Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Operations Research 32 (2005) 2671–2687
computers & operations research
www.elsevier.com/locate/dsw

A portable and scalable algorithm for a class of constrained combinatorial optimization problems

Sancho Salcedo-Sanz*, Carlos Bousoño-Calzón

Department of Signal Theory and Communications, Universidad Carlos III de Madrid, Leganes, Madrid 28911, Spain

Abstract

This paper presents a portable and scalable approach for a class of constrained combinatorial optimization problems (CCOPs) which requires to satisfy a set of constraints and to optimize and objective function simultaneously. In particular, this paper is focused on the class of CCOPs that admits a representation in terms of a square matrix of constraints C.

The algorithm consists of a hybrid neural-genetic algorithm, formed by a Hopfield Neural Network (HNN) which solves the problem's constraints, and a Genetic Algorithm (GA) for optimizing the objective function. This separated management of constraints and optimization procedures makes the proposed algorithm scalable and robust. The portability of the algorithm is given by the fact that the HNN dynamics depends only on the matrix C of constraints.

We show these properties of scalability and portability by solving three different CCOPs with our algorithm, the frequency assignment problem in a mobile telecommunications network, the reduction of the interference in satellite systems and the design of FPGAs with segmented channel routing architecture. We compare our results with previous approaches to these problems, obtaining very good results in all of them.

Keywords: Combinatorial optimization; Genetic algorithms; Hopfield neural networks; Hybrid algorithms; Scalability; Portability

1. Introduction

A class of important combinatorial optimization problems requires to satisfy a set of constraints and to optimize (minimize or maximize) an objective function simultaneously. Problems of this kind arise profusely in computational engineering and computer science fields. Scheduling problems

* Corresponding author. Tel.: +34-91-624-5973; fax: +34-91-624-8749.
E-mail address: sancho@tsc.uc3m.es (S. Salcedo-Sanz).

0305-0548/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2004.03.020

[1,2], assignment problems [3–6,27] or problems of systems design [7–9], are some examples. In this paper, we refer to this kind of problems as constrained combinatorial optimization problems (CCOPs).

Several methods which have been reported as good approaches to concrete examples of CCOPs can be found in the literature [10–14,30,31]. However, these existing algorithms have two main drawbacks: first, the majority of them show lack of scalability. This means that its performance is poor when the size of problems grows. Second, algorithms proposed for CCOPs are usually not portable. By "portable" we mean the algorithm which can be used in two different CCOPs with very few changes. The majority of the existing algorithms are designed to solve a particular problem, being necessary major changes in the algorithm's structure to apply it to another CCOP, when possible.

In this paper, we propose an algorithm for solving a class of CCOPs with very good properties of scalability and very portable. The algorithm works by separating the fulfilment of the constraints from the optimization process. A Genetic Algorithm (GA) is used for the optimization of the objective function, and a local procedure (Hopfield neural network, HNN) for solving the problem's constraints. This separated treatment of objective function and constraints gives our algorithm the desired properties of scalability. Thus, we will show that our algorithm outperforms the existing algorithms in large and difficult examples of CCOPs.

The characterization of the algorithms as "portable", in the sense that could be used for a wide variety of CCOPs with very few changes, comes from an accurate design of the local procedure for solving the problem's constraints. In order to do this, this paper focuses on the class of CCOPs whose solutions admit a representation as an $N \times M$ binary matrix X, with components $x_{ij} \in \{0, 1\}$; and problem's constraints can be represented in an $N \times N$ integer matrix C, the so-called *matrix of constraints*, where every element c_{ij} stands for the minimum separation in columns between two elements of X. That is, any two elements in the solution, $x_{ik} = 1$ and $x_{jl} = 1$, must fulfil:

$$|k - l| \geqslant c_{ij}$$

in order the solution to be feasible.

This class of CCOPs is characterized only by its matrix of constraints C, as well as its objective function $f(X)$.[1] The HNN we use as local procedure is designed in such a way that its dynamics only depends on the matrix C; this way, just by changing the matrix of constraints and the associated objective function associated, our algorithm is directly applicable to any CCOP considered without the need for any more adaptations.[2]

The rest of the paper is structured as follows: in the next section the class of CCOPs we deal with are defined and analyzed. In Section 3, the proposed algorithm is described, by studying the HNN and the GA which form it. Section 4 shows the performance of the proposed algorithm in terms of scalability and portability, by solving three CCOPs, and comparing the results obtained with the results of different algorithms in the literature. Finally, Section 5 closes the paper.

[1] Of course function f could depend not only on X but also in other parameters, different for each problem. However, we will denote it as $f(X)$ remarking its dependence on X.

[2] Next section gives some examples of CCOPs that belong to the class of problems we focus on.

2. Problem definition and analysis

The algorithm presented in this paper deals with CCOPs of the following characteristics: let $\wp$ be an optimization problem whose solutions can be represented as an $N \times M$ binary matrix, X. If it is possible to characterize $\wp$ by an objective function $f(X)$ and an $N \times N$ matrix of constraints C, where each c_{ij} represents the minimum distance in columns between 1s in X, i.e. if $x_{il} = 1$ and $x_{jk} = 1$ then $|l - k| \geqslant c_{ij}$, $\wp$ belongs to the class of CCOPs we are interested in.
Mathematically, these CCOPs can be defined as:
find an $N \times M$ binary matrix X, such as:

$$max(f(X)) \tag{1}$$

and subject to:

$$|k - l| \geqslant c_{ij} \quad \text{if } x_{ik} = 1,\ x_{jl} = 1. \tag{2}$$

Note that the problem is completely defined by the function $f(X)$ and the matrix of constraints C.

There are a lot of CCOPs that admit the above representation. The most classical example of this characterization is the frequency assignment problem in a mobile communication network [3,5,7,15]. In this problem, every element of the matrix C, c_{ij}, represents the minimum distance between frequencies in the communication network needed to avoid interferences, whereas $f(X)$ is the number of frequencies assigned, see [3,4] for details.

Scheduling problems, such as broadcast scheduling in Packet Radio Networks (PRN) also can follow this characterization, with c_{ij} the minimum allowed distance in slots of time between two stations in the PRN, in order to avoid loss of packets [1,9].

We can cite other problems that admit this characterization, such as VLSI and FPGA design [16], reduction of interferences in satellite systems [17,18] or an important class of problems which solutions are permutations (Travelling Salesman Problem, Job Shop Scheduling) [19,20]. The constraints in problems encoded as permutations may also be represented by a matrix C:[3]

$$c_{ij} = \begin{cases} \infty & \text{if } i = j, \\ 1 & \text{otherwise}, \end{cases} \tag{3}$$

with this definition of the matrix C, X will have one 1 per row and column, representing a permutation.

Thus, the class of CCOPs this paper manages includes a wide variety of important problems in computational science.

3. Proposed approach

The algorithm we propose for solving the class of CCOPs mentioned, consists of a hybrid global–local scheme, where a global algorithm looks for the optimization[4] of the objective function

[3] Hereafter, the symbol ∞ in the definition of c_{ij} stands for any number greater than M.

[4] This optimization could be minimization or maximization of the function, however, since a minimization process can be seen as the maximization of the function $f' = -f$, in this paper we will only consider maximization.

and a local procedure manages the fulfilment of problem's constraints. As global algorithm we use a standard GA, due to several reasons. First, they are known to be robust search algorithms [21], second, the standard GA codify the solutions as binary strings, so the solution matrix X can be codified in a GA as a binary string. This allows the construction of hybrid search schemes, mixing global and local algorithms. The separation of the global search from the constraints fulfilment gives our algorithm the desired properties of scalability, outperforming existing algorithms in large difficult CCOPs, as we will show in Section 4.

The properties of portability comes from an accurate design of the local search procedure.

3.1. The local search procedure

The local search procedure consists of a kind of HNN, whose dynamics only depends on the matrix C, and on the initial states of the neurons. This Hopfield network belongs to a class of digital Hopfield networks, where the neurons only can take the values 1 or 0, see [22] for further details. The structure of the HNN can be described as a graph, where the set of vertices are the neurons, and the set of edges defines the connections between the neurons. We map a neuron to every element in the solution matrix X. In order to simplify notation, we shall also use matrix X to denote the neurons in the Hopfield network. The HNN dynamics can be described then in the following way: after a random initialization of every neuron with binary values, the HNN operates in serial mode. This means that only a neuron is updated at time, while the rest remain unchanged. Denoting by $x_{ij}(t)$ the state of a neuron in time t, the update rule is described by:

$$x_{ij}(t) = isgn\left(\sum_{\substack{p=1\\p\neq i}}^{N}\sum_{\substack{q=max(1,c_{i,p+1})\\q\neq j}}^{min(M,j+c_{i,p})} x_{pq}\right) \quad \forall i,j, \tag{4}$$

where the *isgn* operator is defined by:

$$isgn(a) = \begin{cases} 0 & \text{if } a > 0, \\ 1 & \text{otherwise.} \end{cases}$$

Note that the update rule only takes into account neurons x_{pq} equal to 1 and within a distance of c_{ip} in columns of the element x_{ij} being updated. Note also that in this updating rule, the neurons x_{ij} are updated in its natural order, i.e., $i = 1,2,\ldots,N$, $j = 1,2,\ldots,M$. A slight modification of this rule is performed by means of updating the neurons in a random ordering of the rows (variable i). This way the variability in the feasible solution found is increasing. Let $\pi(i)$ be a random permutation of $i = 1,2,\ldots,N$. The new updating rule of the HNN results:

$$x_{\pi(i)j}(t) = isgn\left(\sum_{\substack{p=1\\p\neq \pi(i)}}^{N}\sum_{\substack{q=max(1,c_{\pi(i),p+1})\\q\neq j}}^{min(M,j+c_{\pi(i),p})} x_{pq}\right) \quad \forall i,j. \tag{5}$$

The resulting updating rule runs over the rows of X in the order given by the permutation $\pi(i)$, but the columns are updated in natural order $j = 1,2,\ldots,M$.

A *cycle* is defined as the set of $N \times M$ successive neuron updates in a given order. In a cycle, every neuron is updated once following the given order $\pi(i)$, which is fixed during the execution of

the algorithm. After every cycle, the convergence of the HNN is checked. The HNN is considered converged if none of the neurons have changed their state in the cycle. The final state of the HNN dynamics is a feasible solution, in the sense that it fulfils the constraints of the matrix C.

3.2. The global search algorithm

The optimization of the objective function is performed in our approach by means of a GA, in the following way: a solution X is codified in the GA as a $(N \times M)$-length binary string. The GA population is formed by a fix number of $(N \times M)$-length strings, χ, which codify several solutions to the problem. These solutions are called individuals of the population. The population is then evolved through successive generations by means of the application of the genetic operators: selection, crossover and mutation [21].

Selection is the process by which individuals are randomly sampled with probabilities proportional to their fitness values, which, in this case, is the value of $f(X)$. An elitist strategy, consisting in always passing the highest fitness string to the next generation, is applied in order to preserve the best solution encountered thus far in the evolution. The selected set, of the same size of the initial population, χ, is subjected to the crossover operation. Firstly, the binary strings are coupled at random. Secondly, for each pair of strings, an integer position along the string is selected uniformly at random. Two new strings are composed by swapping all bits between the selected position and the end of the string. This operation is applied to the couples with probability P_c less than one.

By means of the mutation operation, every bit in every string of the population may be changed from 1 to 0, or vice versa, with a very small probability, P_m.

Finally, since crossover and mutation operators may cause the new string to be infeasible, this string is set as the initial state of the HNN, and the result of the neural algorithm substitutes it in the new population.

3.3. The complete algorithm

Given a CCOP defined by an objective function $f(X)$ and a matrix of constraints C, the complete algorithm we propose can be represented in the following way:

Algorithm proposed: GA and HNN
  Initialize GA population at random

while(max. number of generations not reached) **do**
  for(every individual X)
    run the HNN to obtain a feasible X
    calculate $f(X)$
endfor
  selection
  crossover
  mutation
end while

The dynamics of the HNN is completely defined by the matrix of constraints C, whereas the GA can be run knowing its fitness function $f(X)$, with the initial population generated at random. Two different CCOPs defined by $f_1(X), C_1$ and $f_2(X), C_2$, respectively, will be solved by the algorithm without any more changes than the routine for the calculation of $f_1(X)$ and $f_2(X)$.

4. Experiments

The test of the algorithm proposed should show its performance quality and also the good properties of portability claimed. In order to do that, we will solve three different CCOPs using our algorithm: the frequency assignment problem (FAP) in a mobile communication network, the reduction of interference in satellite systems by the reassignment of channels and the design of FPGAs row-based segmented channel routing architecture. For every problem we give a brief description of the problem and its mathematical formulation, the corresponding function $f(X)$ and matrix of constraints C, as well as the solution obtained with our algorithm compared with other algorithms' solution from the literature.

We have used the standard values for the GA parameters $P_c = 0.6$ and $P_m = 0.01$ [21] in all the problems tackled, with a population size $\chi = 50$ individuals.

4.1. The FAP

4.1.1. Brief description

The Frequency Assignment Problem (FAP hereafter) in a mobile communications network can be stated as follows: given a mobile communication network formed by N cells, a set of M available frequencies and vector of frequencies requested of length $N, \mathbf{v}$ (v_i the number of frequencies requested by cell i), achieve an assignment of frequencies to every cell, in such a way that the system is free from interferences.

The minimum distance between frequencies in the system in order to avoid interferences is given by an integer matrix D. The solution of the problem can be represented as an $N \times M$ binary matrix X, where the rows represents the cells and the columns represents the available frequencies. An element $x_{ij} = 1$ means that the frequency j has been assigned to the cell i. Some articles which deal with this problem are [3–5,7,15].

Mathematically the problem can be defined as:

find an assignment X, such as, if $x_{ik} = 1$ and $x_{jl} = 1$:

$$|k - l| \geqslant d_{ij} \tag{6}$$

with the number of requested frequencies v_i in every cell. The function $f(X)$ is the number of frequencies which fulfil Eq. (6), i.e.

$$max\left(f(X) = \sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij}\right) \tag{7}$$

Note that this problem has exactly the form we defined in Section 2 for our algorithm to be applied.

Table 1
Main features of the FAP benchmark problems considered

Problem	Number of cells (N)	Number of frequencies (M)
1	4	11
2	25	73
3	21	381
4	21	533
5	21	533
6	21	221
7	21	309
8	21	309

4.1.2. Results

In this problem the general matrix of constraints C is equal to the matrix D defined above. The number of frequencies in every cell is computed in the routine for the calculation of $f(X)$. Only v_i frequencies are valid for every cell, the rest, if any, are not taking into account. In order to test the performance of our algorithm in this problem we have tackled eight benchmark problems from [23]. The main characteristics of these problems are shown in Table 1. The compatibility matrices and request vectors can be found in [23]. Our algorithm achieves the optimal result in every benchmark problem considered. This shows the very good performance of our approach. However, in order to have further comparison with other techniques, we have tested our algorithm in a simulated GSM network, taking from [24]. In this network, calls are randomly generated in the communication network following a Poisson distribution of parameter λ (calls/sg), which is increased with time. It is supposed that one call is served by one frequency, this way, the request vector is formed by the number of calls in the network in a given time. When the number of calls in the network is higher than the number of available frequencies, the call is rejected. The duration of every call is simulated as a random variable which follows an exponential distribution, with an average duration of 50 s. Fig. 1 shows the performance of our algorithm managing the frequencies requested in the network, compared with two algorithms: a fixed assignment of the frequencies and a HNN as the one in [23]. Note that our algorithm achieves better results than the other two algorithms (smaller probability of rejecting a call).

4.2. The interference reduction in satellite systems problem

4.2.1. Brief description

This problem consists of, given two adjacent satellite systems (Fig. 2), reducing the inter-system co-channel interference by means of rearranging the frequency assignment of carriers system #2, whereas the assignment in system #1 remains fixed, as references see [17,18]. This problem admits a formulation in which the solution is represented by an $N \times M$ binary matrix $\tilde{X}$, where N stands for the carriers in system #1 and M stands for the segments (divisions of a carrier) in system #2. Since every carrier has a different length in frequency, an $N \times N$ matrix L is defined, in such a way that l_{ij} stands for the minimum separation in segments allowed between carriers i and j. The final solution matrix X $(M \times M)$ is obtained from matrix $\tilde{X}$ $(N \times M)$ in a straight forward manner by

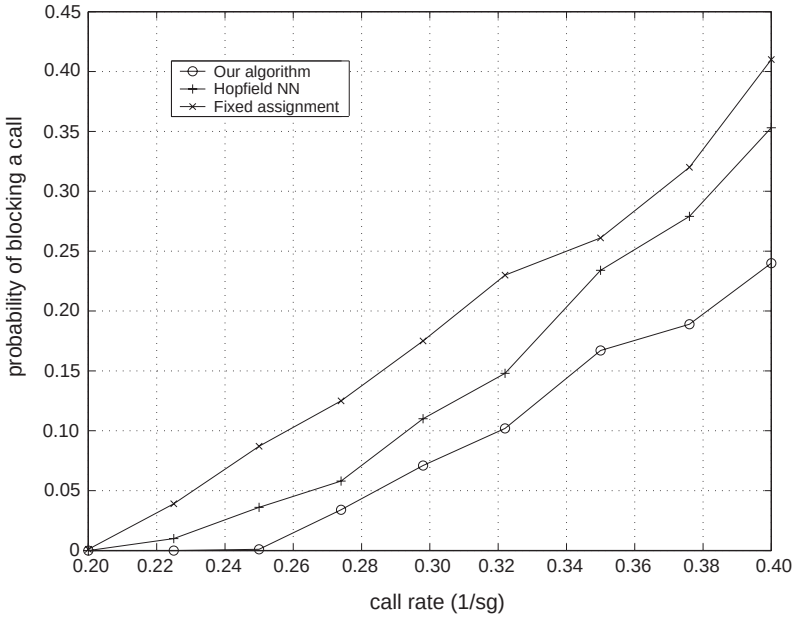


Fig. 1. Performance of different algorithms assigning frequencies in a mobile communications network.

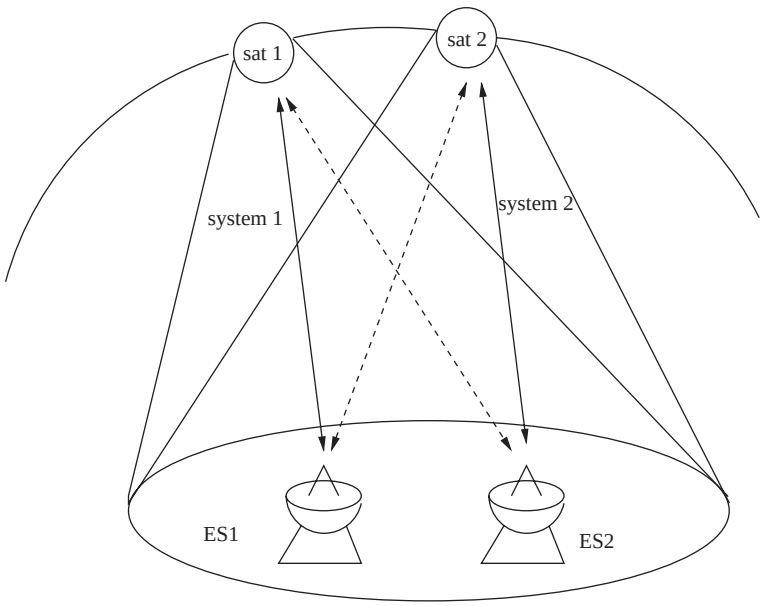


Fig. 2. Outline of co-channel interference.

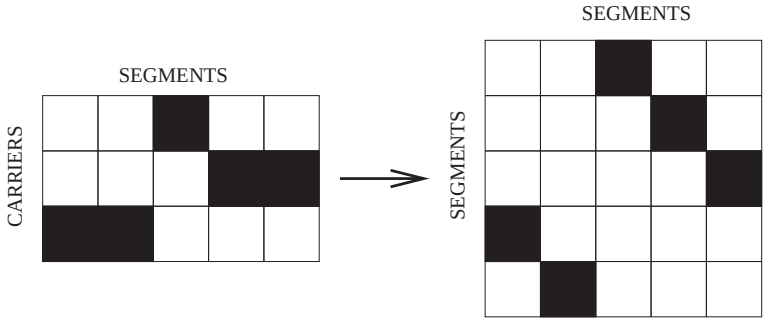


Fig. 3. Example of expansion from matrix ($\tilde{X}$) of carriers–segments to matrix X of segments–segments.

expanding the rows from carriers (N) into segments (M). Fig. 3 gives an example of this step. In order to calculate the objective function $f(X)$, an interference matrix $M \times M, E$, is defined. Every element of this matrix, e_{ij}, stands for the cost in interference of associating a segment in system #2 to other segment in system #1. Thus, the problem consists of reassigning carriers in system #2 in such a way that the total interference, or the largest, interference of the system to be minimum. The mathematical formulation of the problem is as follows:
achieve X such that:

$$min\left(f(X)=\sum_{i=1}^{M}\sum_{j=1}^{M}e_{ij}x_{ij}\right) \tag{8}$$

subject to:

$$\sum_{i=1}^{N}x_{ij}=1, \quad j=1,\ldots,M \tag{9}$$

and in such a way that the assignment fulfils the constraints in L.

The constraints associated to this problem can be represented in a unique matrix C, which is defined as follows:

$$c_{ij}=\begin{cases}\infty & \text{if } i=j,\\ l_{ij} & \text{otherwise.}\end{cases} \tag{10}$$

4.2.2. Results

A set of Benchmark problems from [17] have been selected in order to test the performance of our algorithm in this problem. The set of Benchmarks are formed by five problems of increasing difficulty: There are two easy problems, #1 and #2, one problem of medium difficulty, #3, and two hard problems, #4 and #5. Table 2 shows the main characteristics of the problems.

Our algorithm solves the problem obtaining good quality solutions. Table 3 shows the results obtained and a comparison with other two algorithms results, a Gradual Hopfield Neural Network

Table 2
Main features of the satellite benchmark problems considered

Problem #	Carriers (N)	Segment (M)	Range of carrier	Range of interf.
1	4	6	1–2	5–55
2	4	6	1–2	1–9
3	10	32	1–8	1–10
4	10	32	1–8	1–100
5	10	32	1–8	1–1000

Table 3
Total and largest interference for satellite benchmark problems

Problem #	BB [18] total	GNN [17] total	Ours total	GNN [17] largest	Ours largest
1	100	100	100	30	30
2	13	13	13	4	4
3	91	85	85	7	7
4	929	880	838	64	64
5	10330	8693	6851	640	640

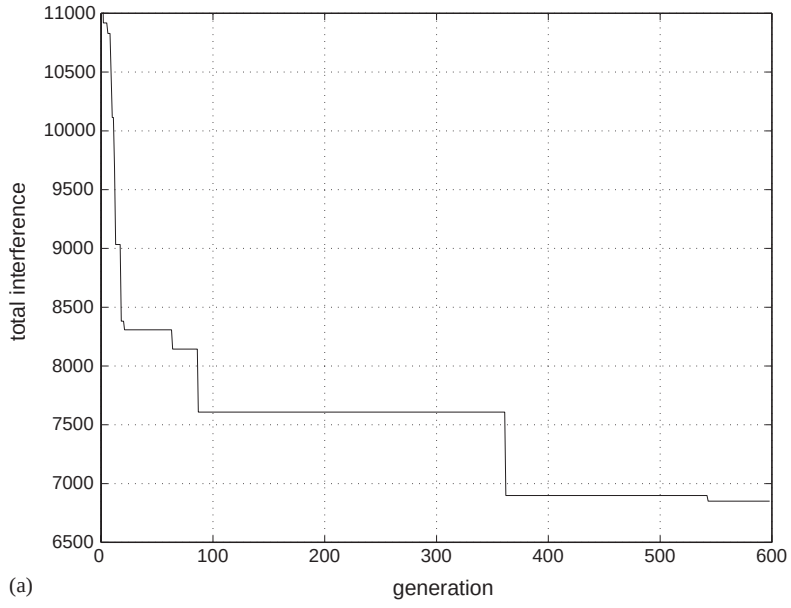
(GNN) and a Branch and Bound algorithm (BB), for the Benchmarks problems tackled. Total interference and largest interference fitness measures have been considered. Note that our algorithm achieves equal or better results than other algorithms. The differences are more pronounced in the hardest problems, #4 and #5 with total interference as the objective function. This shows that our algorithm is more scalable than existing algorithms for this problem. Fig. 4(a) and (b), shows the evolution of the algorithm in the experiment where the best result in terms of total interference was achieved, and its corresponding solution, for the hardest problem #5.

4.3. The FPGA segmented channel routing problem

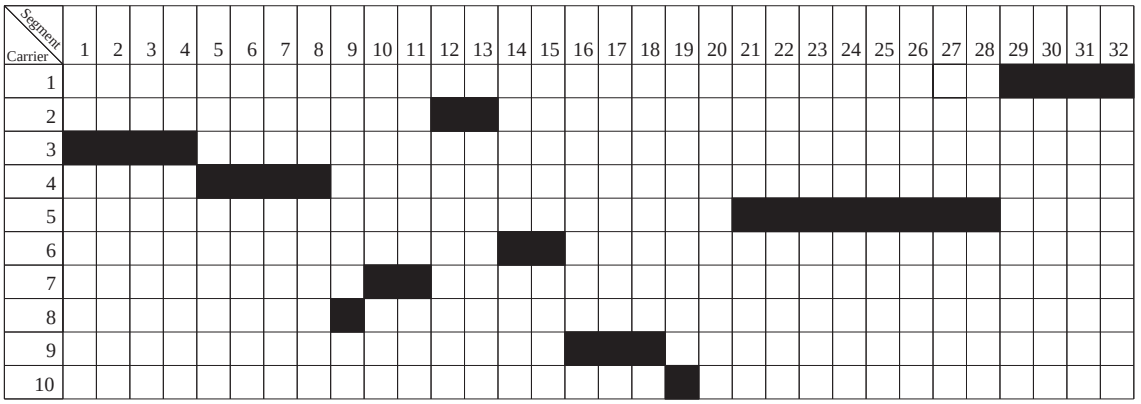
4.3.1. Brief description

The FPGA Segmented Channel Routing Problem (FSCRP) is a CCOP in which a set of N nets must be assigned to M tracks with L columns, in order to design a FPGA. Every track is divided into several horizontal segments. There exist constraints among the nets, in such a way that not all the nets can share the same track. Every net is described by a pair of leftmost and rightmost columns to be interconnected, denoted $left_i$ and $right_i$ for net i. In this article we consider that two nets can share a track if they do not share any column. This approach is slightly different from the standard segmented channel routing problem, where nets only can share a track if they do not share any segment in the channel. In addition an $N \times M$ matrix of *antifuses*,[5] F, is defined, where

[5] An antifuse is a programmable switch that can be located between two adjacent horizontal segments. Each antifuse provide a low resistance bi-directional interconnection between segments if it is required, see [16] for details.



(a)



(b)

Fig. 4. (a) Evolution of the algorithm in the experiment where the best result of total interference was obtained (problem #5), (b) Best solution found (matrix $\tilde{X}$) by our algorithm in the same problem.

every element f_{ij} is 1 if there is an antifuse between columns j and $j+1$ on track i. The goal of the FSCRP is finding a conflict free track assignment of nets with the minimum total number of programmed antifuses. See [16,25]. Fig. 5(a) shows an example of a FSCRP. There are six antifuses (circles), five nets, ten columns and three tracks. Note that nets #2 and #3 and nets #4 and #5 could be assigned to the same track, due to no column is shared by them, however, net #1 must not share track with nets #2, #3 and #4. Fig. 5(b) shows a feasible solution for this problem, of cost 2.

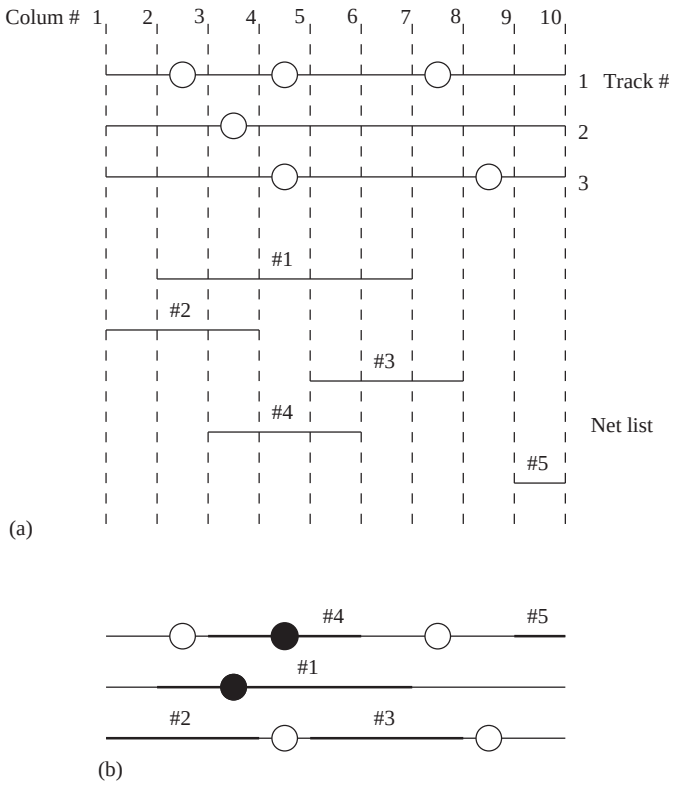


Fig. 5. (a) FPGA segmented channel routing problem (the circles stand for antifuses). (b) A feasible routing solution.

Mathematically, the FSCRP can be formulated as follows:

Let X be a binary $N \times M$ assignment matrix in which every component $x_{ij} = 1$ means that net #i has been assigned to track #j, and $x_{ij} = 0$ stands for no assignment. Let

$$w_{ij} = \sum_{k=left_i}^{right_i - 1} f_{ik} \tag{11}$$

be the routing cost associated with net i when it is assigned to track j. In a channel with M tracks, N nets and a matrix of antifuses F, FSCRP requires to find an assignment X such that:

$$min\left(f(X) = \sum_{i=1}^{N} \sum_{j=1}^{M} w_{ij} x_{ij} \right) \tag{12}$$

subject to:

$$\sum_{i=1}^{M} x_{ij} = 1 \quad \text{and} \quad \sum_{j=1}^{M} \sum_{\substack{k=1 \\ k \neq i}}^{N} d_{ijk} x_{ij} x_{kj} = 0 \quad \forall i, \tag{13}$$

where d_{ijk} is 1 if nets i and k can share the same segment on track j, and 0 otherwise.

Table 4
Main features of FPGA benchmark problems

Problem #	Tracks (M)	Columns (L)	Nets (N)	Avg. net length	#fuses
1	8	30	32	4.96	47
2	8	30	32	5.12	78
3	8	30	32	6.06	84
4	16	48	86	5.97	143
5	16	48	86	6.39	156
6	16	48	86	6.2	175

Table 5
Comparison of the results obtained by our algorithm and the two greedy algorithms considered for the FPGA instances

Problem #	Ours	m-FSCR	l-FSCR
1	11	13	15
2	38	43	42
3	39	45	44
4	74	N/A	86
5	83	N/A	N/A
6	91	N/A	N/A

Taking into account the definitions above, it is possible to describe a given FSCRP instance in terms of an $N \times N$ matrix of constraints, C, where every element c_{ij} stands for the minimum distance required between nets i and j, and it is defined as:

$$c_{ij} = \begin{cases} \infty & \text{if } i = j, \\ 0 & \text{if } (i \neq j) \text{ and } [(left_i \geqslant right_j) \text{ or } (left_j \geqslant right_i)], \\ 1 & \text{otherwise.} \end{cases} \quad (14)$$

4.3.2. Results

In order to test the performance of our algorithm in this problem, six instances have been generated following the indications given in [16]. Table 4 shows the specifications of the simulated instances. The results obtained by our algorithm are compared with the results obtained by two greedy algorithms taken from [26], which have been used previously for comparison in the literature. The first greedy algorithm, so-called m-FSCR, sequentially assigns each net to one track with the minimum cost among the available tracks, in descending order of net length. The second greedy algorithm, l-FSCR, assigns each net from leftmost to rightmost nets. Both heuristics incorporate a backtracking procedure to cope with the case of no available track for a net can be found.

The results obtained by our algorithm, and its comparison with the results obtained by the two greedy algorithms considered, are shown in Table 5. The symbol N/A means that no feasible solution was found by the corresponding algorithm. Note that our algorithm finds a feasible solution in every

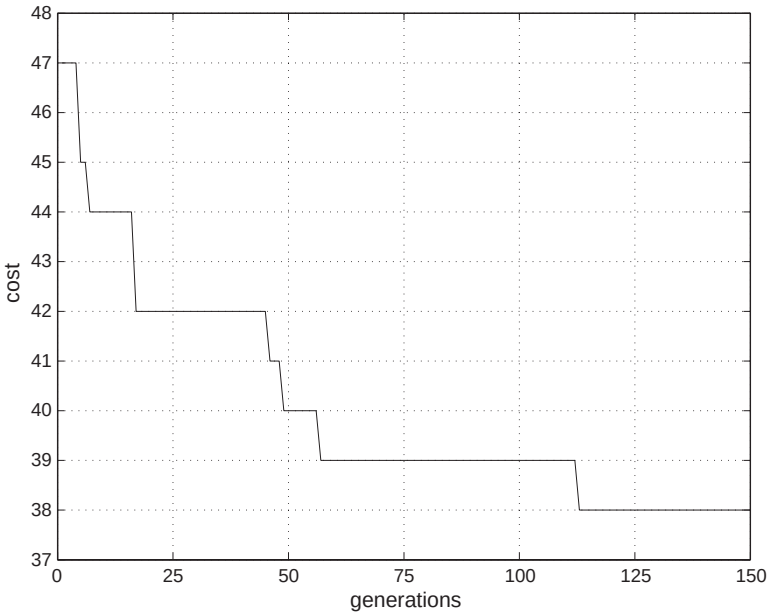


Fig. 6. Best genetic algorithm's evolution in FPGA instance #2.

problem, whereas m-FSCR only achieves a feasible solution in instances #1,#2 and #3, and l-FSCR found a feasible solution in instances #1, #2, #3 and #4. In all of them our algorithm outperforms both greedy algorithms, as can be seen in Table 5.

Fig. 6 shows the evolution of our algorithm in instance #2, (best total fitness of the population). Note how the best solution obtained by our algorithm is improved along the generations by the GA.

4.4. Some remarks on the performance of the digital HNN

The HNN used in this paper is a fast digital network, with very good properties of convergence. In this section we provide some insight into the performance of the HNN proposed, by evaluating it on the satellite interference reduction problems given in Section 4.2.

First, we have tested the properties of convergence of the HNN, by launching 1000 nets for each problem, and computing the number of unfeasible solutions achieved. Note that a solution for a given interference reduction problem is unfeasible only if there is any carrier in system #2 not assigned to the corresponding segments in system #1. Fig. 7 shows the results obtained: in problems #1 and #2 all the HNNs launched provide a feasible solution, as expected, since the size of these problems is small. In problems #3, #4 and #5 the percentage of convergence to feasible solutions was similar, about 98%. These results show that the HNN achieves a feasible solution in the majority of runs, starting from an unfeasible solution.

Second, we analyze the speed of convergence of the network. In order to do this, we analyze the solutions provided by the HNN to the interference reduction problem #5. In this problem, once

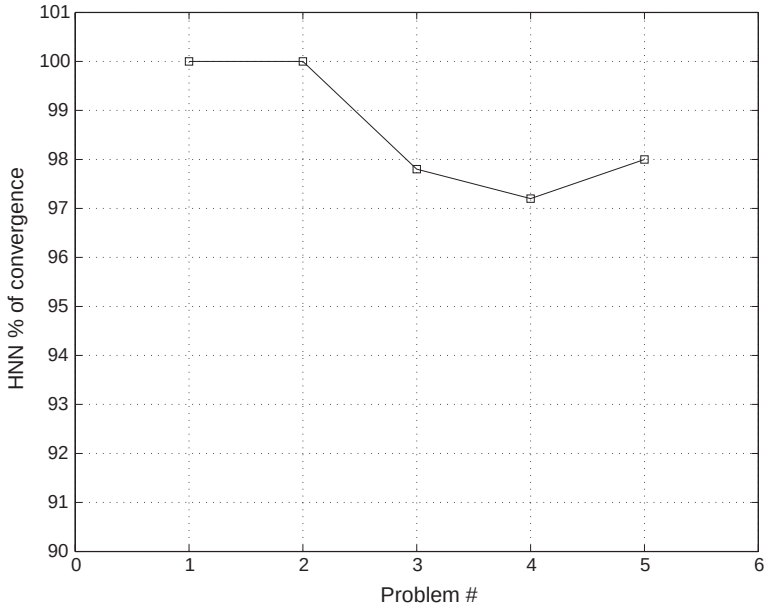


Fig. 7. HNN percentage of convergence in satellite interference reduction problems considered, #1–#5.

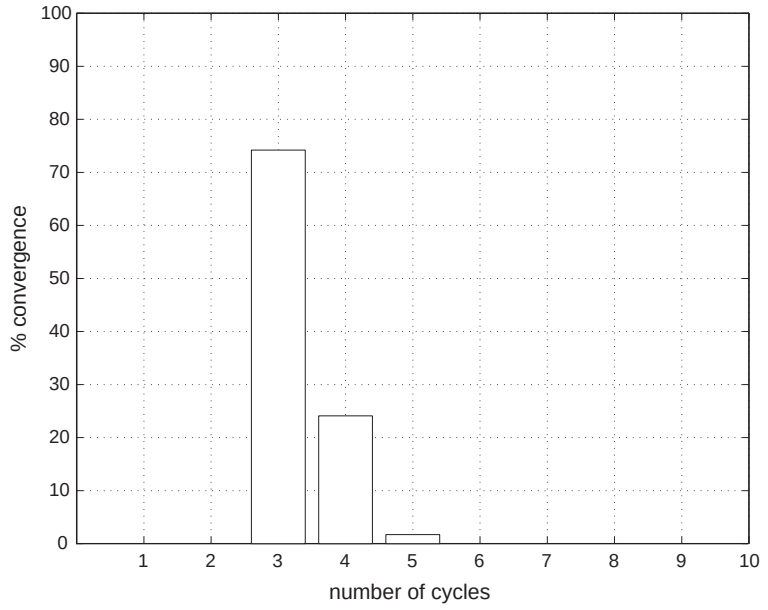


Fig. 8. Number of cycles needed for the HNN convergence, in satellite interference reduction problem #5.

the HNN provides a feasible solution, we have calculated the amount of cycles[6] that the HNN has needed for reaching the convergence. Fig. 8 shows that the 75% of the HNNs launched for solving problem #5 converged in 3 cycles. Over the 23% of the networks launched converged in 4 cycles, and only about the 2% of the networks run converged in 5 cycles. The new updating rule introduced in Section 3.1, does not modify these values, because it only changes the order of updating, not the structure of the algorithm.

In real time, the complete algorithm (HNN and GA) solved the problems #1 and #2 within a few seconds, and problems #3, #4 and #5 in about 2 min. The total time consumed by the algorithm strongly depends on the maximum number of generations of the GA. Depending on the problem, the number of generations in which the algorithm can be considered converged varies. For solving problems #3, #4 and #5 we obtained very good solutions using a maximum number of 600 generations for the GA.

5. Conclusions

In this paper, a portable and scalable algorithm for a class of constrained combinatorial optimization problems (CCOPs) has been presented. The algorithm consists of a bi-dimensional Hopfield neural network (HNN), which depends on a matrix C of constraints. It is hybridized with a genetic algorithm (GA) for optimizing the problem's objective function.

We have shown that there are CCOPs which admit a representation in terms of an objective function $f(X)$ and a matrix C of constraints. This fact ensures the portability of the algorithm, since no modification in the HNN is needed for managing the problem's constraints. The scalability of the algorithm is given by the separate management of constraints and optimization process.

We have tested the performance of our algorithm in three different CCOPs, outperforming previous algorithms on them, and showing the portability and scalability of our approach.

References

[1] Wang G, Ansari N. Optimal broadcast scheduling in packet radio networks using mean field annealing. IEEE Journal on Selected Areas in Communications 1997;15(2):250–9.
[2] Levine D. Application of a hybrid genetic algorithm to airline crew scheduling. Computers & Operations Research 1996;23(6):547–58.
[3] Funabiki N, Okutani N, Nishikawa S. A three-stage heuristic combined neural-network algorithm for channel assignment in cellular mobile systems. IEEE Transactions on Vehicular Technology 2000;49(2):397–403.
[4] Lai WK, Coghill GC. Channel assignment through evolutionary optimization. IEEE Transactions on Vehicular Technology 1996;55(1):91–5.
[5] Hao JK, Dorne R, Galinier P. Tabu search for frequency assignment in mobile radio networks. Journal of Heuristics 1998;4(1):47–62.
[6] Smith K, Palaniswami M. Static and dynamic channel assignment using neural networks. IEEE Journal on Selected Areas in Communications 1997;15(2):238–49.
[7] Sandalidis HG, Stavroulakis PP, Rodriguez-Tellez J. An efficient evolutionary algorithm for channel resource management in cellular mobile systems. IEEE Transactions on Evolutionary Computation 1998;2(4):125–37.

[6] Recall that a cycle is defined as the updating of all the neurons in the Hopfield network, see Section 3.1.

[8] Kim H, Hayashi Y, Nara K. An algorithm for thermal unit maintenance scheduling through combined use of GA, SA and TS. IEEE Transactions on Power Systems 1997;12(1):329–35.
[9] Salcedo-Sanz S, Bousoño-Calzón C. A mixed neural-genetic algorithm for the broadcast scheduling problem. IEEE Transactions on Wireless Communications 2003;2(2):277–83.
[10] Smith K, Palaniswami M, Krishnamoorthy M. A hybrid neural approach to combinatorial optimisation. Computers & Operations Research 1996;23(6):597–610.
[11] Wang CJ, Tsang EPK, Solving constraint satisfaction problems using neural-networks. In: Proceedings of the IEE Second International Conference on Artificial Neural Networks, 1991. p. 295–9.
[12] Tsang EPK, Wang CJ. A generic neural network approach for constraint satisfaction problems. In: Taylor G, editor. Neural network applications. Berlin: Springer; 1992. p. 12–22.
[13] Balicki J, Stateczny A, Zak B. Genetic algorithms and Hopfield neural networks for solving combinatorial problems. Applied Mathematics and Computer Science 1997;7(3):568–92.
[14] Watanabe Y, Mizuguchi N, Fujii Y. Solving optimization problems by using a Hopfield neural network and genetic algorithm combination. Systems and Computers in Japan 1998;29(10):68–73.
[15] Voudouris C, Tsang EPK, Solving the radio link frequency assignment problem using guided local search proceedings. In: Proceedings of the NATO Symposium on Radio Length Frequency Assignment, Sharing and Conservation Systems (Aerospace), Aalborg, Denmark, October 1998.
[16] Funabiki N, Yoda M. A gradual neural-network approach for FPGA segmented channel routing problems. IEEE Transactions on System, Man and Cybernetics Part B 1999;29(4):481–8.
[17] Funabiki N, Nishikawa S. A gradual neural-network approach for frequency assignment in satellite communication systems. IEEE Transactions on Neural Networks 1997;8(6):1359–70.
[18] Mizuike T, Ito Y. Optimization of frequency assignment. IEEE Transactions on Communications 1989;37(10): 1031–41.
[19] Huang YM, Chen RM. Scheduling multiprocessor job with resource and timing constraints using neural networks. IEEE Transactions on System, Man and Cybernetics Part B 1999;29(4):490–502.
[20] Hou ES, Ansari N, Ren H. A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems 1994;5(2):113–20.
[21] Goldberg DE. Genetic algorithm in search, optimization and machine learning. Reading, MA: Addison-Wesley; 1989.
[22] Shrivastava Y, Dasgupta S, Reddy SM. Guaranteed convergence in a class of Hopfield networks. IEEE Transactions on Neural Networks 1992;3:951–61.
[23] Funabiki N, Takefuji Y. A neural network parallel algorithm for channel assignment problems in cellular radio networks. IEEE Transactions on Vehicular Technology 1992;41(4):430–7.
[24] Ramanathan P, Krisna MS, Agrawal P, Kishore S. Dynamic resource allocation schemes during handoff for mobile multimedia wireless networks. IEEE Transactions on Selected Areas in Communications 1999;17(7):1270–83.
[25] Thakur S, Chang YW, Wong DF, Muthukrishnan S. Algorithms for an FPGA switch module and associated routing algorithm for high performance FPGA's. IEEE Transactions on Computer Aided Design 1997;16(1):22–5.
[26] Kaushik R. A bounded search algorithm for segmented channel routing for FPGA's and associated channel architecture issues. Computer Aided Design 1993;12(11):1695–705.
[27] Lau TL, Tsang EPK. Guided genetic algorithm and its application to radio link frequency assignment problems. Constraints 2001;6(4):373–98.
[30] Smith K, Palaniswami M, Krishnamoorthy M. Neural techniques for combinatorial optimization with applications. IEEE Transactions on Neural Networks 1998;9(6):1301–18.
[31] Smith KA, Potvin J-Y, Kwok T. Neural network models for combinatorial optimization: deterministic, stochastic and chaotic approaches. Control and Cybernetics 2002;31(2):183–216.