Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Discrete Mathematics 39 (2012) 67–74

Electronic Notes in DISCRETE MATHEMATICS

www.elsevier.com/locate/endm

A Variable Neighbourhood Search approach to the Cutwidth Minimization Problem

Eduardo G. Pardo [a,1] Nenad Mladenović [b,2]
Juan J. Pantrigo [a,3] Abraham Duarte [a,4]

[a] *Departamento de Ciencias de la Computación, Universidad Rey Juan Carlos, Móstoles, Madrid, Spain*

[b] *Department of Mathematics, SISCM, Brunel University, London, UK*

Abstract

The Cutwidth Minimization Problem, also known as the Minimum Cut Linear Arrangement consists of finding an arrangement of the vertices of a graph on a line, in such a way that the maximum number of edges between each pair of consecutive vertices is minimized. This problem has practical applications in VLSI Design, Network Migration and Graph Drawing, among others. In this paper we propose several heuristics based on the Variable Neighbourhood Search methodology to tackle the problem and we compare them with other state-of-the-art methods.

Keywords: Cutwidth Minimization Problem, Variable Neighbourhood Search.

[1] eduardo.pardo@urjc.es. Supported by: TIN2008-06890-C02-02,TIN2012-35632-C02-02

[2] nenad.mladenovic@brunel.ac.uk

[3] juanjose.pantrigo@urjc.es. Supported by: TIN2008-06890-C02-02, TIN2011-28151

[4] abraham.duarte@urjc.es. Supported by: TIN2009-07516, TIN2012-35632-C02-02

1571-0653/$ – see front matter
doi:10.1016/j.endm.2012.10.010

1 Introduction

The Cutwidth Minimization Problem (CMP) is a NP-Hard [5] classical min-max problem. It consist of finding an ordering of the vertices of a graph on a line, in such a way that the maximum number of edges between each pair of consecutive vertices is minimized. In other words, given a graph $G = (V, E)$ with $n = |V|$, a labeling f of its vertices is an assignation of the integers $\{1, 2, \ldots, n\}$ to each vertex in V, where each one receives a different label. The cutwidth of a vertex v with respect to f, denoted as $CW_f(v)$, is the number of edges $(u, w) \in E$ satisfying $f(u) \leq f(v) < f(w)$. Therefore,

$$CW_f(v) = |\{(u, w) \in E : f(u) \leq f(v) < f(w)\}|$$

On the basis of the definition of the cutwidth of a vertex, it is possible to define the cutwidth of the graph, $CW_f(G)$, as the maximum of all cutwidths among its vertices. Therefore, it can be expressed as follows:

$$CW_f(G) = \max_{v \in V} CW_f(v)$$

In the Figure 1.a it is possible to see an example of a graph G and, in Figure 1.b, an ordering f of the vertices of G. In this figure, it is also shown the cutwidth of each vertex with the corresponding value at the bottom, and the cutwidth of the graph, $CW_f(G) = 6$, represented as a bold line.

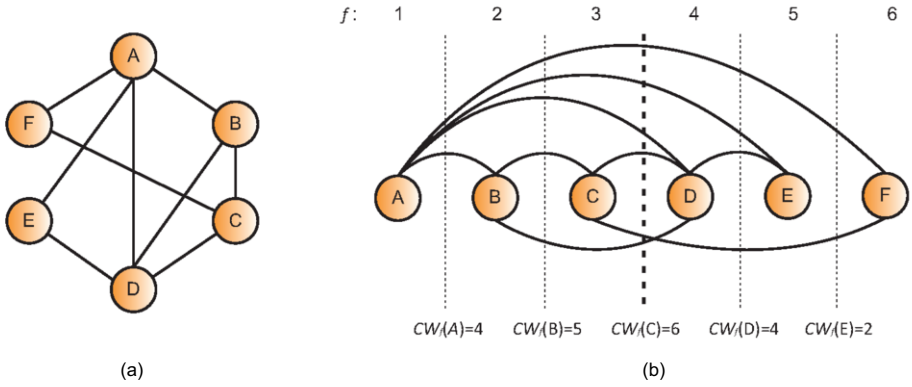

Fig. 1. (a) Graph G with six vertices and nine edges. (b) Ordering f of the vertices of the graph in (a) with the corresponding cutwidth of each vertex.

In this paper we propose to tackle the CMP using the Variable Neighbourhood Search (VNS) methodology. Particularly, we review the best previous heuristics for the problem and we adapt some of the best ideas to first develop a Reduced VNS (RVNS) and then a Basic VNS (BVNS).

2 State of the art

The CMP has also been referred in the literature using other names such as Minimum Cut Linear Arrangement [4], Network Migration Scheduling [14] or Board Permutation Problem [3]. The practical applications of the problem can be found in very different areas such as: Circuit Design [3], Network Reliability [8], Information Retrieval [2], Automatic Graph Drawing [12] or Protein Engineering [1]. Even it is possible to find a patent where the CMP is applied to Networks Migration [14].

There have been previous attempts to produce approximate and exact solutions to the CMP. Cohoon and Sahni were the first ones in proposing heuristic algorithms for the problem [3]. Particularly they tackled a generalization version of the CMP for hypergraphs known as the Board Permutation Problem. These authors proposed several constructive and local search procedures and embedded them in a Simulated Annealing metaheuristic. Later, Resende and Andrade [14] proposed a Greedy Randomized Adaptive Search heuristic paired with a Path Relinking. The most recent approach was carried out by Pantrigo et al. [13] who proposed an Scatter Search algorithm for the problem which outperformed previous results in the state of the art. As far as the exact approaches are concerned, most of them are centered in a particular type of graphs: Harper [7] studied the CMP for hypercubes; Thilikos et al. [16] proposed algorithms for special cases of trees; Rolim et al. [15] worked in the problem for Grids. Additionally, there are some general approaches for general graphs: Luttamagutzi et al. [9] proposed an Integer Linear Programming formulation for the CMP and Martí et al. [10] proposed a Branch & Bound algorithm for finding exact solutions to general graphs.

3 Variable Neighbourhood Search

The Variable Neighbourhood Search (VNS) is a metaheuristic proposed in [11] as a general framework to solve hard problems. It is based on a simple idea: systematical changes of neighbourhood structures within the search procedure. The original metaheuristic has been widely evolved with many extensions, highlighting Variable Neighbourhood Descent (VND), Reduced VNS (RVNS), Basic VNS (BVNS), Skewed VNS (SVNS), General VNS (GVNS), Variable Neighbourhood Decomposition Search (VNDS) and Reactive VNS. See [6] for a recent thorough review. In this paper we focus our attention in the Reduced VNS (RVNS) and in the Basic VNS (BVNS) schemas to tackle the CMP.

3.1 Initial solution

RVNS and BVNS algorithms start from an initial solution which can be constructed at random. However, a constructive procedure allows the method to perform faster. It is not our intention to propose a new constructive procedure for the CMP since there are many in the related literature. Specifically, in [3], they proposed three greedy algorithms to construct solutions for the problem. In [14], authors proposed a new one where they started with a randomized depth-first search ordering and then, the vertices were placed in the position where the increase in the objective function is minimized. Recently, Pantrigo et al. [13] proposed two new constructive procedures (named C1 and C2) based on the GRASP methodology and two additional variants (named C3 and C4) where the random and greedy stages of GRASP were permuted. We have selected the constructive named C1 by Pantrigo et al. to construct an initial solution for our methods. We refer the reader to [13] for further details.

3.2 Reduced Variable Neighbourhood Search

In Algorithm 1 we present the pseudocode of the RVNS schema. It receives three input arguments: an initial solution (f), the largest neighbourhood (k_{max}) and the maximum computing time (t_{max}). The procedure starts by performing a perturbation to the current solution using the function `Shake` (step 5) and obtaining a new solution f'. Then, in step 6, the `NeighbourhoodChange` procedure decides whether the RVNS needs to explore a larger neighbourhood by increasing k (f' is worse than f) or not, $k = 1$ (f' is better than f). Steps 5 and 6 are repeated until k_{max} is reached. This parameter determines the maximum number of different neighbourhoods to be tried in the current iteration when there is no improvement. Steps 3 to 8 are repeated until t_{max} is reached, starting in each iteration from the best found solution.

We propose two different versions of the RVNS schema that we call RVNS_1 and RVNS_2 respectively. Both algorithms start from a solution which is constructed with the procedure presented in Section 3.1. The main difference between RVNS_1 and RVNS_2 lies in the `Shake` procedure. Particularly, in RVNS_1 the shake procedure is based on interchange moves (two vertices are selected at random and they interchange their positions), while RVNS_2 is based on insertion moves (one vertex is selected at random and inserted in a random position). In both cases, the process is repeated k times over the solution f.

Algorithm 1 Pseudocode of RVNS

```
1: Procedure RVNS (f, k_max, t_max)
2: repeat
3:    k ← 1
4:    repeat
5:       f' ← Shake(f, k)
6:       NeighbourhoodChange(f, f', k)
7:    until k = k_max
8:    t ← CPUTime()
9: until t > t_max
10: end RVNS
```

3.3 Basic Variable Neighbourhood Search

The BVNS has a similar schema than the RVNS but it introduces a new step, a local search procedure. Specifically, after step 5 in Algorithm 1, the BVNS try to reach a local optimum by using an improvement strategy. The local search procedure used in this paper is based on insertion moves. A naive implementation would insert any vertex in the solution in any position. This could be very time consuming. However, it is possible to determine, for each vertex, which positions can produce a maximum improvement. To determine these positions, we define the median label of a vertex v, denoted as $\texttt{med}(v)$, as the numerical label which separates the higher half of the vertices adjacent to v in the ordering from the lower half. The best position for any vertex is the median label. Therefore, it is only necessary to perform one insertion when v has an even number of adjacent vertices and two insertions when it has an odd number. The procedure stops when no one of the vertices is able to produce an improvement. For the sake of diversification, the BVNS uses a `Shake` procedure based on interchange moves since the local search is based on insertion moves.

4 Experiments

In this section we present the computational experiments that we performed to compare the different VNS approaches proposed for the CMP. Additionally, we compare the best of them with other state-of-the-art methods. All the algorithms were implemented in Java 6 SE and the experiments were carried out on an Intel Core 2 Quad CPU with 6 GB of RAM. To evaluate the algorithms we have used two sets of instances previously reported by other authors in the evaluation of the CMP. Particularly, we used the sets of instances named

	Constructive	**RVNS$_1$**	**RVNS$_2$**	**BVNS**
Avg.	550.6	261.2	254.9	92.0
Dev. (%)	1122.5	504.1	556.1	59.8
CPUt (s)	30.0	30.1	30.1	30.7

Table 1
Comparison of different VNS approaches for the CMP.

"*Grid*" (81 instances) and "*Harwell-Boeing*" (HB) (87 instances) proposed by [13]. All the instances are available at `http://www.optsicom.es/cutwidth`.

4.1 Comparison of the different VNS aproaches

In order to compare the proposed VNS algorithms we selected a representative 10% of the instances of each data set (16 instances) and we ran each algorithm during 30 seconds per instance. In Table 1 we report the average of the objective function (Avg.), the deviation to best known (Dev.) and the CPU time (CPUt (s)) of each algorithm. All the algorithms started from the same random initial solution. We also report a column with the best solution found by a random constructive after constructing solutions for 30 seconds. It is easy to see that all the methods were able to improve the solution obtained by the random constructive. Particularly, the BVNS method was the best one with an average value of the objective function of 92.00 and a deviation of 59.80%. We will therefore use this method for future comparisons.

4.2 Comparison with other state-of-the-art methods

Once the best VNS approach (BVNS) is selected among the proposed ones, we ran it over the whole data sets and we compared the results with previous attempts to derive approximate solutions in the state of the art. Specifically, the compared procedures were: Simulated Annealing (SA) [3]; Greedy Randomized Adaptive Search Procedure with Path Relinking (GPR) [14]; Scatter Search (SS) [13] and the new BVNS (BVNS). In Table 2 we present the results over the "*Grid*" and "*HB*" data sets. The initial solution for the BVNS was selected among 100 solutions constructed with the algorithm introduced in Section 3.1. The parameters k_{max} and t_{max} of the BVNS were dynamically assigned for each instance. Particularly they were set to 20% of the vertices of the considered instance. In the case of t_{max} this value was set in seconds. Both values were empirically selected. Results presented in Table 2 confirm

	Grid (81)				HB (87)			
	GPR	**SA**	**SS**	**BVNS**	**GPR**	**SA**	**SS**	**BVNS**
Avg.	38.4	16.1	13.0	12.9	364.8	346.2	315.2	314.4
Dev. (%)	201.8	25.4	7.8	7.7	95.1	53.3	3.4	2.2
#Best	2	37	44	48	2	8	47	59
CPU t (s)	235.2	216.1	210.1	45.2	557.5	435.4	430.6	64.3

Table 2
Comparison with the state-of-the-art algorithms.

that the new procedure outperforms previous attempts in the state-of-the-art. In both sets of instances the BVNS was able to reach a higher number of best values and a lower deviation in less computing time. Additionally, we corroborated that the differences among the algorithms are statistically significant by applying the Friedman test to the best solutions obtained by each method.

5 Conclusions and future work

In this paper we have compared the application of different variants of the Variable Neighbourhood Search methodology to the Cutwidth Minimization Problem. Particularly, we have adapted previous heuristics for the problem to the RVNS and BVNS variants. The comparisons performed indicates that the BVNS is the most suitable method and it is able to outperform previous algorithms in the state of the art. As a future work it would be interesting to extend this comparison to other VNS variants such as GVNS or VNDS.

References

[1] Blin, G., G. Fertin, D. Hermelin and S. Vialette, *Fixed-parameter algorithms for protein similarity search under mRNA structure constraints*, Journal of Discrete Algorithms **6** (2008), pp. 618–626.

[2] Botafogo, R. A., *Cluster analysis for hypertext systems*, in: *16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, 1993, pp. 116–125.

[3] Cohoon, J. and S. Sahni, *Heuristics for backplane ordering*, Journal of VLSI and Computer Systems **2** (1987), pp. 37–61.

[4] Díaz, J., A. Gibbons, G. E. Pantziou, M. J. Serna, P. G. Spirakis and J. Toran, *Parallel algorithms for the minimum cut and the minimum length tree layout problems*, Theoretical Computer Science **181** (1997), pp. 267–287.

[5] Gavril, F., *Some NP-complete problems on graphs*, in: *Proceedings of the 11th Conference on Information Sciences and Systems*, 1977, pp. 91–95.

[6] Hansen, P., N. Mladenović and J. A. Moreno Pérez, *Variable neighbourhood search: algorithms and applications*, Ann. Oper. Res. **175** (2008), pp. 367–407.

[7] Harper, L. H., *Optimal numberings and isoperimetric problems on graphs*, Journal of Combinatorial Theory (1966), pp. 385–393.

[8] Karger, D., *A randomized fully Polynomial Time Approximation Scheme for the all-terminal network reliability problem*, SIAM Journal on Computing **29** (1999), pp. 492–514.

[9] Luttamaguzi, J., M. Pelsmajer, Z. Shen and B. Yang, *Integer programming solutions for several optimization problems in graph theory*, Technical report, DIMACS (2005).

[10] Martí, R., J. J. Pantrigo, A. Duarte and E. G. Pardo, *Branch and bound for the Cutwidth Minimization Problem*, Computers & Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.05.016.
URL http://dx.doi.org/10.1016/j.cor.2012.05.016

[11] Mladenović, N. and P. Hansen, *Variable Neighborhood Search*, Computers & Operations Research **24** (1997), pp. 1097–1100.

[12] Mutzel, P., *A polyhedral approach to planar augmentation and related problems*, in: *ESA'95: Proceedings of the 3rd Annual European Symposium on Algorithms*, 1995, pp. 494–507.

[13] Pantrigo, J. J., R. Martí, A. Duarte and E. G. Pardo, *Scatter search for the cutwidth minimization problem*, Annals of Operations Research (2011), dOI:10.1007/s10479-011-0907-2.
URL http://dx.doi.org/10.1007/s10479-011-0907-2

[14] Resende, M. G. C. and D. V. Andrade, *Method and system for Network Migration Scheduling.*, U.S. Patent Application. US2009/0168665 (2009).

[15] Rolim, J., O. Sýkora and I. Vrt'o, *Optimal cutwidths and bisection widths of 2- and 3-dimensional meshes*, in: *Graph-Theoretic Concepts in Computer Science*, Lecture Notes in Computer Science **1017**, 1995 pp. 252–264.

[16] Thilikos, D. M., M. J. Serna and H. L. Bodlaender, *Cutwidth II: Algorithms for partial w-trees of bounded degree,*, J. of Algorithms **56** (2005), pp. 25–49.