POLYPUBLIE
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	An iterated tabu search algorithm for the design of fir filters
Auteur: Author:	Katayoon Moazzami
Date:	2011
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Moazzami, K. (2011). An iterated tabu search algorithm for the design of fir filters [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/755/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/755/
Directeurs de recherche: Advisors:	Michel Gendreau, & François Gagnon
Programme: Program:	Mathématiques appliquées

UNIVERSITÉ DE MONTRÉAL

AN ITERATED TABU SEARCH ALGORITHM FOR THE DESIGN OF FIR FILTERS

KATAYOON MOAZZAMI
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(MATHÉMATIQUES APPLIQUÉES)
DÉCEMBRE 2011

To my best friend, my greatest supporter and my eternal love
my husband, Marc.

ACKNOWLEDGEMENTS

RÉSUMÉ

Les systèmes modernes de télécommunication sans fils occupent une place majeure dans la société actuelle. Dans les dernières années, la complexité des outils qui en découlent n'a cessé d'augmenter car, en plus de prendre en charge les tâches basiques de communication vocale, ceux-ci doivent également supporter une quantité croissante de modules et d'applications parallèles (connexion internet, capture vidéo, guidage par satellite, etc.). En conséquence, l'évolution rapide subie par ces outils qui, dans la majorité des cas, sont alimentés par batteries, a singulièrement accru l'importance du rôle joué par la consommation énergétique, et a ainsi fait de l'efficacité énergétique et de l'informatique éco-responsable des caractéristiques essentielles dans les développements récents de la micro-éléctronique. Afin d'offrir une solution à ces problèmes énergétiques, une partie des recherches s'est focalisée sur la conception de filtres numériques efficaces. Les filtres numériques sont la pierre angulaire de tous les systèmes de traitement de signal numérique. Chaque filtre est implanté par un circuit intégré, qui, lui-même, est composé d'une liste d'éléments de base incluant des additionneurs, des multiplicateurs, des inverseurs, etc. La piste principale suivie par les chercheurs dans le but de réduire la quantité d'énergie consommée par les filtres numériques propose de remplacer les multiplicateurs dans les circuits par des éléments moins énergivores, tels que des additionneurs, des décaleurs et des inverseurs. L'objectif des méthodes introduites dans ce sens consiste généralement à remplacer les multiplicateurs tout en utilisant le moins d'additionneurs possible. En effet, en l'absence de multiplicateurs dans les circuits, les additionneurs deviennent l'élément le plus demandant en ressource énergétique. Dans les faits, la quantité d'additionneurs contenue dans un circuit sans multiplicateurs, aussi connue comme son coût en additionneurs, est communément utilisée afin d'estimer sa consommation énergétique.

Nos travaux se concentrent sur la conception de filtres numériques sans multiplicateurs énergétiquement efficaces. Ils se décomposent en deux contributions majeures : un nouveau modèle de représentation efficace des circuits intégrés, et un algorithme innovateur destiné à la conception de filtres numériques efficaces. Dans un premier temps, notre modélisation des circuits sous la forme de graphes pondérés a l'avantage d'offrir une représentation concise des circuits intégrés, tout en annulant la symétrie présente dans les modèles de représentation actuels. Dans un second temps, notre métaheuristique, qui combine à la

fois une recherche tabou et une recherche tabou itérée, offre un contrôle direct du niveau d'énergie consommée par le circuit qu'elle construit, en fixant la quantité d'additionneurs qu'il contient avant le démarrage du processus de conception. En outre, contrairement aux méthodes existantes, notre approche ne se réfère à aucune architecture spécifique afin de concevoir un circuit. Ce degré de liberté permet à notre méthode d'atteindre une optimisation plus globale de la structure du circuit en comparaison des autres méthodes et, ainsi, de posséder un contrôle plus précis de sa consommation énergétique.

L'algorithme proposé est testé sur un jeu de données contenant plus de 700 filtres de complexité variée. Les résultats obtenus démontrent les performances élevées de notre approche car, en se basant sur le coût en additionneurs, dans plus de 99% des cas, notre méthode conçoit des filtres numériques avec un niveau de consommation énergétique total équivalent au niveau induit uniquement par l'architecture à laquelle les méthodes actuelles se réfèrent. En parallèle, notre méthode fournit également un meilleur contrôle de la longueur de mot interne dans les circuits, qui représente un autre aspect crucial de leur efficacité énergétique. La comparaison avec l'algorithme *Heuristic cumulative benefit* (Hcub) qui, à ce jour, est la méthode la plus performante montre que les filtres construits par notre algorithme utilisent 55% moins d'additionneurs que Hcub, tout en réduisant la taille de ces additionneurs de 33%. Ces améliorations sont obtenues au simple coût d'une augmentation de 17% du nombre de délais dans les circuits. Cependant, la consommation énergétique d'un délai étant de l'ordre de 20% de celle d'un additionneur, si l'on considère le nombre et la taille des additionneurs ainsi que la quantité de délais inclus dans nos circuits afin d'estimer leur consommation énergétique, on peut s'attendre à une économie globale de l'ordre de 65% en comparaison de la meilleure méthode actuelle.

ABSTRACT

In today's modern society, we rely on wireless telecommunication devices that use applications and modules to perform many different tasks and are growing in their complexity day by day. Consequently, the fast evolution of these devices, which, most of the time, are battery-powered, drastically increased the importance of their energy consumption and made energy efficiency and green computing essential features of recent developments in microelectronics.

To deal with the related issues, many researchers have focused their attention to designing energy-efficient digital filters, which are essential building blocks of all digital signal processing systems. Any digital filter is implemented by an integrated circuit composed by a list of basic elements, including adders, multipliers, shifts, etc. One of the paths that researchers have followed in order to decrease the amount of energy used by the integrated circuits was to replace the multipliers in the circuit structure with less energy-consuming elements such as adders, shifts and inverters. The goal of these methods is usually to perform the replacement of multipliers while using the least amount of adders, as, for multiplierless circuits, adders become the most energy-consuming elements. In fact, the quantity of adders contained in a multiplierless circuit, also known as its adder cost, is commonly used as an estimate of its power consumption.

In our research we focus on energy-efficient multiplierless filters. Our work has two main contributions: a new model to efficiently represent integrated circuits, and an innovative algorithm to design efficient digital filters. On one hand, the main advantage of our new graph-based model is that it is able to represent any integrated circuit in a concise form, while avoiding symmetry in the representation. On the other hand, our metaheuristic, that combines both a tabu search and an iterated tabu search, offers a direct control of the level of energy consumed by the circuits it constructs, by fixing the number of adders that they contain. Besides, unlike other existing methods used for designing multiplierless filters, our approach does not refer to any specific architecture in the corresponding circuit structure. This degree of freedom allows our method to have a more globalized view on the optimization of circuit structure compared to the other methods, and thus, a better control on its power consumption.

The proposed algorithm is tested on a benchmark containing more than 700 filters of dif-

ferent orders of complexity. The obtained results demonstrate the high accuracy of the proposed approach as, based on the adder cost estimation, in more than 99% of the cases our method designs integrated circuits with a level of energy consumption equivalent to those implied only by the most accurate circuit architectures from which existing algorithms build their circuits, and absolutely no deviation from the desired filtering specifications. In parallel, our method also provides a better control of the internal wordlength in the circuits, which is another crucial point to improve the energy-efficiency. The comparison to the current state-of-the-art algorithm *Heuristic cumulative benefit (Hcub)* when designing all the benchmark filters shows that filters constructed with our algorithm are using 55% less adders than Hcub, while decreasing their size by 33%. This improvement can be reached at the cost of an increase of 17% in the number of delays in the circuits. However, by considering the number and the size of adders used in the circuit as well as the quantity of delays it contains as an estimate of the power consumed by the circuit, assuming that the energy consumption of a delay is in the order of 20% of the consumption of an adder, we can approximately expect an overall energy saving of 65% in our circuits compared to the best current method.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1   Problem definition

The demand for increasing level of performance in third and fourth generation telecommunication systems calls for devices with more competitive properties, one of which is energy-efficiency. This has made low-power high-performance digital signal processing (DSP) a critical path to follow.

Digital filters, which are some of the basic instrument to process a signal, are considered as essential bluiding blocks of DSP systems. Any filter is implemented by an integrated circuit, which in turn, is composed of different basic elements such as shifts, registers, adders and multipliers. In this work, we introduce a new graph-based model for representing the integrated circuits efficiently and concisely.

Using operations research methods for designing efficient digital filters is not new. Many different optimization algorithms have been applied to filter design, as it will be shown in the next chapter. These algorithms are usually based on graph theory, linear programming or metaheuristic approaches such as genetic algorithms, simulated annealing, tabu search and so forth.

One way to improve the energy consumption of a circuit is to reduce the occurence of the elements that consume the most amount of energy in the circuits. Since multipliers are the most energy consuming components in the circuits, and as adders are the second ones, a great amount of research has been dedicated to the design of efficient multiplier-less filters that require the least number of adders in their circuit structure.

Therefore, many research efforts have been dedicated to the design of low-complexity multiplier-less filters, most of which are focused on an important family of filters, namely, Finite Impulse Response (FIR) filters. FIR filters do not allow any recursive loops in their circuit structure, in other words, the output of these filters depends on the present and previous values of the input signal and no output values. Our research is also adapted to the design of energy-efficient multiplier-less FIR filters, as we do not allow any loops in the

model we use for representing the integrated circuits that the filters are based on.

Most of the works that have focused on minimizing the complexity of filters in order to optimize their energy consumption are based on circuit structures derived from existing architectures. However, there are a few papers that do not follow this hypothesis and consider their circuit structures to be free (Châtelain et Gagnon (2007); Joliveau *et al.* (2011)). Our work is based on the same idea, which means that we do not refer to any defined architecture. Thus, our algorithm has a higher degree of liberty concerning the placements of the circuit elements. This results in the design of non-traditional-looking circuit structures that require much less circuit elements.

In order to analyze the performance of filters, their specifications in the frequency domain are tested: the magnitude response of each filter must fit into a given mask, which usually defines the locations of the passbands and the attenuation bands, as well as the level of error tolerated. In order to provide accurate filters, our algorithm tries to reduce the average distance between the magnitude response of the filter it is designing and the specification mask. Notions related to the filters, their properties and their fitness are discussed in length in the next chapters.

As mentioned before, one of the operations research methods that has been used in filter design and has yielded interesting results is tabu search (Glover (1986)). There is also another version of tabu search that is considered to be an extension to it, the iterated tabu search proposed by Misevicius *et al.* (2006)). Our algorithm is a combination of tabu search and iterated tabu search. The design of more than 700 digital filters with various degrees of complexity using this algorithm and the comparison to the state-of-the-art method *Heuristic cumilative benefit (Hcub)* demonstrate the very high accuracy of our approach. The details of algorithm stages and experimental results are extensively discussed in the following chapters.

1.2   Research objectives

In our research, we address the issues related to the design of low-complexity energy-efficient filters. To do so, we focused our attention towards the design of non-recursive multiplier-less filters that consume much less energy compared to those produced by state-of-the-art algorithms. We made this possible by developing an algorithm that benefits from a well-defined model for representing the integrated circuits, as well as by taking advantage of powerful metaheuristics like tabu search and iterated tabu search. It is also important to

note that the proposed methodology offers a very higher degree of liberty than traditional approaches, as it does not rely to any defined circuit architectures. Although it considerably increases the size of the solution space that the method must explore, it also opens to the potential identification of a new family of efficient integrated circuits with free structure and low power consumption, that would have been ignored until today, because of the limitations of classical methods.

1.3   Thesis structure

In this chapter we have given an overview of the problem in hand and the approach we use to solve it. In the second chapter, we discuss the previous work in digital filter design and the related operations research methods used in this field. We will also introduce basic notions regarding digital filtering, tabu search and iterated tabu search metaheuristics. In the third chapter, we present our graph-based model for representation of integrated circuits and then move on to describing our algorithm and its basic ingredients. Chapter 4 is dedicated to the presentation and a discussion of computational results achieved. In chapter 5, some conclusions are drawn and directions for future work are proposed.

Chapter 2

LITERATURE REVIEW

As this research is a combination of the two very different domains of operations research and electronics, we will explain the concepts and related work in these two domains to familiarize the reader with the context of our research. This chapter consists of two main sections. In section 2.1 digital filtering and other signal processing notions related to our research are discussed. Section 2.2 focuses on operations research and mainly on the tabu search metaheuristic that is applied in our work.

2.1 Digital filtering

Digital signal processing (DSP) algorithms are used to transform and analyze signals that have been collected from various sources. Digital signal processing has various applications such as communication, control, meteorology and biomedical. Unlike analog systems, the performance of a digital system is not dependent on the tolerance of electrical components. Digital filters can be considered as the essential building blocks of digital signal processing, as any time invariant DSP algorithm can be described as a digital filter. Physically, filtering is performed by integrated circuits. These integrated circuits, or digital filters, are used to execute various basic functions, such as pulse shading, modulation, demodulation, as well as more complex applications like image compression, audio equalization, interference reduction or even separation of the POTS (Plain Old Telephone Service) and DSL (Digital Subscriber Line) signals (Smith (2007)).
The circuits contain simple elements that perform basic operations such as addition, multiplications by a constant, and delay. Figure 2.1 shows an example of an integrated circuit that implements a digital filter .
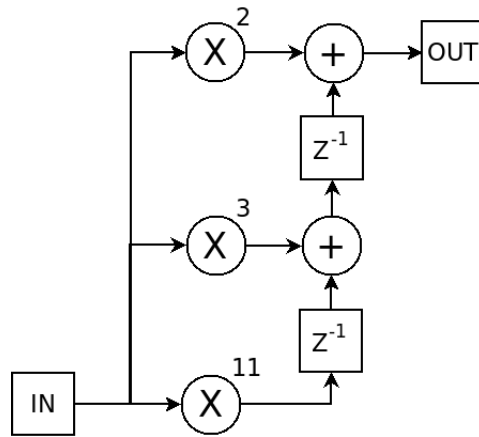
Figure 2.1 Illustration of a typical Finite Impulse Response (FIR) digital filter. The circuit, built according to the transpose form architecture, performs simple operations such as multiplications by a given constant c ($\otimes^c$), additions ($\oplus$), and delays (z^{-1}).
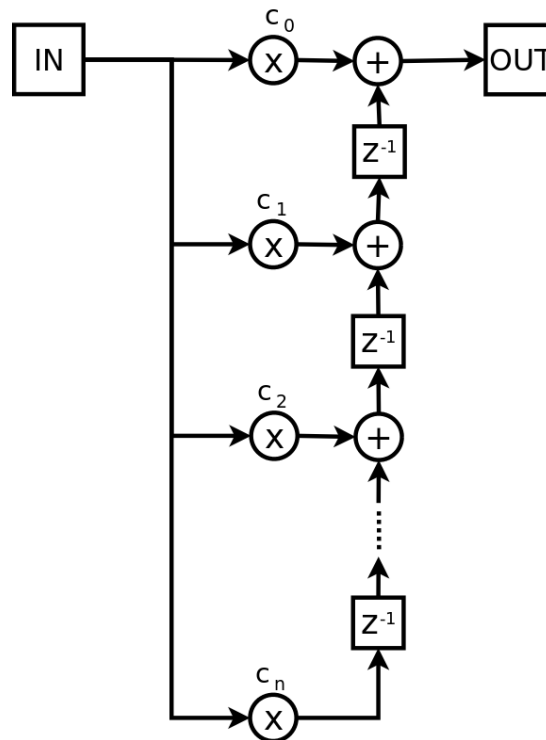
Figure 2.2 Generic form of a FIR filter under its canonical representation. According to this structure, a filter of order n is implemented by a circuit containing n adders , n delays and $n+1$ multipliers.

2.1.1 Canonical representation and order of a filter

The circuits implementing a filter are designed according to a defined architecture. Among these architectures, the canonical representation of a filter, also known as the transpose form architecture, is the circuit structure that uses the least amount of multipliers, adders and delays to implement a digital filter. This architecture is also used to determine the order of a filter. By definition, the order of a filter corresponds to the number of adders needed to implement a filter according to its canonical representation, which is also equal to the number of multipliers minus one. According to this definition, the order of a filter is usually considered as an indicator of its complexity. The concept of order can also be applied to circuits, assuming that the order of a circuit is equal to the order of the filter it implements. Figure 2.2 illustrates the generic form of a filter of order n implemented under its canonical representation.

2.1.2 Digital filter design

Traditionally, two principal families of filters exist: Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters. The main difference between these families is that, IIR circuits contain recursive loops (i.e., cycles) whereas FIR circuits do not allow them. This basically means that the response of a FIR filter is dependent only on present and a finite number of previous values of the input signal while, IIR filters depend not only on input data, but also on some output values.

The use of operations research (OR) techniques in the design of digital filters is not new. Throughout the years, many different algorithms have been proposed for optimizing different aspects of filter design. Genetic algorithms are one of the most popular algorithms used (Suckley (1991); Cemes et Ait-Boudaoud (1993); Wade *et al.* (1994); Xu et Daley (1995); Lee *et al.* (1998); Oner (1998); Lian et Cen (2003); Ling et Yong (2004); Ahmad et Antoniou (2006); Dey *et al.* (2010); Shing-Tai (2010); Boudjelaba *et al.* (2011)), but other techniques such as local search (Kodek et Steiglitz (1981); Samueli (1989); Xu et Nowrouzian (1999)), ant colony optimization (Karaboga *et al.* (2004)), simulated annealing (Ling (2007); Boudjelaba *et al.* (2011)) and tabu search (Karaboga *et al.* (1997); Fanni *et al.* (1998); Traferro *et al.* (1999); Traferro et Uncini (2000); Kalinli et Karaboga (2005); Watcharasitthiwat *et al.* (2006); Ling (2007); Boudjelaba *et al.* (2011)) have been employed as well. The earlier algorithms, which mainly applied local search techniques or genetic algorithms, lacked a certain level of elegance as they were not completely adapted

to the problems they were applied to. Later on, better algorithms were developed. These algorithms were mainly based on tabu search (Karaboga *et al*. (1997); Fanni *et al*. (1998); Traferro *et al*. (1999); Traferro et Uncini (2000); Kalinli et Karaboga (2005); Watcharasitthiwat *et al*. (2006); Ling (2007); Boudjelaba *et al*. (2011)) or more complex genetic algorithms (Lian et Cen (2003); Ling et Yong (2004); Ahmad et Antoniou (2006); Ling (2007)) and they yielded better results than their predecessors.

Currently, much more sophisticated algorithms are used. These algorithms, which are mainly hybrid algorithms, try to design better filters by incorporating favorable aspects of other traditional algorithms such as GAs, tabu search and ant colony algorithms (Karaboga *et al*. (2004); Kalinli et Karaboga (2005); Ling (2007)). In order to improve the computation time, parallel versions of metaheuristic algorithms have also been considered (Xu et Daley (1995); Karaboga *et al*. (1997); Kalinli et Karaboga (2005); Watcharasitthiwat *et al*. (2006)).

Algorithms proposed for optimization of filter design have concentrated on optimization of different aspects, such as the value of objective function (usually based on frequency response of the filter) (Cemes et Ait-Boudaoud (1993); Wade *et al*. (1994); Xu et Daley (1995); Watcharasitthiwat *et al*. (2006); Ling (2007)), computational complexity (Suckley (1991); Fanni *et al*. (1998); Dey *et al*. (2010); Boudjelaba *et al*. (2011)), coefficient accuracy (Samueli (1989); Oner (1998); Xu et Nowrouzian (1999); Kalinli et Karaboga (2005); Shing-Tai (2010)), low filter complexity (Samueli (1989); Lian et Cen (2003); Kalinli et Karaboga (2005)), high speed (Lian et Cen (2003); Kalinli et Karaboga (2005)) or chip area (Wade *et al*. (1994)). These algorithms have been applied to both IIR filters (Xu et Nowrouzian (1999); Karaboga *et al*. (2004)) and FIR filters, but mainly to FIR filters (Cemes et Ait-Boudaoud (1993); Wade *et al*. (1994); Traferro *et al*. (1999); Xu et Nowrouzian (1999); Kalinli et Karaboga (2005); Ahmad et Antoniou (2006); Watcharasitthiwat *et al*. (2006); Ling (2007)), and most of all to multiplier-less FIR filters (i.e., FIR filters that do not contain multipliers) (Cemes et Ait-Boudaoud (1993); Wade *et al*. (1994); Fanni *et al*. (1998); Lee *et al*. (1998); Lian et Cen (2003); Ahmad et Antoniou (2006); Ling (2007)). A majority of the algorithms proposed for designing multiplier-less FIR filters are based on signed powers-of-two (SPoT) approaches that express the coefficients of the filter as sums of signed powers-of-two (Cemes et Ait-Boudaoud (1993); Traferro *et al*. (1999); Xu et Nowrouzian (1999); Lian et Cen (2003); Ahmad et Antoniou (2006); Shing-Tai (2010)).

This research looks into designing FIR filters that have a low filter complexity and, as a

result, consume less energy. Multipliers are the type of circuit elements that use the most amount of energy in the circuit, so we will focus on the design of multiplier-less FIR filters. But our research idea goes further than that. Unlike other existing algorithms that optimize the filter while keeping the underlying architectures, filters designed by our algorithm will not be limited by any circuit structures. Thus, the process that is proposed is more global as it considers the entire circuit, whereas existing methods only focus on the replacement of multipliers. This idea of free structure circuits was first proposed in Châtelain et Gagnon (2007) for IIR filters, where the authors designed a genetic algorithm that achieved lower filter complexity and higher data rate. The concept of energy efficient circuits with free structure was then improved in Joliveau *et al.* (2011), where the authors proposed an evolutionary variable neighbourhood search (E-VNS) algorithm for designing IIR multiplier-less filters. This work will focus on developing an iterated tabu algorithm that provides low-complexity multiplier-less FIR filters with a fixed level of energy consumption.

2.1.3   Multiplier-less filters

Although FIR and IIR filters in their traditional sense are simple and fast, they use multipliers. Unfortunately, multipliers consume significant amount of power in comparison to other elements in the circuit. Thus, the main trend followed for obtaining energy-efficient filters, is the replacement of multipliers by other circuit elements. This process consists of replacing the constant multiplication (multipliers) by multiplier blocks, which are sub-circuits essentially constituted of additions (adders), multiplication by -1 (inverters) and multiplication by power of 2 (left and right shifts)(table 2.1).

Table 2.1 List of the components used in multiplier-less integrated circuits for digital signal processing.

Name	Symbol	Signal processing
Adder	$\oplus$	Adds the two input signals
Delay	z^{-1}	Delays the signal of 1 time clock
Left shift	$\overset{n}{\leftarrow}$	Multiplies the input signal by 2^n
Right shift	$\overset{n}{\rightarrow}$	Multiplies the input signal by 2^{-n}
Inverter	$\rhd\!\circ$	Multiplies the input signal by -1

In the absence of multipliers in the circuit, adders become the new central element concerning circuit complexity and power consumption. Consequently, the goal of multiplierless circuit design methods is to use the least number of adders in the circuit. In fact, the number of adders that the circuit contains, also known as *adder cost*, is commonly used as a good estimate of its energy consumption.

Generation of multiplier blocks from a set of constants is known as the Multiple Constant multiplication or MCM problem and finding its optimal solution (i.e., the one with the fewest additions) is NP-Hard. Among the existing algorithms, three principal approaches can be distinguished:

- *digit-based recoding* (Avizienis (1961)), which generates the decomposition of the coefficients directly from their digit representation ;
- *common subexpression elimination* (Pasko *et al.* (1999); Coleman (2001); Macleod et Dempster (2005); Thong et Nicolici (2009)), which are extensions of digit-based recoding methods, whose basic idea is to find common subpatterns in the representation of the coefficients after they are converted to a convenient number system ;
- *graph-based algorithms* (Bernstein (1986); Bull et Horrocks (1991); Dempster et Macleod (1995); Gustafsson *et al.* (2006); Voronenko et Puschel (2007)), which iteratively construct the graph representing the multiplier blocks.

These techniques, which are usually based on evolutionary methods such as genetic algorithm, allow designing accurate low complexity digital filters with a small adder cost.

While the digit-based recoding methods, such as the *Canonical Signed Digit* (CSD) representation (Avizienis (1961)), try to express independently each multiplier by a multiplier block (Figure 2.3a), common subexpression elimination and graph-based algorithms, including the *Bull-Horrocks algorithm* (BHA) (Bull et Horrocks (1991)), the *n-dimensional Reduced Adder Graph* (RAG-n) algorithm (Dempster et Macleod (1995)) , and the *heuristic cumulative benefit* (Hcub) (Voronenko et Puschel (2007)), reach lower adder costs by considering parallel multiple constant multiplication (Figure 2.3b).

Although these methods, and particularly the state-of-the-art algorithm Hcub, manage to construct multiplier-less circuits with few adders and low power consumption, their approach is limited, as they focus only on the replacement of multipliers. They thus rely on existing architectures, which imply the presence of extra adders, to define the remaining part of the circuit. In order to provide a more global approach, we propose a new multiplierless circuit design algorithm that optimizes the entire circuit structure without referring to
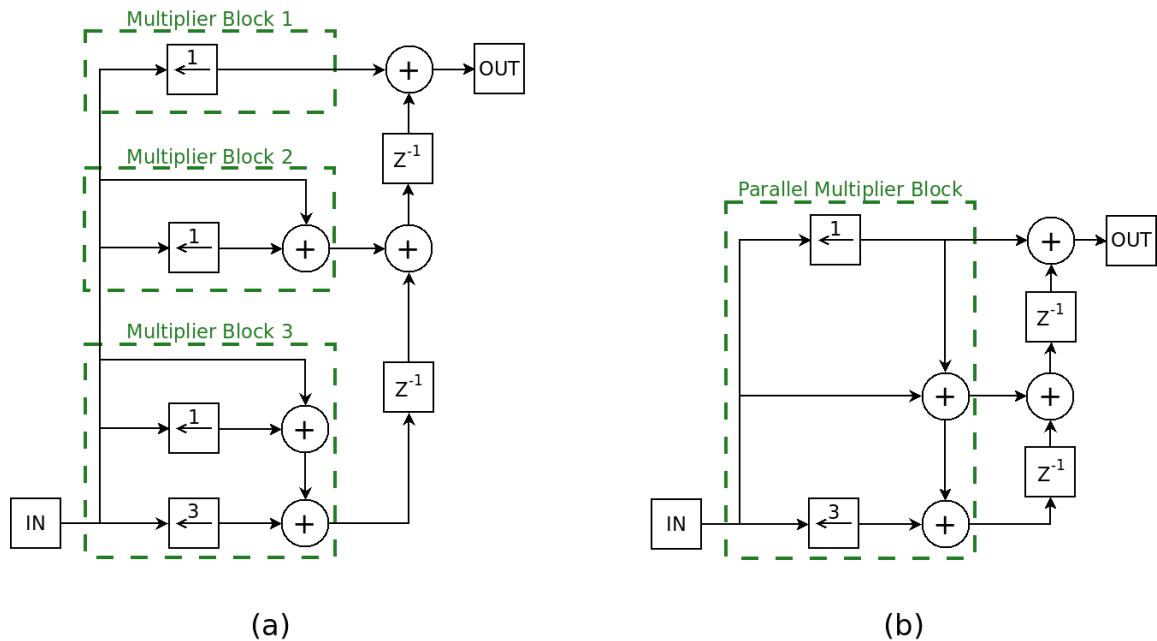
Figure 2.3 Design of the FIR filter of Figure 2.1 using MCM methods in order to (a) replace each multiplier by a multiplier block or (b) replace all three multipliers by a unique parallel multiplier block.

any defined architecture. In the following section we will discuss the notions and previous work related to the operations research part of our work.

2.2 Tabu search

Tabu search is an adaptive procedure used for solving a wide range of combinatorial optimization problems. Tabu search, as it is known today, was proposed by Glover (1986) who viewed it as a metaheuristic (i.e., a master strategy that guides and uses heuristics to solve combinatorial problems) rather than a heuristic (i.e., intelligent procedures that seek optimal or near-optimal solutions but cannot guarantee reaching such solutions). It is worth mentioning that at the same time, a very similar heuristic called *steepest ascent , mildest descent* was independently proposed by Hansen (1986).

Origins of tabu search go back to procedures applied in solving non-linear integer programming problems (the method was used as a tool for implementing the oscillating assignment strategy) (Glover (1977)). Tabu search has proven to be very efficient in finding optimal

or near-optimal solutions for many different combinatorial optimization problems, including traveling salesman problems (Knox et Glover (1989); Malek *et al.* (1989b,a)), graph coloring (Hertz et de Werra (1987)), scheduling problems (Glover et McMillan (1986); Eck (1989)), maximum stable set problems (Friden *et al.* (1989)), stochastic programming (Lokketangen et Woodruff (1996)), mixed integer programming (Lokketangen et Glover (1996); Cranic *et al.* (2000)), real-time decision problems (Gendreau *et al.* (1999)) and so forth.

2.2.1   Neighborhood search

Assuming that a combinatorial optimization problem P can be represented by:

$$(P)\quad \text{Minimize } c(x) : x \in X \subseteq \mathbb{R}_n,$$

where the objective function $c(X)$ can be linear or nonlinear. Most heuristic methods that tries to solve a combinatorial optimization problem can thus be viewed as a sequence of *moves* from one solution of X to the other. Therefore, a move can be described as a mapping defined on X :

$$s : X \to X.$$

The set of moves that can be applied to a solution x can be represented by the set $S(x)$, and all the solutions that may be reached from a solution x form the neighbourhood $N(x)$ of x.

Neighborhood searches generally consists of exploring the solution space by iteratively moving from a solution x to a solution in its neighborhood $N(x)$. Hill climbing heuristics are the simplest of neighborhood searh heuristics. A summarized version of the hill climbing heuristic can be seen below (Glover (1989b)).

Hill Climbing Heuristic for (P)

1. Select an initial $x \in X$.

2. Select some $s \in S(x)$ such that

$$c(s(x)) < c(x).$$

   If not such s exists, x is a local optimum and the method stops. Otherwise,

3. Let $x := s(x)$ and return to step 2.

The hill climbing method basically moves from one solution to another until it reaches a local optimum. The main problem of this heuristic is that when the local optimum is found, the algorithm stops and there is a good chance that this local optimum is not the global optimum.

2.2.2   Tabu search algorithm

Tabu search can be seen as an extension of hill climbing heuristics. The main difference between the two methods is that, unlike hill climbing heuristics, tabu search does not stop in local optima but it keeps moving to the best solution inside the neighborhood of the current solution, while expecting that this non-improving move can lead to the identification of a better local optimum further in the process. In order to avoid cycling between a solution and a local optimum that has been previously visited by the algorithm, tabu search uses information structures called *tabu lists*. This list stores a given number of moves that would lead to a previous local optimum.

The size and contents of tabu lists depend on the problem but they are usually not very long (typically short-term memory is used for the lists) and mostly they contain the inverse of the last few modifications made to the current solution. Based on the characteristics of the problem in hand, some *tabu constraints* are defined and tabu lists are constructed according to these constraints. Tabu constraints are usually stated to make reversal or in some cases repetition of some moves impossible. This, in most cases, is done by forbidding certain attributes of these moves.

Thus, a tabu list can be defined as T where :

$$T(x) = \{s \in S : s(x) \text{ violates the tabu conditions}\}.$$

A simple tabu search (Gendreau et Potvin (2010)) is presented in the following template which applies the most commonly used version of tabu search called the *best improvement* tabu search (i.e., at each iteration the best available move is chosen):

SIMPLE TABU SEARCH

Notation
- S: the current solution,
- S^*: the best-known solution,
- f^*: the value of S^*,
- $N(S)$: the neighborhood of S,
- $\tilde{N}(S)$: the admissible subset of $N(S)$ (i.e., non-tabu or allowed by aspiration),
- T: the tabu list.

Initialization
Choose (construct) an initial solution S_0.
Set $S \leftarrow S_0$, $f^* \leftarrow f(S_0)$, $S^* \leftarrow S_0$, $T \leftarrow \emptyset$.

Search
While termination criterion not satisfied do
    select S'' in $argmin_{S' \in \tilde{N}(S)}[f(S')]$;
    set $S \leftarrow S''$;
    if $f(S) < f^*$, then set $f^* \leftarrow f(S)$, $S^* \leftarrow S$;
    record tabu for the current move in T (delete oldest entry if necessary).

In the following subsections, we will discuss some features that can be added to tabu search in order to improve its adaptability to complex implementations.

2.2.3   Aspiration function

Although tabu lists are powerful tools for avoiding cycling, they may also prohibit attractive moves. Thus, there is another important aspect of tabu search to consider and that is a function called *aspiration criteria* which basically defines the criteria where we

Examine another move
(enlarging list if appropriate)

Evaluate Each Candidate Move

Does the move yield a higher evaluation
than any other move found admissible so far
(from the current candidate list)?

NO

YES
(Potential acceptance)

Check Tabu Status

Is the candidate tabu?

Tabu   **Not Tabu**

Check Aspiration Level

Does move satisfy aspiraiton
criteria?

YES

Move Is Admissible

Designate as best admissible
candidate

NO   (Admissibility recorded)

Candidate List Check

Is there a "good probability" of better
moves left, or should candidate
list be extended?

NO

Make the Chosen Best Admissible Move

Figure 2.4 Evaluation of moves.

can ignore the tabu constraints. In other words, tabu lists and the aspiration function play a dual role; you can choose a move by ignoring tabu conditions, while aspiration criteria are satisfied. This can be very well seen in Figure 2.4. The most commonly used aspiration function, allows a move to be chosen if its resulting objective function value is better than the current best solution.

For a more complete description of aspiration functions one can refer to Glover (1989b); Glover et Laguna (2002). There are also more complex examples of aspiration levels that can be found in de Werra et Hertz (1989); Hertz et de Werra (1991).

2.2.4   Intensification and diversification

The basic idea of intensification is to explore regions of the search space that seem more promising more thoroughly. But intensification by itself is not sufficient for reaching the best results, therefore a complementary concept called diversification has been proposed which diversifies the search by guiding it to lesser or never explored regions of the search space.

While applying tabu search to search space one can use short-term, intermediate and long-term memory as a tool to achieve better results. Short-term memory is usually used for tabu lists and intermediate and long-term memories serve in two more complicated elements that can be incorporated into tabu search, namely, search intensification and diversification.

Intermediate memory is used as a means for learning about favourable features of good solutions during a particular period of time. Through this learning period, certain good characteristics or *attributes* of attractive solutions are distinguished. Then, this knowledge is applied for seeking new solutions that contain these attributes. For a better understanding of intensification, one can refer to Glover (1990).

In order to prohibit the tabu search from spending most of its efforts in a limited region of the search space (something the search has a tendency towards) and probably missing some good solutions in other regions, long-term memory is used. Long-term memory works as a tool for diversifying the search and as a result guides the search to regions that have not been explored or have been explored less. There are three main ways for achieving diversification: strategic oscillation, restart diversification and continuous diversification. For more information on these techniques one can refer to Glover (1989b, 1990); Soriano et Gendreau (1996).

Short-term memory is considered as the core of tabu search, as it is an aggressive way to

Begin with a Starting Current Solution

Obtain the solution by initialization or from an intermediate or long-term memory component

Create a Candidate List of Moves

(if applied, each move would generate a new solution from the current solution)

Choose the Best Admissible Candidate

(Admissibility is based on the tabu restrictions and aspiration criteria). Designate the solution obtained as the new current solution. Record it as the new Best Solution if it improves on the previous best.

Stopping Criterion

Stop, if a specified number of iterations has elapsed in total or since the last Best Solution was found.

Stop   **Continue**

Terminate Globally or Transfer

A transfer initiates an intensification or diversification phase embodied in an intermediate or long-term memory component.

Update Admissibility Conditions

Update Tabu restrictions and aspiration criteria.

Figure 2.5 Tabu search

find the best possible moves. The choice of best move is usually related to the change in objective value it causes. But if the real objective function value is complex to determine, approximation is used. For better understanding of evaluation process one can refer to Glover (1989a); Glover et Laguna (2002). A general view of tabu search and its components discussed earlier is provided in Figure 2.5.

According to Misevicius *et al*. (2006), the repetitive transition between an intensification phase handled by a classic tabu search, and a diversification phase that follows a *reconstruct and improve* paradigm can be considered as an *iterated tabu search* (ITS). In this framework, the goal of the intensification phase is to search for better (locally optimal) solutions in a specific area of the solution space, whereas the diversification phase escapes local optimum by moving towards new regions in the solution space. In this sense, the diversification can follow a *ruin and recreate* principle, that consists of deleting a part of a solution (ruin) and then reconstructing a new solution from the part that has not been destructed (recreate). The principle advantage of this strategy is its effectiveness in dealing with solution spaces that contain deep valleys.

Except for the first iteration, intensification is always applied to the solution that has just been reconstructed (i.e., the output of diversification). However, there are two main alternatives for selecting the candidate for diversification: exploitation and exploration. Exploitation is achieved by choosing the best local optimum identified during the previous intensification phase as a candidate for destruction and reconstruction. In the case of exploration a variety of policies may be used. In fact, each locally optimized solution encountered during the intensification can be considered as a potential candidate for diversification. Figure 2.6 illustrates the general framework of an iterated tabu search.

Diversification

Perform a ruin & recreate

Intensification

Perform a classic tabu search

Start

Evaluation

Save the best solution so far

End

Candidate acceptance

Exploitation: Select the best local optimum identified
during previous intensification
OR
Exploration: Select one local optimum identified during
previous intensification
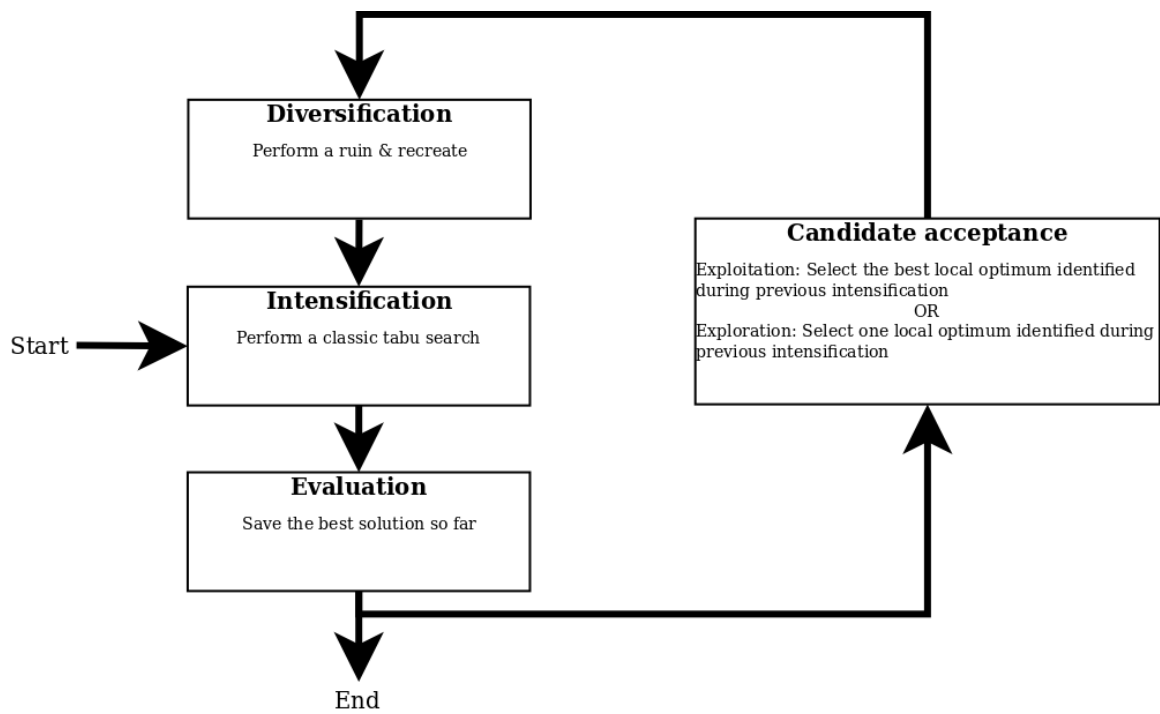
Figure 2.6 Generalized framework of an iterated tabu search.

Chapter 3

MODEL & ALGORITHM

This chapter presents our approach to designing low complexity multiplier-less digital filters. Section 3.1 defines the graph-based model that we introduce to accurately represent integrated circuits. Section 3.2 describes the fitness function that we use to estimate the quality of a filter. And section 3.3 is dedicated to the different components of our optimization algorithm.

3.1   A new graph-based model to accurately represent integrated circuits

As previously illustrated in Figure 2.1, integrated circuits are traditionally characterized by a diagram that indicates how the elements that constitute a circuit are assembled. This schematic representation is based on a library of symbols that allocates a specific icon to each type of circuit element (Table 2.1). These symbols are linked by lines or arrows that indicate the connections between the circuit elements.

Although this traditional representation has the advantage of being easy to read and understand, the same circuit realization can be represented in different ways. Figure 3.1 shows a graphical example of how the same circuit realization can be represented in 12 different ways using the traditional schematic representation. This symmetry in the representation, which is mostly due to the commutativity of the operators inside the circuits (addition, multiplication), can cause serious problems for an optimization algorithm to accurately explore the solution space during the design process. That is the reason why we developed a new graph-based model that prohibits symmetry in the representation of integrated circuits.

Figure 3.1 An example of the symmetry in the traditional schematic representation.

3.1.1 Model definition

In order to face the issues of the traditional schematic representation related to symmetry, Joliveau *et al.* (2011) introduced a binary tree model to efficiently represent any multipl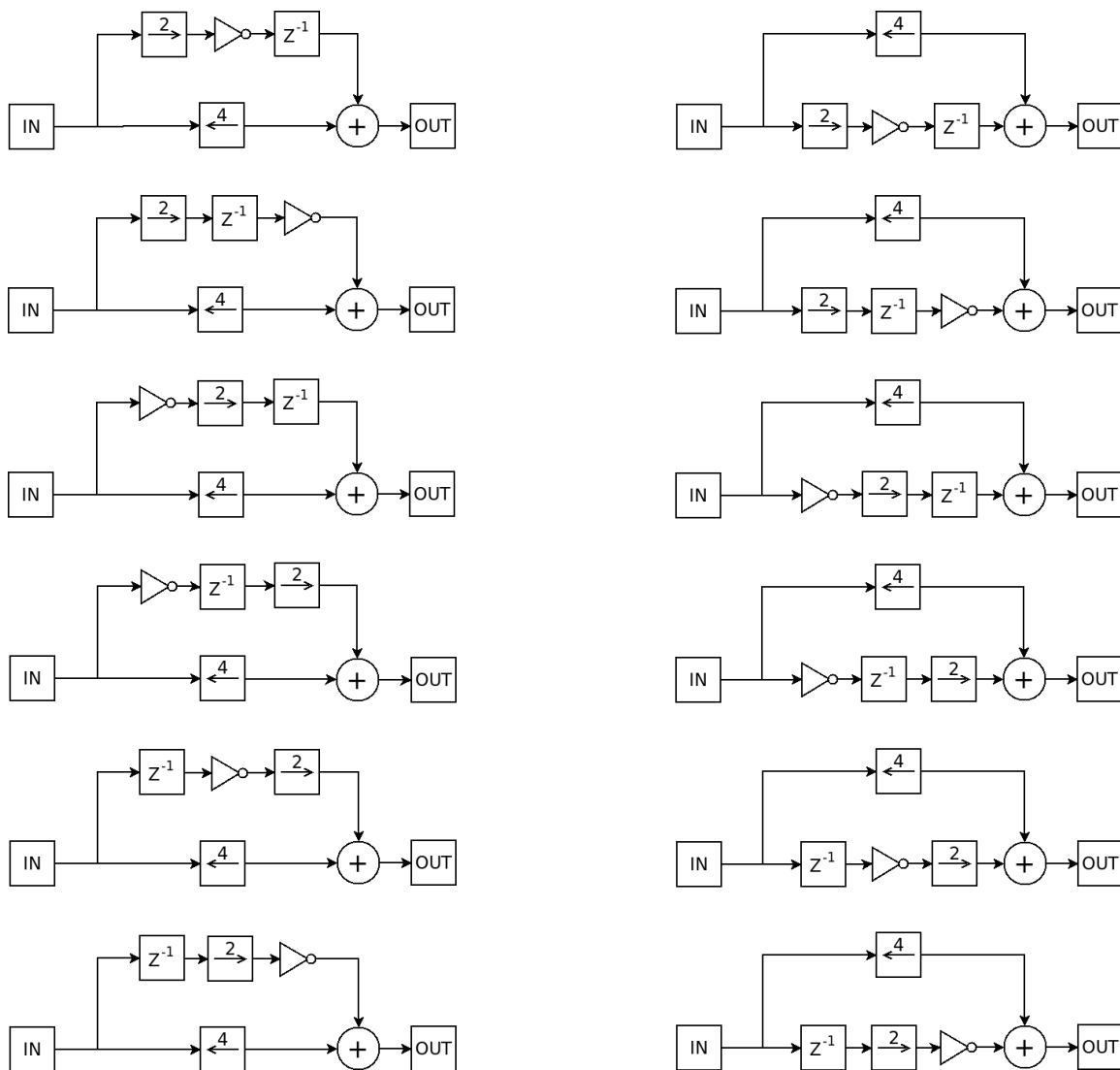ier-less integrated circuit realization (recursive and non-recursive) under a unique way. As our research only focuses on FIR digital filters, we derived the idea of a unique representation to non-recursive circuits from Joliveau *et al.* (2011) and defined a graph-based model adapted to our context. This model is centered around the location of adders in the circuits and their connections with other circuit elements (shift, inverter and delay). Following our model, a circuit containing N adders is thus symbolized by a directed acyclic graph $G = (X, E)$ with a source X_0 that stands for the circuit input, a sink X_{N+1} that stands for the circuit output, and N vertices X_i, $i \in [1, N]$ that represent the adders. Each edge of the graph symbolizes a *connection path* between two adders of the circuit, or between an adder and the circuit input/output. A connection path is defined as a path between two adders in the circuit that does not pass through another adder. Since adders have two inputs, there can be a maximum of two connection paths between two given adders i and j, which would lead to a graph with up to two distinct edges linking the vertices X_i and X_j.

Each edge of the graph $e_k \in E$ is weighted according to two functions: the process function $P(e_k)$ and the delay function $D(e_k)$. The process function considers all the shifts and the inverters on the connection path represented by the edge and indicates the corresponding constant by which the signal is multiplied. For example, if a connection path represented by the edge e_k contains one inverter (i.e., a multiplication by -1) and three left shifts (i.e., three consecutive multiplications by 2), the process value of e_k will be given by $P(e_k) = -2^3$. As for the delay function, it simply shows the number of delays on the connection path.

Figure 3.2 gives an example of how the same integrated circuit is illustrated according to the classical schematic model and the proposed graph-based model.

3.1.2 Ensuring the feasibility of the circuit structure

While presenting a novel approach to represent non recursive integrated circuits, one of the priorities is to provide a model that guarantees the feasibility of the circuit structure. This can be done by introducing a set of rules for the model which ensure that the quantity of inputs and outputs of each element is respected and that there are no recursive loops. In order to describe these rules, we must introduce the functions $d_\Gamma^-(X)$, $d_\Gamma^+(X)$ and $N_\Gamma^-(X)$ that respectively indicate the in-degree (i.e., the number of predecessors), the out-degree

Traditional schematic representation
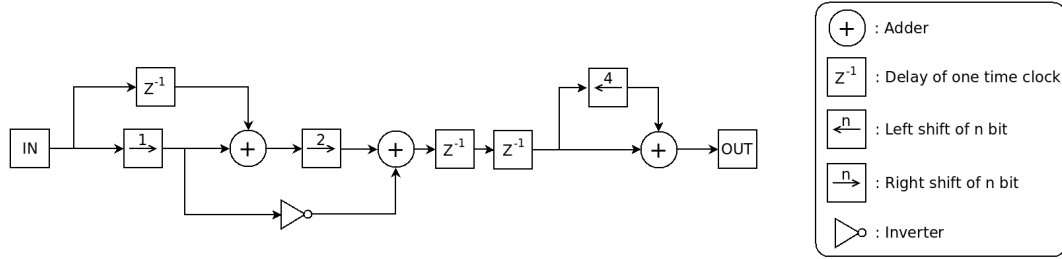
Proposed Graph-based representation
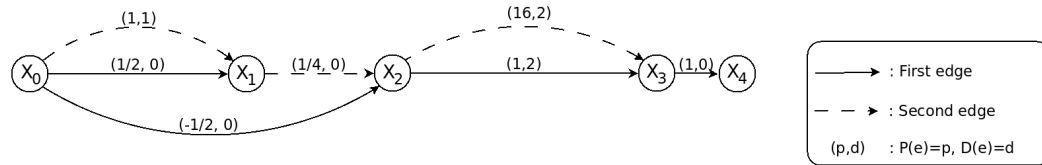
Figure 3.2 Illustration of the same integrated circuit using the traditional schematic representation and the proposed graph-based representation.

(i.e., the number of successors) and the in-neighborhood (i.e., the list of predecessors) of a given vertex X.

Rule 1: Respecting the number of input and output of circuit elements

Rule 1a: The circuit input has no predecessors and at least one successor

$$d_\Gamma^-(X_0) = 0 \text{ and } d_\Gamma^+(X_0) >= 1.$$

Rule 1b: The circuit adders have exactly two predecessors and at least one successor

$$\forall i \in [1,N], \ d_\Gamma^-(X_i) = 2 \text{ and } d_\Gamma^+(X_i) >= 1.$$

Rule 1c: The circuit output has exactly one predecessor and no successor

$$\forall i \in [1,N], \ d_\Gamma^-(X_{N+1}) = 1 \text{ and } d_\Gamma^+(X_{N+1}) = 0.$$

Rule 2: non-recursivity of the circuit

$$\forall i \in [1, N+1], \ i \in N_{\Gamma}^{-}(X_j) \Rightarrow i < j.$$

3.1.3   Prohibiting the symmetry in the representation

As mentioned before, one of the goals of our model is to avoid symmetry in the representation. In order to achieve this goal, we add another set of rules to the model. By defining the weight function for the graph edges, our model already takes care of the symmetry related to shift, delays and inverters. Thus, only the symmetry stemming from the commutativity of the addition remains. This can be addressed by defining an order on the two edges e_{k1} and e_{k2} respectively connecting vertices X_{i_1} and X_{i_2} to vertex X_j. A function $ArgTail(e_k)$ that indicates the index of the tail (i.e., the initial vertex) of an edge $e_k \in E$, is used for this matter.

Rule 3: Determining which of the two edges entering in a vertex X_j, $j \in [1,N]$ representing an adder is the first edge ($Edge_1(j)$) and which is the second edge ($Edge_2(j)$)

> **Rule 3a**: The index of the tail of the first edge is less than or equal to the index of the tail of the second edge:
>
> $$ArgTail(Edge_1(X_j)) \leq ArgTail(Edge_2(X_j)).$$

> **Rule 3b**: If both edges e_{k1} and e_{k2} come from the same predecessor, the first edge contains less delays than the second edge:
>
> $$if \ (X_{i_1} = X_{i_2}),$$
> $$D(Edge_1(X_j)) \leq D(Edge_2(X_j)).$$
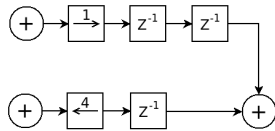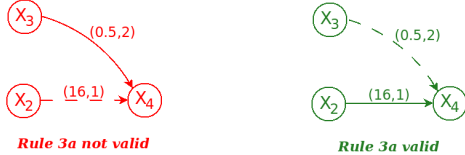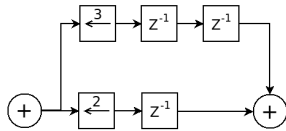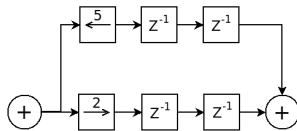
> **Rule 3c**: If both edges e_{k1} and e_{k2} come from the same predecessor and contain the same number of delays, the multiplication constant of the first edge is less than the multiplication constant of the second edge
>
> $$if \ ((X_{i_1} = X_{i_2}) \wedge (D(Edge_1(X_j)) = D(Edge_2(X_j)))),$$
> $$P(Edge_1(X_j)) \leq P(Edge_2(X_j)).$$

A graphical example of each of these rules can be viewed in Table 3.1.

Table 3.1 Illustration of the 3 rules to avoid symmetry in the model.

3.2 Evaluation of integrated circuits for digital filtering

3.2.1 Time domain and frequency domain

A digital signal can be viewed as a sequence of numbers in a binary representation. The principle of digital filtering is to send each component of the signal (i.e., each set of bits representing a number) one after the other to the integrated circuit input and then to extract the processed response from the circuit output. When performing such operations, we say that we are analyzing the filter on the *time domain*. The specifications of a digital filter in the time domain can be extracted from its *impulse response $x(t)$*, which is the response of the circuit to a Dirac pulse (i.e., a signal composed of a one and a sequence of zeros). However, the desired specifications of a filter to be designed are usually not expressed on the time domain but in the *frequency domain*. Fortunately, it is rather simple to express the impulse response in the frequency domain by using a Fast Fourier Transform (FFT) on $x(t)$ and by considering the complex modulus of the coefficients obtained. The expression of the impulse response in the frequency domain is called the *frequency response*. Although naturally continuous, the set of frequencies that the processed signal can contain is tradinionally discretized by using uniform subsampling techniques into a finite set Ω delimited by the lower frequency ω_{min} and the higher frequency ω_{max} that the signal can reach. From the frequency response $X(\omega)$, one can easily deduce the *magnitude response $M(\omega)$*, for every frequency $\omega \in \Omega$ which is expressed in decibel (dB), according to the formula:

$$\forall \omega \in \Omega, \, M(\omega) = 20 \log_{10}(X(\omega)).$$

In digital filters, the frequency response is usually between 0 and 1 (i.e., $X(\omega) \in]0,1]$). Thus by expressing the magnitude of the frequency response in a logarithmic scale, we can easily deduce that $M(\omega) \in]-\infty, 0]$. Figure 3.3 shows how to extract both the impulse response and the magnitude response of an integrated circuit by shifting from the time domain to the frequency domain.

3.2.2 Passband and attenuation band

Digital filters should respect a set of characteristics defined on their magnitude response. These characteristics or specifications correspond to frequency intervals, or frequency bands, for which the magnitude must be the closest possible to a given magni-

Time domain

| Dirac input signal | Integrated circuit | Impulse response |

$\Downarrow$

| Magnitude Resp. | Frequency Resp. | Fast Fourrier Transform (FFT) |

Frequency domain
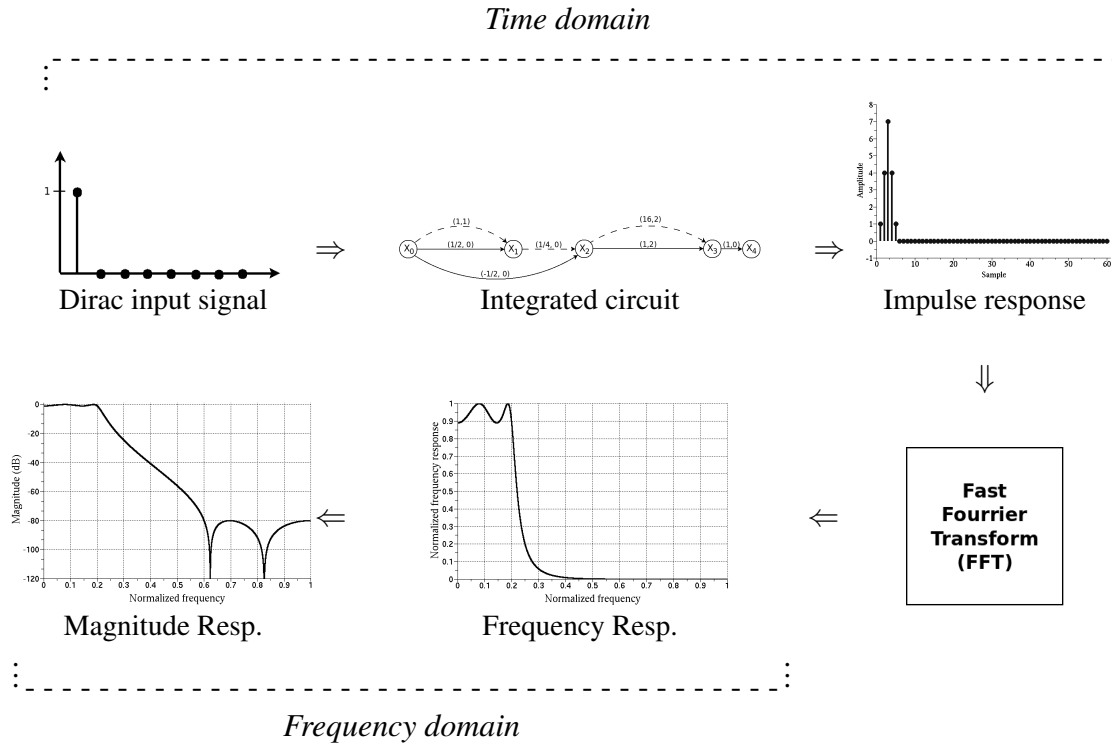
Figure 3.3 Extraction of the filter specification from its circuit implementation.

tude *m*.

In signal filtering, there are two types of bands: the passbands and the attenuation bands. In the passband, the magnitude of a signal is supposed to be unchanged by the filter and to have a $m = 0$ dB gain. Whereas for the attenuation band the magnitude of the signal has to be attenuated as much as possible. In practice a given degree of deviation is usually tolerated in both band types. For passbands, we refer to the ripple in the passband r_p, which is defined as the lowest magnitude for which the signal is still considered to be close enough to 0 dB, while for attenuation bands, we consider the attenuation level Att_a, which is the maximum magnitude for which the signal is still considered attenuated.

3.2.3   Specification mask and fitness function

Before running the algorithm, in order to describe the requirements of the filter to be designed, we build a specification mask in which the normalized magnitude response of the filter should fit. This mask is specified by an upper bound $M^+(\omega)$ and a lower bound

$M^-(\omega)$ defined for each normalized frequency $\omega \in [0,1]$ according to the limitations of the passbands and attenuations bands, as well as the authorized level of deviation in them:

$$M^-(\omega) = \begin{cases} r_p & \text{if } \omega \in \Omega_p \\ -\infty & \text{else} \end{cases} \quad ; \quad \text{and} \quad M^+(\omega) = \begin{cases} Att_a & \text{if } \omega \in \Omega_a \\ 0 & \text{else} \end{cases} .$$

where:

- Ω_p contains all the discretized frequencies in the passbands ;
- Ω_a contains all the discretized frequencies in the attenuation bands.

The basic idea of our fitness function is to evaluate the quality of a circuit according to the average squared distance between its normalized magnitude response $M(\omega)$ and the specification mask boundaries $(M^-(\omega), M^+(\omega))$. We thus obtain the following function Z that the algorithm will try to minimize:

$$Z = \sum_{\omega \in \Omega_p} \beta_p(\omega) \frac{M(\omega)^2}{|\Omega_p|} + \sum_{\omega \in \Omega_a} \beta_a(\omega) \frac{(M(\omega) - Att_a)^2}{|\Omega_a|}$$

where:

- $\beta(\omega)$ is a vector of penalty parameters for normalized frequency ω:

- $\beta_p(\omega) = \begin{cases} 1 & \text{if } M^-(\omega) \le M(\omega) \le M^+(\omega) \\ \beta & \text{else} \end{cases} ;$

- $\beta_a(\omega) = \begin{cases} 0 & \text{if } M^-(\omega) \le M(\omega) \le M^+(\omega) \\ \beta & \text{else} \end{cases} .$

- $|\Omega_p|$ and $|\Omega_a|$ respectively indicate the number of frequencies considered in the passband and in the attenuation band.

It is important to note that the objective function Z does not only refer to the average squared distance between the magnitude response of the circuit and the mask boundaries, but it also introduces a vector of penalty parameter $\beta(\omega)$. This choice has been made according to the application of digital filters in telecommunication where, although a small error in the passband is tolerated, magnitude close to 0 dB are really favored, whereas the respect of the attenuation level in the attenuation bands is usually enough for a filter to be considered accurate.

The function Z thus takes into account in two situations:

- the case where the magnitude of the filter does not respect the boundaries of the mask, which is penalized ($\beta(\omega) = \beta$) ;

– the case where, for a passband frequency $\omega \in \Omega_p$, the magnitude of the filter is inside the mask but less than 0 dB, which is considered by Z, but not penalized ($\beta_p(\omega) = 1$).

If a solution fits the mask for every normalized frequency $\omega \in [0, 1]$, it means that the corresponding circuit fully respects the desired specifications. In such case, we will consider the integrated circuit as *specification feasible (SF)*.

Figure 3.4 illustrates the evaluation of a low-pass filter which is defined as the succession of one passband $\Omega_p = \omega \in [0, \omega_p]$, and one attenuation band $\Omega_a = \omega \in [\omega_a, 1]$. The two more complex steps while computing the value of Z for a given circuit are the extraction of the impulse response and the FFT performed to compute $M(\omega)$ from $X(t)$. These steps rely on two parameters: the length of the Dirac signal used to compute the impulse response (L), which also corresponds to the length of the FFT computed, and the number of adders in the circuit (N). Using our model, the impulse response can be determined by a simple algorithm with a complexity order of $O(NL)$, whereas a standard FFT can be performed by an algorithm with a complexity order of $O(Llog(L))$. A precision of $L = 256$ is enough to extract the main features of the frequency response and in our experiments we do not use more than $N = 15$ adders to design a circuit, this makes determining
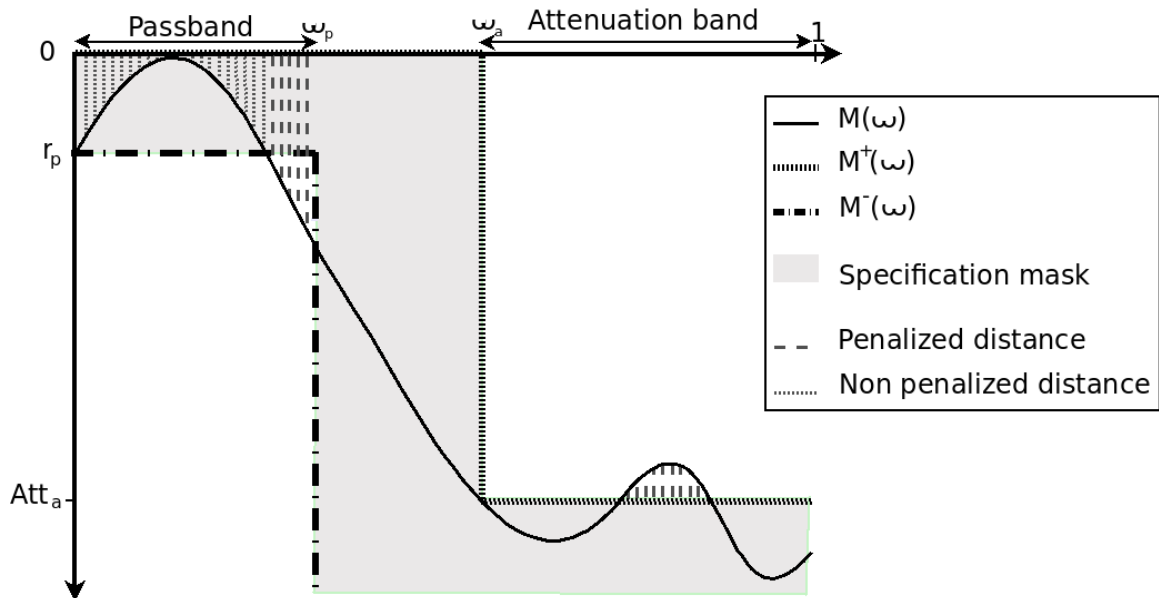
Figure 3.4 Evaluation of an integrated circuit: Example of a low-pass filter.

the value of Z from a circuit possible in a few milliseconds.

3.3   An iterated tabu search for integrated circuit design

As we mentioned previously, our circuit design algorithm is not based on any defined circuit architecture. It thus progresses in the solution space by inserting, removing and moving components inside the circuit while trying to satisfy at best a quality criterion defined on the circuit magnitude response as measured by the fitness function Z.
A major advantage of our algorithm is that it fixes the number of adders in the circuit and then attempts to achieve the best circuit possible without inserting other adders. Thus, the number of adders in a multiplier-less circuit being used as an estimate of the circuit complexity and its power consumption, our algorithm is aimed to design the digital filter that respect at best a pattern of magnitude response for a fixed degree of complexity and power consumption.

Our method can be decomposed in four steps (Figure 3.5):

1. A heuristic (Random Structure Generation - RSG) that randomly generates a starting population;

2. A tabu search that performs a first optimization of the circuits in the population;

3. A selection process that identifies the subset of circuits from the previous step which will be improved in the next step;

4. An iterated tabu search that performs a second optimization of the selected circuits while exploring the solution space more efficiently than the tabu search in step 2.

The algorithm has two stopping conditions:
   – When it constructs a circuit with a magnitude response that meets the specifications of the desired mask. This stopping condition can be reached while performing the tabu search (stopping condition 1a) or the iterated tabu search (stopping condition 1b).
   – If the iterated tabu search reaches the maximum number of iterations authorized for each circuit of the selection (stopping condition 2).
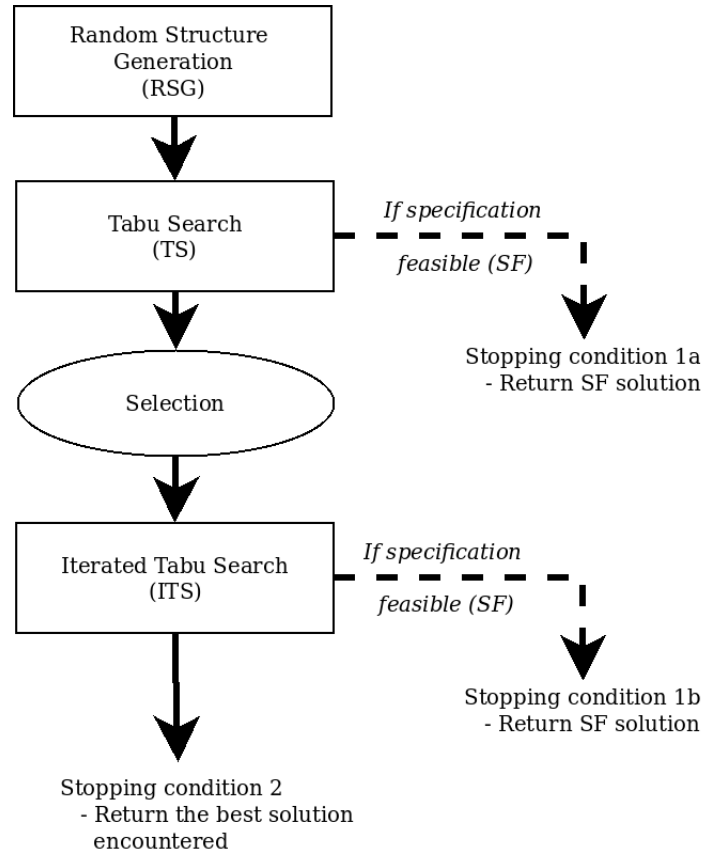
Figure 3.5 Schematics of the algorithm.

3.3.1   Random Structure Generation (RSG)

The Random Structure Generation heuristic (RSG) is a very simple algorithm that is aimed at providing a set of starting solutions for the tabu search that contain the same number of adders (i.e., the number of vertices in the graph is fixed *a priori*). It consists in building the starting structure of circuits (i.e., determining the edges in their graph-based representation) by randomly selecting the two predecessors of each vertex X_i, $i \in [1, N]$ representing an adder, while respecting the rules of the model, and by creating an edge between the vertex X_N (the last adder) and X_{N+1} (the circuit output). This process is repeated P_{TS} times in order to provide P_{TS} starting solutions, where P_{TS} is a parameter indicating the size of the population for the next step of the algorithm (tabu search).

At this stage the algorithm still does not consider the presence of other components (shift, inverter and delay) than adders in the circuits. Thus, the weights of every edge e in the

graph is set to $P(e) = 1$ and $D(e) = 0$.

3.3.2 Tabu Search (TS)

For the second step of our algorithm we use a tabu search. This tabu search is based on a meta-neighborhood N that contains three independent sub-neighborhoods N_1, N_2 and N_3:

Neighborhood N: Including/Removing shifts, delays and inverters or changing the configurations of the adders.

    Sub-neighborhood N_1: Change the process weight of each edge in the graph. The number of moves is controlled by the parameter *maxShift* that indicates the maximum number of consecutive shifts between two adders, whereas the presence of an inverter is symbolized by the sign of $P(e)$ (i.e., if the edge e contains an inverter, $P(e) < 0$)

$$\forall e \in E, \forall s \in [0, maxShift], \quad P(e) = \pm 2^{\pm s}.$$

    Sub-neighborhood N_2: Change the delay weight of each edge in the graph. The number of moves is controlled by the parameter *maxDelay* that indicates the maximum number of consecutive delays between two adders

$$\forall e \in E, \forall d \in [0, maxDelay], \quad D(e) = d.$$

    Sub-neighborhood N_3: Change the predecessors of each vertex while respecting the rules of the model. Both weights of the edge remain the same.

$$\forall X_j \in [X_1, X_N], \forall i_1 \in [0, ArgTail(Edge_2(X_j))],$$
$$ArgTail(Edge_1(X_j)) = i_1;$$

$$\forall X_j \in [X_1, X_N], \forall i_2 \in [ArgTail(Edge_2(X_j)), j-1],$$
$$ArgTail(Edge_2(X_j)) = i_2.$$

The algorithm explores the neighborhood N completely which means that at the end of an iteration the algorithm will move to the best solution encountered inside N_1, N_2 and N_3. However it is interesting to note that unlike traditional methods that use the previous move or solution in the tabu list, our algorithm inserts the value of the function Z.

The tabu search is independently applied to every individual in the population and it either stops when a defined maximum number of iterations are executed or until it finds a solution that respects the specification mask completely (the algorithm stops in this case as well). If all the population is considered and the tabu search does not find any solutions that respect the mask, the algorithm moves to the next step.

3.3.3 Selection

This step consists of selecting a portion of the population computed by the tabu search to be used as a starting solution for the iterated tabu search step of the algorithm. Several selection strategies like roulette-wheel selection, stochastic universal sampling or traditional tournament selection have already been proposed in the past. In our implementation, we simply use a truncation selection process where we select the fittest individuals of the input population. The number of solutions selected is P_{ITS}, which is the size of the population of the iterated tabu search.

3.3.4 Iterated Tabu Search (ITS)

As mentioned in the previous chapter, the principal idea of ITS is the concept of intensification and diversification (I&D) that can be decomposed in three main components: intensification, diversification and candidate acceptance (selection).

In our circuit design ITS, these components are implemented as follows:

- *Intensification* consists of performing a standard tabu search on a meta-neighborhood containing the two previously introduced sub-neighborhoods N_1 and N_2.
- *Candidate acceptance* selects the candidate that will be used in next step. In our implementation, we will consider two strategies for candidate acceptance: exploitation and exploration. The exploitation strategy simply consists of selecting the best solution encountered by the tabu search performed during the intensification process, whereas the exploration strategy selects the last local optimum reached by the tabu search.
- *Diversification* follows a ruin and recreate principle by first erasing randomly chosen edges from the solution, and then reconstructing all the possible solutions according to the model rules and determining the best one. Figure 3.6 illustrates this process on a simple circuit.

Solution s selected during candidate acceptance, Z(s) = 4

Altered solution (removal of the first edge between X2 and X3)

Reconstructed solution s1, Z(s1) = 6 *Reconstructed solution s2, Z(s2) = 3*

Selection of the solution with the best finess value
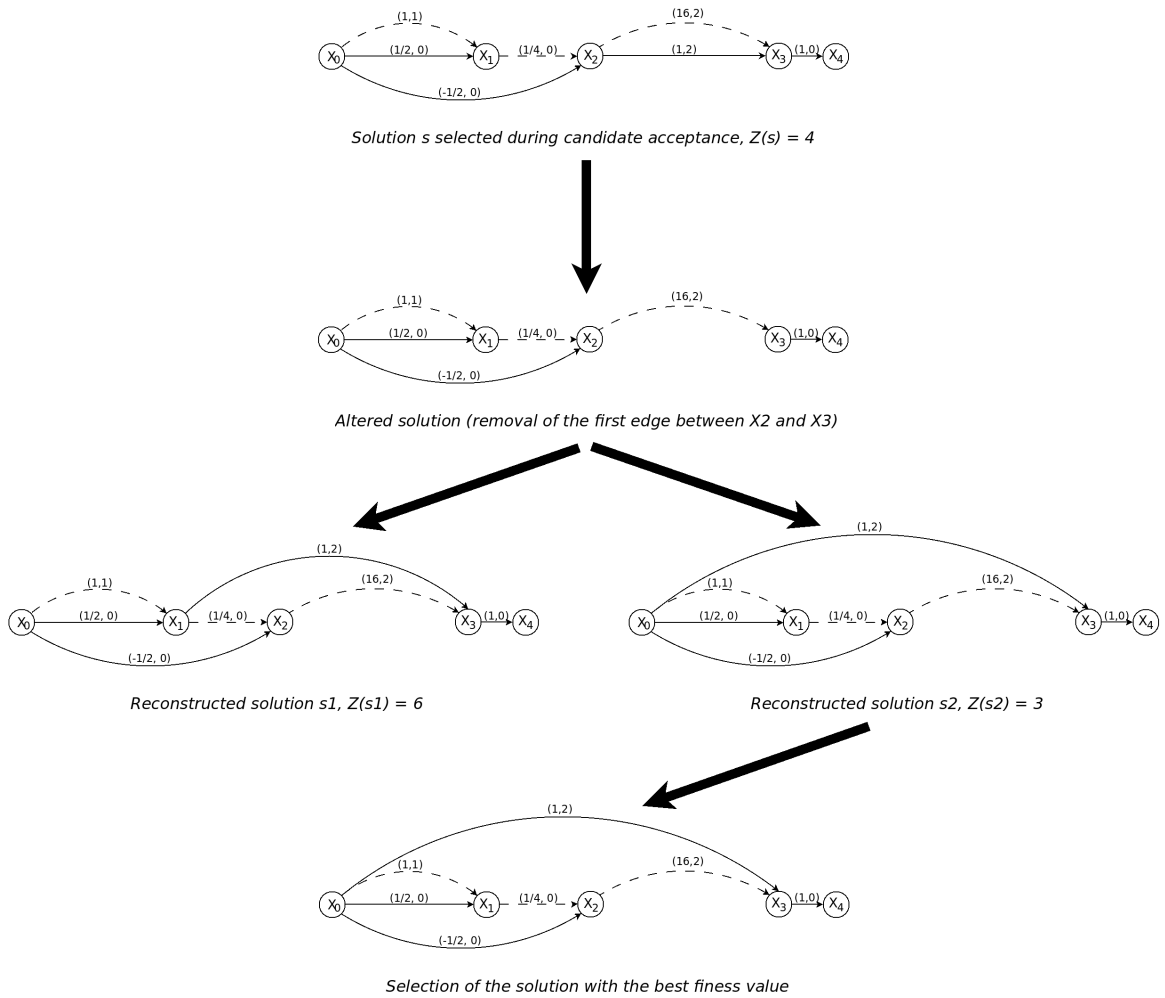
Figure 3.6 Illustration of the diversification process executed in our Iterated Tabu Search algorithm for circuit design.

The iterated tabu search is performed a maximum number of times on every member of the population. The ITS stops whenever it finds a solution that respects the mask, or if it can not find any feasible solutions after considering all the population. In this situation, it returns the best solution encountered during the whole process.

Chapter 4

EXPERIMENTAL RESULTS

This chapter is dedicated to the analysis of the performance of our algorithm. First, an analysis of the algorithm process and the advantages of its parallelization is provided. Then, the results achieved by our iterated tabu search during different simulations are discussed and compared to those of the current state-of-the-art method Hcub. To have a better understanding of our analysis and the experiments conducted, it is important to mention a few things first.

– Traditionally, the degree of complexity of a filter is given by its order. From the canonical representation of FIR filters, for MCM algorithms the order of a circuit also corresponds to both the number of multipliers it must replace minus one, and the number of structural adders contained in the circuit architecture it is referring to. In our benchmark, we consider the design of digital filters with different orders and, as a result, different complexity levels.

– It is important to note that the number of adders inside multiplier-less circuits (or adder cost) is commonly used as an estimate of their power consumption. As traditional multiplier-less circuit design methods rely on defined circuit architectures that contains adders, and try to replace multipliers by inserting extra adders, the number of *structural adders*, i.e., the number of adders implied by the architecture the circuit relies on, can be used as a lower bound on the total number of adders in the circuit, and thus of its adder cost. Among the classical FIR circuit architectures, the direct form and the transpose form are the ones that use the smallest number of adders. For a filter of order n both these architectures imply the presence of $n+1$ multipliers and n structural adders.

In order to demonstrate the high level of accuracy of our algorithm, in all our experiments, the number of adders in the circuits constructed by our method is set such that it is equal to the order of the corresponding masks. In this way, we limit the maximum quantity of adders used by our algorithm to the minimal quantity of adders that

could potentially be used by any MCM algorithm.

4.1   Benchmark and algorithm parameters

As discussed before, there are many different parameters in different stages of our algorithm that need to be set. Here we give a list of these parameters and the values we set them to during our tests:

- The Random Structure Generation (RSG) of the algorithm computes $P_{TS} = 100$ starting solutions;
- The tabu search (TS) performs 100 moves for each individual in the population;
- If no SF solution has been found yet, the $P_{ITS} = 10$ best solutions encountered are selected;
- The tabu search inside ITS (intensification) is performed at most 100 times on each circuit, and once every 20 iterations, up to 5 edges of the current solution are erased and reconstructed (diversification);
- While exploring the sub-neighbourhood N_1 and N_2 during both TS and ITS, the algorithm respectively authorizes the presence of up to *maxShift* $= 6$ consecutive shifts and *maxDelay* $= 5$ consecutive delays on each edge.

These values have been calibrated following a set of simulations to determine the influence of each parameter on both the quality of the solution extracted by the algorithm and the computation time needed to identify it. For example, the choice of P_{TS} has been made because we realized that once this parameter was greater than 100, the marginal gain in quality of the solution was small, in comparison to the amount of extra time needed by the algorithm to compute a solution. Furthermore, we created a benchmark containing more than 700 specification masks constructed from the specifications of digital filters of order 10 to 15.

We used this testing benchmark to analyze our algorithm, to evaluate its performance and to compare it to the performance of current state-of-the-art filter design methods. Our benchmark contains 120 masks of order 10, 120 masks of order 11, 120 masks of order 12, 120 masks of order 13, 120 masks of order 14, and 120 masks of order 15. Let us denote that the order of a mask correspond to the order of the filter from which the the mask has been constructed, which, according to the canonical representation of filters, also corresponds to the number of multipliers and the number of structural adders used by classical methods to design the circuits that satisfy the specifications described by the

mask.

4.2   Algorithm analysis

4.2.1   Neighbourhood study

Table 4.1 gives a comparative analysis of the efficiency of the sub-neighbourhoods explored during TS and ITS. In both processes, the neighbourhood N_2, that consists of adding and removing delays inside a circuit, identifies the best local optimum in more than 55% of the cases on average. This observation indicates that one of the main advantages of our algorithm is its ability to determine the right quantity of delays and their placement in the circuit, which is an aspect of the problem that current circuit design algorithms tend to ignore.

4.2.2   Stopping conditions

Table 4.2 shows the distribution of the stopping condition activated to terminate the algorithm. We can see that for smaller orders, TS is usually sufficient to construct accurate circuits. However, when the order is higher, although TS still manages to construct circuits that fit into their specification mask, in most of the cases the algorithm needs to perform ITS to find such circuits.
Besides, the algorithm stops after finding a solution that perfectly respects the mask limits

Table 4.1 Proportion of local optima identified by each sub-neighbourhood while performing TS and ITS.

Order/Adder-cost	Tabu Search			Iterated Tabu Search	
	N_1	N_2	N_3	N_1	N_2
10	26.0%	60.5%	13.5%	44.5%	55.5%
11	24.8%	60.2%	15.0%	45.0%	55.0%
12	25.0%	59.2%	15.8%	48.6%	51.4%
13	26.6%	54.7%	18.7%	39.6%	60.4%
14	26.2%	54.7%	19.1%	40.0%	60.0%
15	24.4%	56.1%	19.5%	42.5%	57.5%
Average	**25.5%**	**57.5%**	**17.0%**	**43.4%**	**56.6%**

Table 4.2 Distribution of the stopping condition activated to terminate the algorithm (Cond. 1a: during TS, Cond. 1b: during ITS, Cond. 2: after ITS) when designing filters of order $n \in [10, 15]$ with circuits containing n adders.

Order/Adder-cost	Stopping condition				
	Cond. 1a	Cond. 1b		Cond. 2	
		Exploration	*Exploitation*	*Exploration*	*Exploitation*
10	79.5%	19.5%	20.5%	1.0%	0.0%
11	61.5%	35.0%	35.0%	3.5%	3.0%
12	58.1%	40.9%	37.6%	1.0%	4.3%
13	53.0%	39.2%	41.0%	7.8%	6.0%
14	35.9%	51.6%	56.4%	12.5%	7.7%
15	23.9%	64.6%	62.4%	11.5%	13.7%
Average	**52.0%**	**41.8%**	**42.2%**	**6.2%**	**5.8%**

Table 4.3 Comparison of the average computation time (s) needed by the algorithm to design filters of order $n \in [10, 15]$ with circuits containing n adders executed using a single core and using 16 cores in parallel.

Order/Adder-cost	Computation time					
	TS		ITS		Total	
	Serial	Parallel	Serial	Parallel	Serial	Parallel
10	341 s	25 s	305 s	31 s	646 s	56 s
11	387 s	28 s	367 s	38 s	754 s	66 s
12	457 s	32 s	402 s	43 s	859 s	75 s
13	508 s	36 s	464 s	49 s	972 s	85 s
14	557 s	40 s	534 s	63 s	1091 s	103 s
15	622 s	44 s	596 s	67 s	1218 s	111 s
Average	**479 s**	**34 s**	**445 s**	**49 s**	**923 s**	**83 s**

in 94.2% of the cases with the exploitation strategy compared to 93.8% with the exploration strategy.

4.2.3 Parallelization

While performing TS and ITS, the circuits in the corresponding populations are considered independently. Thus, both these steps can easily be parallelized in order to reduce the computation time of our algorithm.

Table 4.3 compares the computation times needed to perform TS, ITS and the whole algorithm when it is running sequentially on a single processor (*serial*), or when the design process is distributed on 16 processors (*parrallel*). For each step (i.e., TS, ITS, Total), the time indicated in the table corresponds to the total time needed to perform the step on the entire population of circuits. Distributing the algorithm on several processors drastically improves the time used by the method to construct the best circuit. In average, we obtain a speedup factor (i.e., the ratio between the sequential time and the parallel time) of 11.1 for the whole method, and speedup factors of 13.8 and 9.08 for TS and ITS, respectively.

Let us add that, if the population size parameters P_{TS} and P_{ITS} are divisible by the number of processors used, it clearly improves the usage of the computing resources to
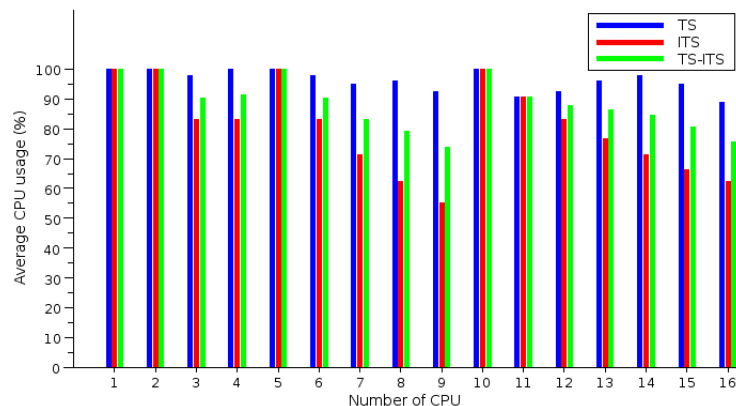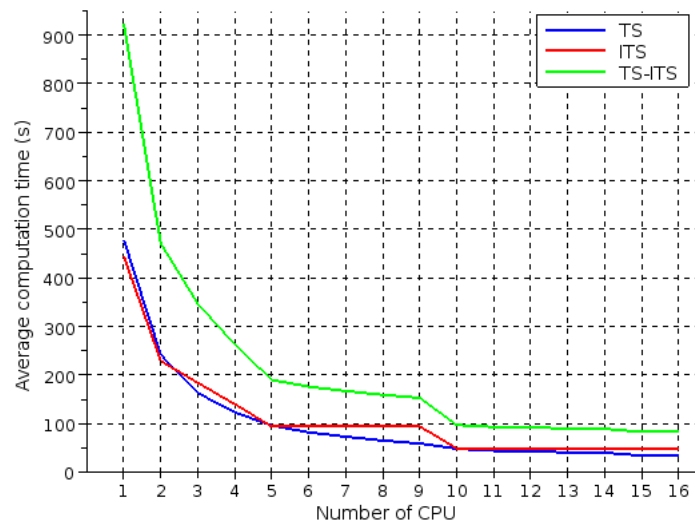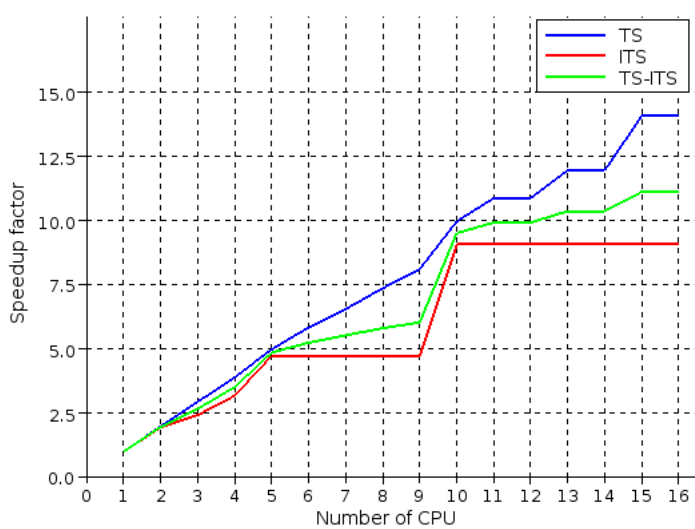

Figure 4.1 Average usage of a CPU (%) according to the number of processors used during the parallelization.

(a)

(b)

Figure 4.2 Evolution of (a) the average computation time (s) and (b) the speedup factor according to the number of CPU used when designing filters for every mask of the benchmark.

construct a circuit. For example, when the algorithm is running on 10 processors, with $P_{TS} = 100$ and $P_{ITS} = 10$, the workload is uniformly distributed as each CPU handles the same number of circuits for both TS and ITS processes (Figure 4.1). In this situation, we almost manage to divide the computation time by the number of CPUs used by reaching a speedup factor of 9.5 (Figure 4.2(b)) for the whole design process, whereas designing a circuit just takes 10 seconds more than when we use 16 processors (Figure 4.2(a)).

4.3 Performance analysis

In order to measure the performance of our algorithm to design a circuit with a magnitude $M(\omega)$ that respects, at best, the specifications defined by a mask $(M^-(\omega), M^+(\omega))$, we propose to use two metrics:

   – The Error Rate (ER), which indicates the proportions of passband and attenuation band frequencies that do not respect the boundaries of the mask. The error rate is given in percentage (%) ;

$$ER = \frac{\sum_{\omega \in \Omega_p} \delta_p(\omega) + \sum_{\omega \in \Omega_a} \delta_a(\omega)}{|\Omega_p| + |\Omega_a|}$$

where:

   – $\delta_p(\omega) = \begin{cases} 1 & \text{if } \omega \in \Omega_p \text{ and } M(\omega) < M^-(\omega) \\ 0 & \text{else} \end{cases}$;

   – $\delta_a(\omega) = \begin{cases} 1 & \text{if } \omega \in \Omega_a \text{ and } M(\omega) > M^+(\omega) \\ 0 & \text{else} \end{cases}$;

   – the Net Error (NE), which indicates the average distance between the circuit magnitude and the mask for the frequencies that do not respect the mask boundaries. The net error is given in decibel (dB);

$$NE = \frac{\sum_{\omega \in \Omega_p} \delta_p(\omega)\left(M^-(\omega) - M(\omega)\right) + \sum_{\omega \in \Omega_a} \delta_a(\omega)\left(M(\omega) - M^+(\omega)\right)}{\sum_{\omega \in \Omega_p} \delta_p(\omega) + \sum_{\omega \in \Omega_a} \delta_a(\omega)}$$

Let us remark that in the case where a solution fully respects the mask boundaries (i.e., $\sum_{\omega \in \Omega_p} \delta_p(\omega) + \sum_{\omega \in \Omega_a} \delta_a(\omega) = 0$), the value of the net error is set to 0.

Table 4.4 Average and standard deviation of the error rate (%) when designing filter of order $n \in [10, 15]$ with circuits containing n adders.

Order/Adder-cost	Error rate			
	Exploration		Exploitation	
	Mean	Std. dev.	Mean	Std. dev.
10	0.1%	0.2%	0.1%	0.2%
11	0.2%	0.4%	0.2%	0.5%
12	0.1%	0.2%	0.2%	0.3%
13	0.2%	0.4%	0.3%	0.5%
14	0.4%	0.9%	0.3%	0.6%
15	0.5%	0.9%	0.4%	0.8%
Average	**0.25%**	**0.50%**	**0.25%**	**0.48%**

Table 4.5 Average and standard deviation of the net error (dB) when designing filter of order $n \in [10, 15]$ with circuits containing n adders.

Order/Adder-cost	Net error			
	Exploration		Exploitation	
	Mean	Std. dev.	Mean	Std. dev.
10	0.01 dB	0.03 dB	0.01 dB	0.03 dB
11	0.04 dB	0.21 dB	0.03 dB	0.06 dB
12	0.03 dB	0.04 dB	0.04 dB	0.12 dB
13	0.06 dB	0.27 dB	0.05 dB	0.07 dB
14	0.05 dB	0.09 dB	0.06 dB	0.11 dB
15	0.11 dB	0.30 dB	0.10 dB	0.20 dB
Average	**0.05 dB**	**0.16 dB**	**0.05 dB**	**0.10 dB**

4.3.1 Design accuracy

The average error rate and net error reached by our algorithm when designing filters for all the masks of the benchmark are shown in Table 4.4 and Table 4.5, respectively. Both of these tables also compare the cases where the ITS follows an exploration or an exploitation strategy.

The results obtained are quite impressive. The method provides filters that respect the boundaries of the masks for 99.75% of the passbands and attenuation bands frequencies. Besides, as indicated in Table 4.5, even for the few cases where the designed circuit specifications do not exactly respect the mask, the error is very small. In average the distance between the circuit frequency response and the mask is only 0.05 dB.

Comparing the results related to when ITS follows exploration or exploitation strategies, we can see that both strategies show very close performances, having the same error rate and net error average on the whole benchmark. However, since for both error rate and net error the standard deviation is smaller in the case of the exploitation strategy, this strategy seems to be slightly more accurate than the exploration strategy.

Considering that we restricted the number of adders in our circuits to the quantity of structural adders implied by the circuit architectures with the lowest adder cost (i.e., the order of the mask), these results are very strong as they demonstrate that our method can provide digital filters that have the same degree of performance than those designed by current design methods, with a much better adder cost, and thus, a much lower energy consumption.

4.3.2 Evolution of the error rate

Figure 4.3 illustrates the evolution of the average error rate (Figure 4.3(a)) and the average net error (Figure 4.3(b)) of the best solution over time. On these figures, we also identified when the transition between the tabu search and the iterated tabu search is taking place. For all the mask orders, it is easy to observe that TS provides a first powerful optimization that tends to converge very rapidly and that ITS is able to help the solution escape from the local optimum identified by TS and to lead the method to a solution with an error very close to 0.

Figure 4.4 and Figure 4.5 compare the evolution of the error rate and the net error in the passband and the attenuation band.
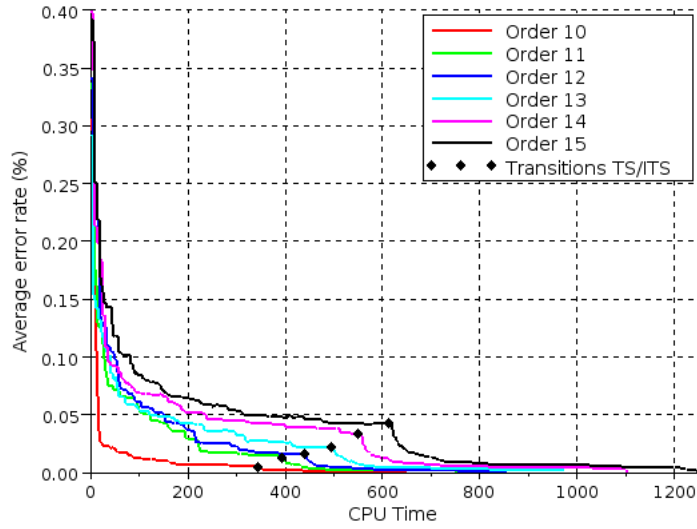
(a)
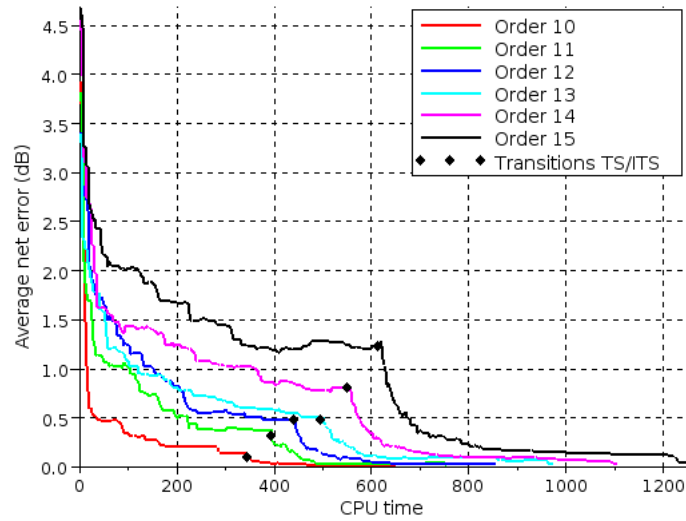
(b)

Figure 4.3 Evolution of (a) the total average error rate (%) and (b) the total average net error (dB) when constructing filters of order $n \in [10, 15]$ with circuits containing n adders.
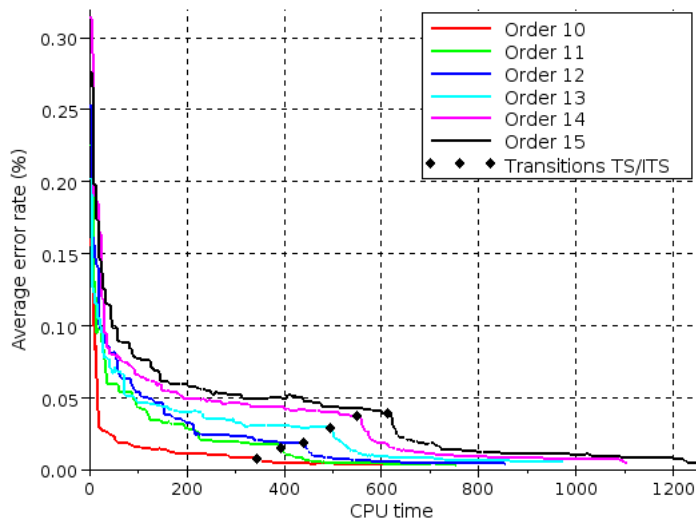
(a)
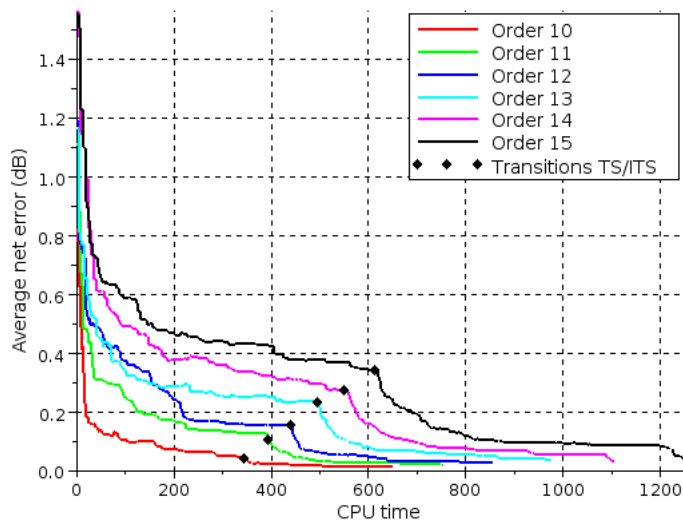
(b)

Figure 4.4 Evolution of (a) the passband average error rate (%) and (b) the passband average net error (dB) when constructing filters of order $n \in [10,15]$ with circuits containing n adders.

(a)

(b)

Figure 4.5 Evolution of (a) the attenuation band average error rate (%) and (b) the attenuation band average net error (dB) when constructing filters of order $n \in [10, 15]$ with circuits containing n adders.
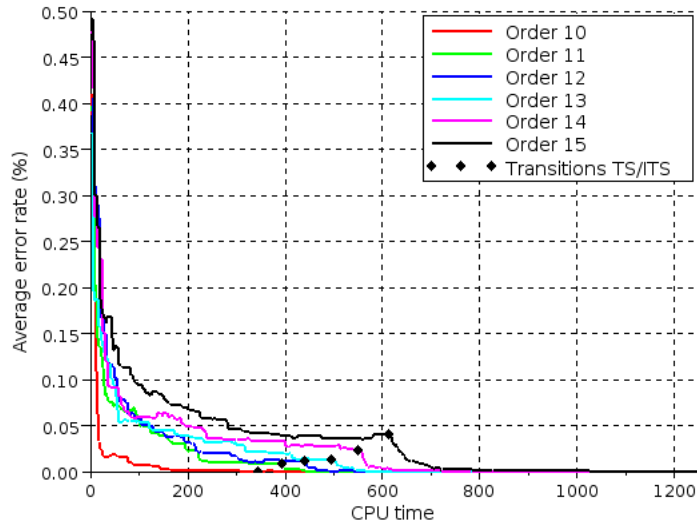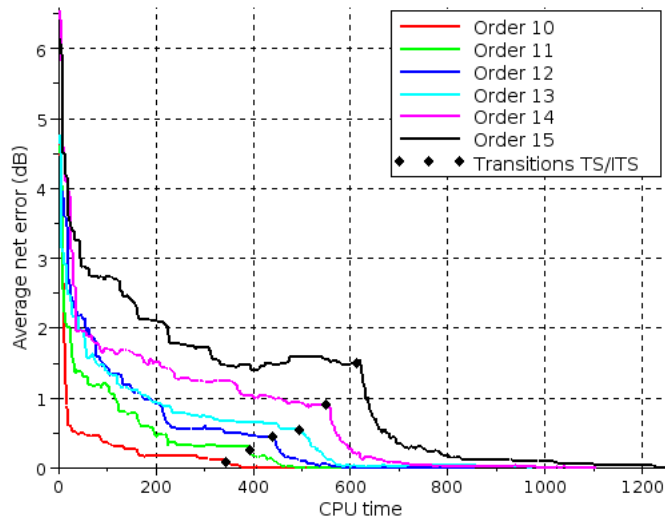
4.4 Power consumption

In order to demonstrate the energy efficiency of the circuits built by our approach, we compare its performance to those of the best current algorithm (Hcub), according to two metrics: the adder cost (i.e., the number of adders in the circuits) and the adder size (i.e., the number of bits needed to perform an addition without losing information). Let us also precise that, besides minimizing the adder cost, Hcub also minimizes the non-output fundamental sum (NOFS), which is the sum of all the different intermediate powers of two used to express the coefficients of the multipliers. In other words, Hcub focuses on decomposing a set of constants into a sum of powers of two, while minimizing both the number of different intermediate powers of two used in this process and the sum of these powers of two.

4.4.1 Minimum adder cost

With MCM algorithms, such as Hcub, the number of adders used in the circuit is related to the filter order, as it indicates the number of structural adders that the circuit will contain, but it also rely on the precision, in terms of bits, used to implement the coefficients of the multipliers: a higher precision leads to a better respect of the specifications as well as a greater number of adders in the circuits. Unlike MCM algorithms, when using our method the user has a direct control on the total number of adders used in the circuits.

Table 4.6 Comparison of the minimum adder-cost needed by our algorithm (TS-ITS) and Hcub to construct circuit with a design error smaller than 1% error.

Order	Average minimum adder cost	
	TS-ITS	Hcub
10	7.7	17.8
11	8.3	18.1
12	8.9	20.8
13	9.6	21.0
14	10.3	23.6
15	11.0	24.2
Average	**9.3**	**20.9**

Table 4.6 compares the minimum number of adders needed by our algorithm and Hcub to design the filter in the benchmark with a maximum tolerated design error of 1%. Results show that, on the average, our algorithm needs 55.5% less adders than Hcub to construct the integrated circuits. The adder-cost being traditionally used as an estimate of the power consumption of multiplier-less integrated circuits, we can thus claim that our algorithm constructs circuits that potentially save more than half of the energy consumed by the integrated circuits designed using the best current method.

4.4.2 Minimum adder size

Another factor that has a great influence on the power consumption of an integrated circuit is the size of its adders. The size of an adder can be defined as the quantity of bits that the adder uses to perform the addition between two numbers in their binary representation. Obviously, the more bits an adder has to consider, the more energy the circuit will consume. In a circuit, the minimum adder size (i.e., the minimum number of bits used to perform the addition between two numbers), directly depends on the shifts, the inverters and the other adders located previously in the circuit. Figure 4.6 shows how the sum of two numbers n_1 and n_2 with the same wordlength of 4 bits (2 bits for the integer part and 2 bits for the fractional part) can lead to a number with a wordlength of 8 bits. This is the result of the following: before addition n_1 is inverted and left shifted of 2 bits and n_2 is left shifted of 1 bit. In this example, the augmentation of the wordlength results from 3 different elements:

– two extra bits are added to the right of the original representation because of the right shift;

– one extra bit is added to the left of the original representation because of the left shift;

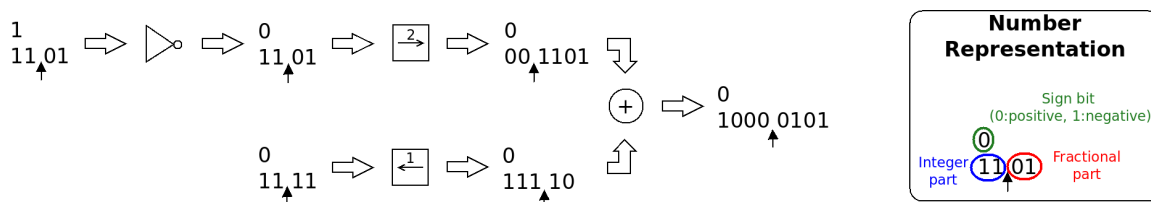– another extra bit is added to avoid overflow in the addition.

Figure 4.6 Binary evolution of two numbers through a small sub-circuit composed of one inverter, one left shift of 1 bit, one right shift of 2 bits and an adder.

From a given integrated circuit and a fixed wordlength W to represent the number sent to the circuit input, one can determine the size $Size(X_i)$ of each adder in the circuit by the formula:

$$\forall i \in [1,N], Size(X_i) = W + L(X_i) + R(X_i),$$

where

– The function $L(X_i)$ indicates the number of extra bits on the left needed by the adder X_i according to the left boundary of W:

$$\begin{cases} L(X_0) = \quad\quad 0 \\ L(X_i) = \max \Big(\quad L\big(ArgTail(Edge_1(X_i))\big) + log_2\big(|P(Edge_1(X_i))|\big), \\ \quad\quad\quad\quad L\big(ArgTail(Edge_2(X_i))\big) + log_2\big(|P(Edge_2(X_i))|\big)\Big) + 1, \quad \text{if } i \in [1,N]; \end{cases}$$

– The function $R(X_i)$ indicates the number of extra bits on the right needed by the adder X_i according to the right boundary of W:

$$\begin{cases} R(X_0) = \quad\quad 0 \\ R(X_i) = \max \Big(\quad R\big(ArgTail(Edge_1(X_i))\big) - log_2\big(|P(Edge_1(X_i))|\big), \\ \quad\quad\quad\quad R\big(ArgTail(Edge_2(X_i))\big) - log_2\big(|P(Edge_2(X_i))|\big)\Big), \quad \text{if } i \in [1,N]; \end{cases}$$

It is important to remark that, as the function L and R respectively consider the left bound and the right bound of the initial wordlength W, this can lead to a negative number. For example, $L(X_i) = -2$ means that the left bound of the word computed by the adder i is located two bits on the right of the left bound of the initial words. Figure 4.7 gives an example of the computation of the adder size inside an integrated circuit.
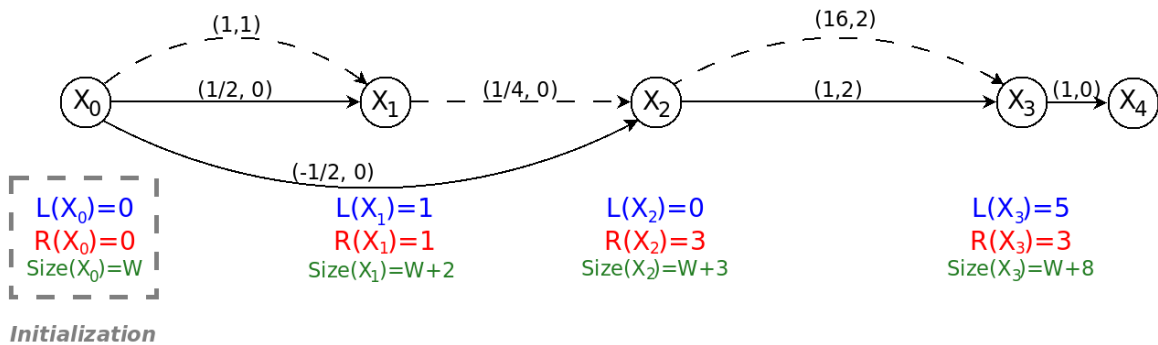

Figure 4.7 Computation of the size of adders in an integrated circuit.

From these formulas, we extracted the average minimum adder size of the circuits designed in the previous subsection (Table 4.7). Our results are based on the assumption that $W = 0$, i.e., the table only shows the quantity of extra bits added to the initial wordlength. It is important to note that when Hcub tries to design the circuits using the fewest quantity of adders, it also minimizes the non-output fundamental sum (NOFS) in order to reduce the size of the adders.

Table 4.7 Comparison of the average minimum adder size in the circuits designed by our algorithm (TS-ITS) and Hcub (NOFS minimization) with a maximum design error is fixed to 1%.

Order	Average minimum adder size	
	TS-ITS	Hcub
10	7.7	12.1
11	8.7	12.3
12	9.4	13.1
13	8.7	13.5
14	10.0	14.6
15	9.6	14.8
Average	**9.0**	**13.4**

According to the results shown in Table 4.7, it is clear that our method not only decreases the number of adders used in the circuits, but it also drastically reduces the size of the adders, that is 33% less compared to those used by Hcub.

This reduction in the adder size can be explained by the fact that our algorithm tends to use much less shifts than MCM algorithms. In fact, in our algorithm the adder size is implicitly controlled by the parameter *maxShift* that indicates the maximum bits by which a number can be shifted in a connection path. In order to have a better control on the adder size while constructing the circuit, we could even define a range in which a number can be shifted, i.e., we could limit the maximum authorised adder size in the circuit, which is an option that cannot be considered by current multiplier-less filter design algorithms.

4.4.3   Quantity of delay

By having no restriction on the circuit structure, we drastically increase the degree of freedom of our algorithm. A part of this freedom reflects in the fact that our algorithm can insert delays at any place in the circuits, whereas existing methods are restricted to the location of the delays in the architecture they refer to.

Table 4.8 Comparison of the number of delays in the circuits designed by our algorithm (TS-ITS) and Hcub with a design error smaller than 1%.

Order	Average number of delays	
	TS-ITS	Hcub
10	10.8	10.0
11	12.3	11.0
12	13.9	12.0
13	16.0	13.0
14	17.0	14.0
15	17.6	15.0
Average	**14.6**	**12.5**

By comparing the number of delays used in the circuits designed with our algorithm and Hcub (Table 4.8), we can see that, on the average, our algorithm needs 17% more delays when compared to other methods. Fortunately, the complexity of a delay being equivalent to 20% of that of an adder and the level of energy consumed by a delay being directly related to the wordlength inside the circuit, and thus of the adder size, the loss in complexity and energy efficiency due to the extra delays incorporated by our algorithm is not really significant in comparison to the gain caused by the much smaller adder cost and adder size.

4.4.4   Energy efficiency

Traditionally, the amount of energy $\mathscr{E}(F)$ consumed by a mutliplier-less filter F is estimated by the quantity of adders $\mathscr{A}(F)$ used in the circuit that implements F, also known as the adder cost of F:

$$\mathscr{E}(F) \equiv \mathscr{A}(F) \tag{4.1}$$

However, the metric $\mathscr{E}$ lacks precision as it does not consider the adder size and the role that delays play. Although the direct influence of shifts and inverters is not really significant in the power consumption, each delay has an impact that can be estimated as 20% of the impact of an adder. Moreover, the adder size is another important aspect of the energy consumption since addition of two numbers coded using n bits uses only half the energy needed for the addition of two numbers coded using $2n$ bits, the same way than memorizing a number coded using n bits uses half the energy needed for memorizing a number coded using $2n$ bits. From these observations, we propose a new metric $\mathscr{E}'(F)$ that measures the energy consumption of a multiplier-less filter F according to the number of adders $\mathscr{A}(F)$, the average size of these adders $\mathscr{S}(F)$ (which also corresponds to the average number of bits used to code a number in the circuit), and the number of delays $\mathscr{D}(F)$ inside its circuit implementation:

$$\mathscr{E}'(F) \equiv \mathscr{S}(F) * (\mathscr{A}(F) + 0.2 * \mathscr{D}(F)) \tag{4.2}$$

From the results obtained in the previous subsections according to the adder cost, the adder size, and the number of delays, based on the metric $\mathscr{E}'(F)$ we can expect a global reduction of the energy consumption of the order of 65% with our algorithm in comparison to the results achieved by the best current algorithm Hcub.

4.4.5 Free structure of the integrated circuits

Figure 4.8 compares the structure of two integrated circuits constructed by our method (Fig. 4.8(a)) and the algorithm Hcub (Fig. 4.8(b)) to design a lowpass filter with the same specifications (i.e., a cutoff frequency of $\omega_p = 0.5$, an attenuation band frequency of $\omega_a = 0.8566$, a maximum deviation in the passband of $r_p = 1$dB, and an attenuation level of $Att_a = 50$dB).

As clearly seen in the figure, in the circuit designed by Hcub, the separation between the parallel multiplier block and the architecture to which the method is referring (in this example, the transpose form) is obvious, whereas it is hard to identify any kind of pattern in the structure of our circuit and that is the reason why we consider the integrated circuits designed by our method as *free structure* integrated circuits.
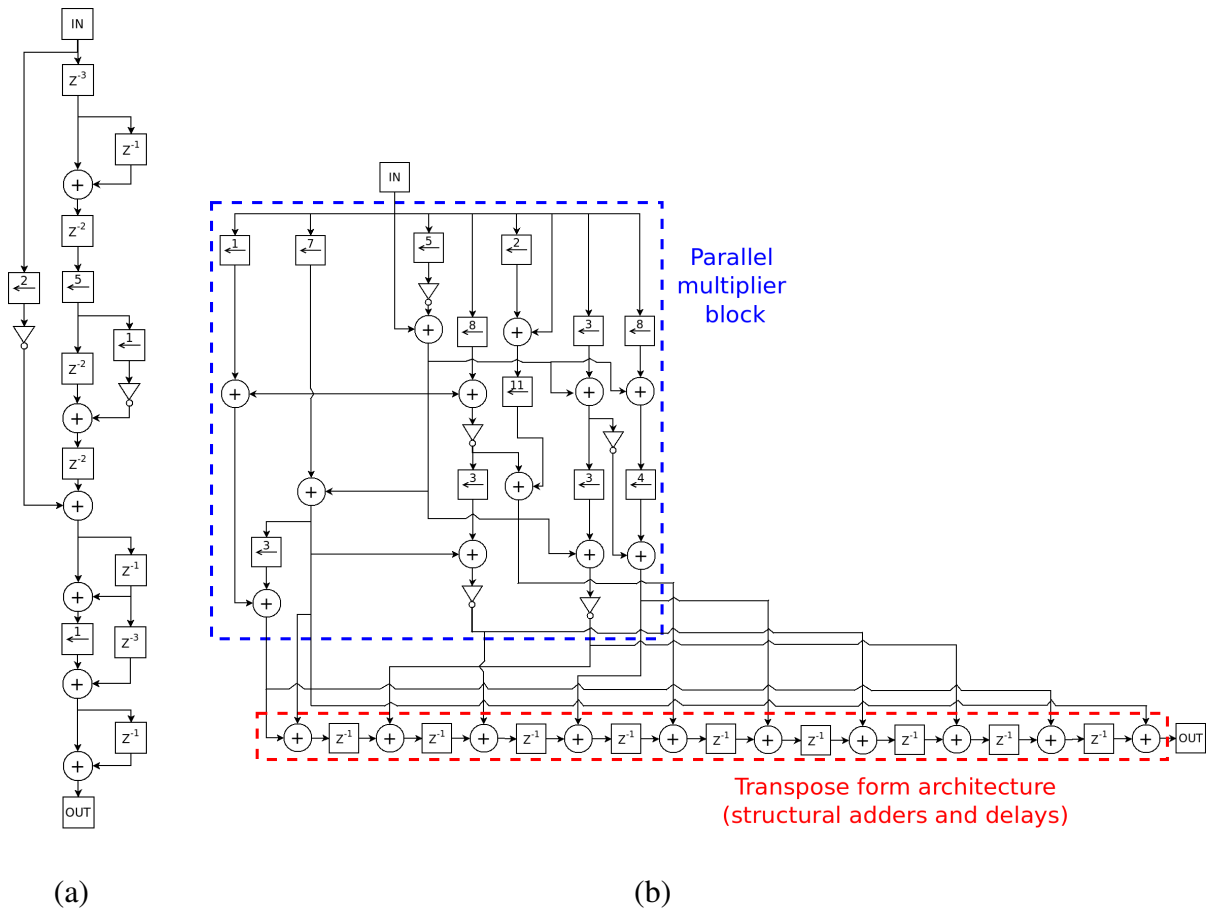
(a)   (b)

Figure 4.8 Illustration of the integrated circuit designed to implement a lowpass filter ($\omega_p =$ 0.5, $\omega_a = 0.8566$, $r_p = 1$dB, $Att_a = 50$dB) using (a) our algorithm and (b) Hcub.

Chapter 5

CONCLUSION

5.1 Summary

With the recent advancements in microelectronics and the improvements made to current wireless telecommunication systems, the growing interest in more complex devices that can handle more and more sophisticated tasks everyday has made energy consumption an issue worth addressing. To this end, we propose a novel approach for designing energy-efficient integrated circuits for performing the digital signal processing tasks in hand.

Unlike other existing algorithms that refer to certain circuit architectures, our method designs circuits that correspond to filters with the same desired specifications as those of the current methods, but that more or less follow a free structure. This allows the algorithm to globalize its efforts for minimizing the number of adders in the circuit rather than optimizing parts of the circuit by replacing multipliers with adders in a portion of the circuit, as the traditional methods do.

One of the main advantages of our algorithm is that it requires no more adders than the minimum number of adders needed by the existing methods in the unlikely perfect situation where all the coefficients would be powers of two, which in terms of energy consumption is significant since the adder-cost of a circuit (i.e., the number of adders used in the circuit structure) is a good estimate of the energy consumption of the circuit.

To reach these results, we first introduced a new graph-based model for representation of the circuit structures and we then defined a population based algorithm that combines a tabu search and an iterated tabu search along with destruct and reconstruct strategy to iteratively insert, place and remove components (adders, delays, inverters and shifts) in the circuit until it reaches a set of desired specifications defined on the frequency domain.

While designing our algorithm, we decided that it would be interesting to give the user some control over the number of adders that can be used in the circuit structure. Unlike the other existing algorithms that really do not have much flexibility in choosing the number of adders, our algorithm focuses on finding the best possible circuits that contain only this

number of adders and fulfill the filter requirements. We are thus able to present the user with circuits that have a fixed degree of power consumption.

Other than the direct control on the number of adders used in the circuit, the algorithm provides an implicit control on the number of shifts than can be used by setting certain parameters, these parameters in turn affect the number of bits needed for performing the additions in the circuit and, as a result, the adder size and the energy consumed by the adders. No other algorithm gives this much freedom to the user in terms of choice of the elements in a circuit and its structure. The number of delays used in our circuits are slightly higher than those of other algorithms but since the complexity of a delay is equivalent to 20% of that of an adder this does not affect the performance of our algorithm much and our algorithm still outperforms the existing algorithms by far.

In order to test the performance of our filters and compare it to that of state-of-the-art algorithms, we performed a variety of simulations on a benchmark containing more than 700 digital filters. Even though our algorithm constructs circuits with an adder-cost limited to the quantity of adders implied by the best circuit architectures to which current design algorithms refer to, it is still able to produce solutions that respect the desired specifications for 99.75 % of the frequencies in the passband and attenuation bands. Besides, in the few cases where the specifications are not fully respected, the corresponding net error of 0.05 dB that occurs, is not really significant in practice.

The comparison between the proposed method and the state-of-the-art Multiple Constant Multiplication algorithm Hcub (heuristic cumulative benefit) also proves that, with a tolerated error rate less than 1%, circuits built by our algorithm need 55% less adders with a size 33% smaller, at the cost of 17% more delays. Altogether, these results lead to a total reduction of the filter energy consumption in the order of 65%.

5.2   Limitations of the proposed solution

Although our method achieves very good results, like any other method it has certain limitations. One of which can be due to our model. As in its definition we did not allow any recursive loops in the circuit structure, it can only be used for designing FIR filters. However, we believe that we can extend our model to adapt it for designing Infinite Impulse Response (IIR) filters.

One of the other points worth more attention is the implicit control the algorithm has on the wordlength. This can be fixed by improving the design of the algorithm for it to fix the

wordlength according to what the user needs. Like many other filter design algorithms, our algorithm has difficulty dealing with filters of very large order since the process demands great computational resources. This issue could be solved by making some improvements in the algorithm for it to be able to decompose the problem into smaller more manageable sub-problems.

5.3 Future work

For future work we could consider several different directions to follow. One of the most important might be to focus on the identification of new metrics that could be directly considered by the algorithm in order to increase the precision of the power consumption estimation of the circuits. It could also be very interesting to adapt our model and method to non-recursive circuits for designing IIR filters. It can also be fruitful to manipulate the parameters for the maximum number of shifts and delays used and analyze the performance of the filters, as well as the effect on the energy consumption. We also plan to study the application of this innovative approach to the design of other types of integrated circuits for other purposes, including equalizers, modulators or data converters. Finally, we would like to stress that, although our algorithm may not fit into any traditional classification for designing digital filters, the results reached by it make it worth consideration and it and other algorithms that follow its premise could form a new class on their own.

REFERENCES

AHMAD, S. et ANTONIOU, A. (2006). Cascade-form multiplierless FIR filter design using orthogonal genetic algorithm. *IEEE International Symposium on Signal Processing and Information Technology*. pp. 932–937.

AVIZIENIS, A. (1961). Signed-digit number representation for fast parallel arithmetic. *IRE Transanction on Electronics and Computer*, vol. 10, pp. 389–400.

BERNSTEIN, R. (1986). Multiplication by integer constants. *Software - Practice and experience*, vol. 16, pp. 641–652.

BOUDJELABA, K., CHIKOUCHE, D. et ROS, F. (2011). Evolutionary techniques for the synthesis of 2-D FIR filters. *2011 IEEE Statistical Signal Processing Workshop (SSP)*. pp. 601–604.

BULL, D. et HORROCKS, D. (1991). Primitive operator digital filters. *IEE Proceedings G - Circuits, Devices & Systems*. pp. 401–412.

CEMES, R. et AIT-BOUDAOUD, D. (1993). Genetic approach to design of multiplierless FIR filters. *Electronics Letters*, vol. 29, pp. 2090–3001.

CHÂTELAIN, B. et GAGNON, F. (2007). Multiplierless evolutionary filter design. *International Symposium on Signals, Circuits and Systems*. pp. 175–178.

COLEMAN, J. (2001). Cascaded coefficient number systems lead to fir filterrs of striking computational efficiency. *Proceeding of the International IEEE Conference in Electronics, Circuits and Systems*. pp. 513–516.

CRANIC, T., GENDREAU, M. et FARVOLDEN, J. (2000). Simplex-based tabu search for multicommodity capacitated fixed charge network design problem. *INFORMS Journal on Computing*, vol. 12, pp. 223–236.

DE WERRA, D. et HERTZ, A. (1989). Tabu search techniques: a tutorial and an application to neural networks. *OR Spektrum*, vol. 11, pp. 131–141.

DEMPSTER, A. et MACLEOD, M. (1995). Use of minimum-adder multiplier blocks in FIR digital filters. *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 42, pp. 569–577.

DEY, A., SAHA, S., SAHA, A. et GHOSH, S. (2010). Method of Genetic Algorithm (GA) for FIR Filter Construction: Design and Development with Newer Approaches in

Neural Network Platform. *International Journal of Advanced Computer Science and Applications*, vol. 1, pp. 87–90.

ECK, B. (1989). *Good solutions to job shop scheduling problems via tabu search*. Columbia University. Research report, Department of Industrial Engineering and Operations Research.

FANNI, A., MARCHESI, M., PILO, F. et SERRI, A. (1998). Tabu search metaheuristic for designing digital filters. *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 17, pp. 789–796.

FRIDEN, C., HERTZ, A. et DE WERRA, D. (1989). Stabulus:A technique for finding stable sets in large graphs with tabu search. *Computing*, vol. 42, pp. 35–44.

GENDREAU, M., GUERTIN, F., POTVIN, J. et TAILLARD, D. (1999). Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, vol. 33, pp. 381–390.

GENDREAU, M. et POTVIN, J. (2010). *Handbook of Metaheuristics*, Springer, chapitre Tabu search. pp. 41–59.

GLOVER, F. (1977). Heuristics for Integer Programming Using Surrogate Constraints. *Decision Sciences*, vol. 8, pp. 156–166.

GLOVER, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, vol. 13, pp. 533–549.

GLOVER, F. (1989a). *Candidate list strategies and tabu search*. University of Colorado. CAAI Research Report.

GLOVER, F. (1989b). Tabu search part I. *ORSA Journal on Computing*, vol. 1, pp. 190–206.

GLOVER, F. (1990). Tabu Search: A Tutorial. *Special Issue on the Practice of Mathematical Programming, Interfaces*, vol. 20, pp. 74–94.

GLOVER, F. et LAGUNA, M. (2002). *Tabu Search*. Kluwer Academic Publications.

GLOVER, F. et MCMILLAN, C. (1986). The general employee scheduling problem: An integration of management science and artificial intelligence. *Computers and Operations Research*, vol. 13, pp. 563–593.

GUSTAFSSON, O., DEMPSTER, A., JOHANSSON, K., MACLEOD, M. et WANHAMMAR, L. (2006). Simplified design of constant coefficient multipliers. *Circuits, Systems and Signal Processing*, vol. 25, pp. 225–251.

HANSEN, P. (1986). *The Steepest Ascent Mildest Descent heuristic for combinatorial programming*. Presented at the congress on Numerical Methods in Combinatorial Optimization.

HERTZ, A. et DE WERRA, D. (1987). Using tabu search techniques for graph coloring. *Computing*, vol. 29, pp. 345–351.

HERTZ, A. et DE WERRA, D. (1991). The tabu search metaheuristic: how we used it. *Annals of Mathematics and Artificial Intelligence*, vol. 1, pp. 111–121.

JOLIVEAU, M., GIARD, P., GENDREAU, M., GAGNON, F. et THIBEAULT, C. (2011). Design of low complexity multiplierless digital filters with optimized free structure using a population-based metaheuristic. *Proceedings of the 10th International Symposium on Signals, Circuits and Systems (ISSCS)*. pp. 1–4.

KALINLI, A. et KARABOGA, N. (2005). Parallel tabu search algorithm for digital filter design. *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 24, pp. 1284–1298.

KARABOGA, D., HORROCKS, D., KARABOGA, N. et KALINLI, A. (1997). Designing digital FIR filters using Tabu search algorithm. *Proceedings of 1997 IEEE International Symposium on Circuits and Systems. Circuits and Systems in the Information Age*. pp. 2236–2239.

KARABOGA, N., KALINLI, A. et KARABOGA, D. (2004). Designing digital IIR filters using ant colony optimisation algorithm. *Engineering Applications of Artificial Intelligence*, vol. 17, pp. 301–309.

KNOX, J. et GLOVER, F. (1989). *Comparative testing of traveling salesman heuristic derived from tabu search, genetic algorithms and simulated annealing*. University of Colorado. Center for Applied Artificial Intelligence.

KODEK, D. et STEIGLITZ, K. (1981). Comparison of optimal and local search methods for designing finite wordlength FIR digital filters. *IEEE Transactions on Circuits and Systems*, vol. 28, pp. 28–32.

LEE, A., AHMADI, M., JULLIEN, G., MILLER, W. et LASHKARI, R. (1998). Digital filter design using genetic algorithm. *Proceedings of IEEE Symposium on Advances in Digital Filtering and Signal Processing*. pp. 34–38.

LIAN, Y. et CEN, L. (2003). A genetic algorithm for the design of low power high-speed FIR filters. *Proceedings of Seventh International Symposium on Signal Processing and its Applications*. pp. 181–184.

LING, C. (2007). A hybrid genetic algorithm for the design of FIR filters with SPoT coefficients. *Signal Processing*, vol. 87, pp. 528–540.

LING, C. et YONG, L. (2004). A modified micro-genetic algorithm for the design of multiplierless digital FIR filters. *2004 IEEE Region 10 Conference*. pp. 52–55.

LOKKETANGEN, A. et GLOVER, F. (1996). *Meta-Heuristics: Theory and Applications*, Kluwer Academic Publications, chapitre Probabilistic move selection in tabu search for 0/1 mixed integer programming problems. pp. 467–488.

LOKKETANGEN, A. et WOODRUFF, D. (1996). Progressives hedging and tabu search applied to mixed integer (0,1) multistage stochastic programming. *Journal of Heuristics*, vol. 2, pp. 111–128.

MACLEOD, M. et DEMPSTER, A. (2005). Multiplierless FIR filter design algorithms. *IEEE Signal Processing Letters*, vol. 12, pp. 186–189.

MALEK, M., GURUSWAMY, M., OWENS, H. et PANDYA, M. (1989a). Serial and parallel search techniques for the traveling salseman problem. *Annals of OR:Linkages with Artificial Intelligence*.

MALEK, M., HEAP, M., KAPUR, R. et MOURAD, A. (1989b). *A fault tolerant implementation of the traveling salseman problem.* University of Texas-Austin. Research report, Department of Electrical and Computer Engineering.

MISEVICIUS, A., LENKEVICIUS, A. et RUBLIAUSKAS, D. (2006). Iterated tabu search: an improvment to standard tabu search. *Information Technology and Control*, vol. 35, pp. 187–197.

ONER, M. (1998). A genetic algorithm for optimisation of linear phase FIR filter coefficients. *Thirty-Second Asilomar Conference on Signals, Systems and Computers*. pp. 1397–1400.

PASKO, R., SCHAUMONT, P., DRUDDER, V., VERNALDE, S. et DURACKOVA, D. (1999). A new algorithm for elimination of common subexpression. *IEEE Transactions on Computer-Aidded Design of Integrated Circuits and Systems*, vol. 18, pp. 58–68.

SAMUELI, H. (1989). An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients. *IEEE Transactions on Circuits and Systems*, vol. 36, pp. 1044–1047.

SHING-TAI, P. (2010). A canonic-signed-digit coded genetic algorithm for designing finite impulse response digital filter. *Digital Signal Processing*, vol. 20, pp. 314–327.

SMITH, J. (2007). *Introduction to Digital Filters: with Audio Applications*. W3K Publishing.

SORIANO, P. et GENDREAU, M. (1996). Diversification strategies in tabu search algorithms for maximum clique problems. *Annals of Operations Research*, vol. 63, pp. 198–207.

SUCKLEY, D. (1991). Genetic algorithm in the design of FIR filters. *IEE Proceedings G (Circuits, Devices and Systems)*. pp. 234–238.

THONG, J. et NICOLICI, N. (2009). Time-efficient single constant multiplication based on overlapping digit patterns. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. pp. 1353–1357.

TRAFERRO, S., CAPPARELLI, F., PIAZZA, F. et UNCINI, A. (1999). Efficient allocation of power of two terms in FIR digital filter design using tabu search. *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems VLSI*. pp. 411–414.

TRAFERRO, S. et UNCINI, A. (2000). Power-of-two adaptive filters using tabu search. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, pp. 566–569.

VORONENKO, Y. et PUSCHEL, M. (2007). Multiplierless multiple constant multiplication. *ACM Transactions on Algorithm*, vol. 3, pp. 1–38.

WADE, G., ROBERTS, A. et WILLIAMS, G. (1994). Multiplier-less FIR filter design using a genetic algorithm. *IEEE proceedings-vision,image and signal proceesing*. pp. 175–180.

WATCHARASITTHIWAT, K., JEERASUDA, K. et PARAMOTE, W. (2006). Designing digital FIR filters using multiple tabu search algorithm. *Proceedings of International Conference on Communications, Circuits and Systems, ICCCAS*. pp. 171–175.

XU, D. et DALEY, M. (1995). Design of optimal digital filter using a parallel genetic algorithm. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, pp. 673–675.

XU, X. et NOWROUZIAN, B. (1999). Local search algorithm for the design of multiplierless digital filters with CSD multiplier coefficients. *IEEE Canadian Conference on Electrical and Computer Engineering*. pp. 811–816.