Operations Research Letters 14 (1993) 133–137
North-Holland

October 1993

The noising method: a new method for combinatorial optimization

Irène Charon and Olivier Hudry

Ecole Nationale Supérieure des Télécommunications, Paris, France

Received March 1992
Revised June 1993

This paper presents the principles and the first results of a new combinatorial optimization method that we call the noising method. It is applied to the NP-hard problem of the clique partitioning of a graph. The results obtained from this problem which are definitely better than those provided by standard iterative-improvement methods and compare favorably with those of simulated annealing, show the relevance of the noising method.

heuristics; combinatorial optimization; clique partitioning of a graph; NP-hard problems

1. Introduction

This paper deals with combinatorial optimization problems of the form $\min\{f(s): s \in S\}$. The elements of S are called *solutions*, and f is called the *evaluation function*. If the size of S is not small and if we do not know fast algorithms to solve the problem exactly (as is the case with NP-hard problems, see Garey and Johnson [5]), then it may be necessary to use heuristics – that is, methods which provide approximate solutions near the optimum. The aim of this paper is to present a new heuristic that we call the *noising method*, and the first results obtained from its application to the problem of clique partitioning.

2. The principles of the noising method

Many heuristics for combinatorial optimization problems use 'elementary transformations'. A transformation is any operation that transforms a solution s of S into a solution s' of S. Elementary transformations constitute a subset of the set of transformations: an elementary transformation consists generally in changing one feature of s without changing its global structure; it defines the neighborhood $N(s)$ of a solution s as the set of all the solutions s' that can be obtained from s by means of the elementary transformation.

Such a transformation makes possible the definition of an iterative-improvement method. From a current solution s, we take a solution s' in the neighborhood $N(s)$ of s: if $f(s') < f(s)$, then s' becomes the current solution, otherwise we keep s. The process is then reiterated with the current solution. When there is no s' in $N(s)$ better than the current solution s, we have reached a local minimum with respect to this neighborhood (that is, with respect to this elementary transformation), and the process is over. This type of heuristic, which consists in applying iterative-improvements until a local minimum is reached is called *descent*.

Correspondence to: Prof. Irène Charon, ENST, 46 rue Barrault, 75634 Paris Cedex 13, France.

The noising method is based on descent. We start with an initial solution and we repeat the following operations:

- add noise to the data in order to change the values taken by the function f;
- apply a descent to the current solution for the noised data.

At each iteration, the amount of the added noise decreases until it reaches 0 at the last iteration. The final solution is the best solution computed during the process.

The noising method was applied to the clique partitioning problem and compared with simulated annealing, one of the most efficient technics for many combinatorial problems. For both, it possible to choose an approximate *a priori* CPU time; the greater the CPU time is, the better the average results are. In general, the noising method is preferable to simulated annealing for the instances that we tried and for the chosen time intervals (as demonstrated below).

3. The clique partitioning problem

Let G be a complete non-oriented graph (a 'clique') on n vertices. With each edge $\{x, y\}$, a (negative or positive real) weight $G(x, y)$ is associated. The problem consists in partitioning the vertex set V of G into p cliques $V_1, V_2, \ldots, V_p$, where p is not fixed, so that we may minimize the sum of the weights of edges with their two extremities in the same clique V_k $(k = 1, 2, \ldots, p)$. This problem is NP-hard (see Wakabayashi [9]), and remains so even if the weights of G are only -1 or $+1$ (see Krivanek and Moravek [6]).

Hence a solution s of the clique partitioning problem is a partition of V into $V_1, V_2, \ldots, V_p$, and the evaluation function f is then defined by:

$$f(s, G) = \sum_{k=1}^{p} \sum_{x,\, y \in V_k} G(x, y).$$

From a current solution, the elementary transformation (Régnier [8]) consists in removing a vertex from one of the cliques V_k and putting it into another clique (possibly empty if we want to create a new one).

4. Application of the noising method to the clique partitioning problem

To add noise to the data, we use the following technique. Let MAX_VAL denote the greatest absolute value of the weights of the edges of G. At each step of the noising method, a 'perturbation rate', here called RATE, is chosen (RATE is not necessarily less than 1); then, for each edge $\{x, y\}$, a random number r is uniformly drawn between -1 and $+1$, and $G'(x, y) = G(x, y) + r \times \text{RATE} \times \text{MAX\_VAL}$ takes the place of $G(x, y)$. The coefficient RATE decreases during the running between two extreme values RATE_MAX and RATE_MIN.

We call NOISED(RATE) the result G' that is obtained by this operation; and we call DESCENT(s, G') the solution found by applying a descent to s with the noised data, given by G'.

Below, we present the main algorithm that we tried. We grouped noising iterations into what we call 'cycles'; at the end of each cycle, we started a new one with the best solution found since the beginning of the whole algorithm. Moreover, we successively alternated a descent for the noised data (G') and a descent for the original data (G).

Noising method

```
Parameters
  RATE_MAX, RATE_MIN: reals
  NB_ITER_PER_CYCLE, NB_CYCLES: integers
Variables
  BEST_SOLUTION, CURRENT_SOLUTION: of type 'solution'
  BEST_EVALUATION, EVALUATION, RATE, STEP: real
  CYCLE_METER, ITER_METER: integers
Begin
  Draw the initial CURRENT_SOLUTION randomly
  STEP := (RATE_MAX - RATE_MIN)/(NB_ITER_PER_CYCLE × NB_CYCLES - 1)
  RATE := RATE_MAX
  BEST_EVALUATION := f(CURRENT_SOLUTION, G)
  for CYCLE_METER in 1 to NB_CYCLES do:
    for ITER_METER in 1 to NB_ITER_PER_CYCLE do:
      G' := NOISED(RATE)
      CURRENT_SOLUTION := DESCENT(CURRENT_SOLUTION, G')
      CURRENT_SOLUTION := DESCENT(CURRENT_SOLUTION, G)
      EVALUATION := f(CURRENT_SOLUTION, G)
      if EVALUATION < BEST_EVALUATION then
        BEST_EVALUATION := EVALUATION
        BEST_SOLUTION := CURRENT_SOLUTION
      end if
      RATE := RATE - STEP
    end for
    CURRENT_SOLUTION := BEST_SOLUTION
  end for
  return BEST_SOLUTION
end.
```

5. Results

We present the results obtained with the algorithm, compared with those of simulated annealing and of a method that we call *descents*. The descents-method has an integer parameter q and produces the best solution obtained by the application of q successive descents-that is, by the repetition of q times of the following operations: the drawing of a random initial solution and the applying of a descent to this initial solution.

The adopted simulated annealing follows a usual pattern (see de Amorim, Barthélemy and Ribeiro [1], Bonomi and Lutton [4] and van Laarhoven and Aarts [7]): the initial temperature is computed by the method proposed by Aragon, Johnson, McGeoch and Schevon [2] with an initial acceptance ratio of 50%, the decrease of temperature is geometric with a ratio of 0.925, the number of proposed elementary transformations is proportional to n^2 (where n is the number of vertices), and the number of temperature changes is fixed. One of the main advantages of such a model is that it is possible, more or less, to foresee the necessary amount of CPU time.

As there is no systematic way of choosing the parameters of simulated annealing (as is shown by the very different values found in the literature) nor for the noising method, we tried several sets of values for these parameters. The best parameters we found are given in Tables 1, 2 and 3 between brackets; they correspond with RATE_MAX, RATE_MIN, NB_ITER_PER_CYCLE and NB_CYCLES for the noising method; with the number of temperature changes and the coefficient of proportionality (this

Table 1
Example 1: graph on 150 vertices, with weights randomly chosen in $\{-1, 1\}$

Method	Time	Average-eval	Best-eval	Worst-eval
Descents (1)	28	−824	−874	−762
Descents (10)	289	−869	−895	−836
Noising (0.9, 0.5, 15, 10)	326	−932	−939 (2)	−919
Simulated annealing (15, 3)	331	−927	−938	−917
Noising (0.95, 0.5, 25, 10)	530	−935	−939 (4)	−929
Simulated annealing (15, 5)	551	−930	−939 (2)	−920

Table 2
Example 2: graph on 99 vertices, with weights randomly chosen between −100 and 100

Method	Time	Average-eval	Best-eval	Worst-eval
Descents (1)	9	−16125	−17874	−14276
Descents (15)	130	−17551	−17938	−16780
Noising (0.8, 0.4, 15, 8)	108	−18457	−18528 (6)	−18249
Simulated annealing (12, 4)	115	−18361	−18528 (6)	−18090
Noising (0.8, 0.5, 25, 15)	342	−18507	−18528 (13)	−18442
Simulated annealing (12, 12)	352	−18463	−18528 (10)	−18221

Table 3
Example 3: graph on 150 vertices, with weights randomly chosen between −1000 and 1000

Method	Time	Average-eval	Best-eval	Worst-eval
Descents (1)	33	−376447	−398028	−353560
Descents (4)	143	−395973	−414132	−374440
Noising (0.8, 0.15, 10, 5)	157	−428594	−433908	−419876
Simulated annealing (11, 2)	165	−425800	−433908	−415285
Noising (0.8, 0.15, 15, 15)	575	−433046	−433908	−428302
Simulated annealing (11, 7)	580	−430458	−433908	−419315
Noising (0.85, 0.2, 35, 15)	1395	−433881	−434918 (1)	−433491
Simulated annealing (12, 15)	1361	−432084	−433908	−421472

coefficient times n^2 equals the number of proposed elementary transformations) for simulated annealing; with the number q of descents for the descents-method.

We focused our attention on three graphs. The results shown below have always been derived from 20 trials. The time (denoted by 'Time') is the average CPU time, in seconds, obtained on a SUN workstation; the program is written in Ada. 'Average-eval' is the average of the 20 evaluations obtained during the 20 trials, while 'Best-eval' is the best and 'Worst-eval' the worst of them. In the column of Best-eval, we put between parentheses the number of times that the best evaluation is encountered, when it is presumed to be the global minimum.

From these tables, we can note that the best results provided by the noising method are always at least as good as those of simulated annealing and attained more often. In addition, the noising method is much more reliable, as shown by 'Worst-eval', and thus the average values obtained with our method are better.

During other trials, we observed that the noising method is almost as efficient with many other choices for the parameters; the choice of 0.8 for RATE_MAX is always good; for RATE_MIN, it is possible to choose it equal to 0.4. NB_ITER_PER_CYCLE and NB_CYCLES must be chosen according to the CPU-time desired; we would suggest choosing NB_ITER_PER_CYCLE between NB_CYCLES and twice NB_CYCLES.

6. Other applications

More recently, we applied the noising method to two other problems: the traveling salesman problem and one arising in voting theory (median orders of tournaments [3]). For the first, the noising method was particularly efficient: we observed that simulated annealing required between 6 and 14 times more CPU-time to get the same solution quality. For the second problem, this ratio was about 4. These results will be the subject of future publications.

References

[1] S.G. de Amorim, J.-P. Barthélemy and C.C. Ribeiro, "Clustering and clique partitioning: simulated annealing and tabu search approaches", *J. Classification* **9**, 17–62 (1992).

[2] C.R. Aragon, D.S. Johnson, L.A. McGeoch and C. Schevon, "Optimization by simulated annealing: an experimental evaluation", Workshop on Statistical Physics in Engineering and Biology, Yorktown Heights, 1984.

[3] J.-P. Barthélemy and B. Monjardet, "The median procedure in cluster analysis and social choice theory", *Mathematical Social Sciences* **1**, 235–267 (1981).

[4] E. Bonomi and J.-L. Lutton, "The *N*-city travelling salesman problem: statistical mechanics and the Metropolis algorithm", *SIAM Rev.* **26**, 551–568 (1984).

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability, a Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[6] M. Krivanek and J. Moravek, "NP-hard problems in hierarchical-tree clustering", *Acta Informatica* **23**, 311–323 (1986).

[7] P.J.M. van Laarhoven and E.H.L. Aarts, *Simulated Annealing: Theory and Applications*, Kluwer, Dordrecht, 1987.

[8] S. Régnier, "Sur quelques aspects mathématiques des problèmes de classification automatique", *I.C.C. Bulletin* **4**, 85–111 (1965).

[9] Y. Wakabayashi, "Aggregation of binary relations: algorithmic and polyhedral investigations", PhD. Thesis, University of Augsbourg, 1986.