for power and hardware optimization in linear DSP synthesis, *IEEE Transactions on VLSI*, **8**, 4, 419-424.

Pasko, R., Schaumont, P., Derudder, V., Vernalde, S. ve Durackova, D., (1999). A new algorithm for elimination of common subexpressions, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **18**, 58-68.

Verna, A. ve Ienne, P., (2004). Improved use of the carry-save representation for the synthesis of complex arithmetic circuits, *Proceedings,* International Conference on Computer-Aided Design, 791-798.

Voronenko, Y. ve Püschel, M., (2007). Multiplierless multiple constant multiplication, *ACM Transactions on Algorithms*, **3**, 2.