Descent with mutations metaheuristic

Irène Charon and Olivier Hudry

[1]*Department of Computer Science, École nationale supérieure des télécommunications, 46, rue Barrault, 75634 Paris Cedex 13, France,* {`charon,hudry`}`@enst.fr`

Keywords: Aggregation of symmetric relations into equivalence relations, clique partitioning of a graph, combinatorial optimization, metaheuristics, noising methods, simulated annealing.

Abstract. We study here the application of a metaheuristic, issued from the noising methods and that we call “descent with mutations”, to a problem arising in the field of the aggregation of symmetric relations: the clique partitioning of a weighted graph. This local search metaheuristic, of which the design is very simple, is compared with another very efficient metaheuristic, which is a simulated annealing improved by the addition of some ingredients coming from the noising methods. The experiments show that the descent with mutations is at least as efficient for the studied problem as this improved simulated annealing, usually a little better, while, above all, it is much easier to design and to apply.

1 Introduction

Since 1993, when we published the first paper on the noising methods, many variants of them have been tried (see [4], [7-8]). In this paper, we deal with a metaheuristic that we may consider as a variant of the noising methods too: the descent with mutations (DWM) [9]. This method looks like the usual descent, but with random elementary transformations which are performed, from time to time, in a blind way, in the sense that they are accepted whatever their effects on the function f to optimize (such an elementary transformation performed without respect to its effect on f will be called a mutation in the sequel). The density of performed mutations decreases during the process, so that the process at its end is the same as a classic descent.

To experiment this new variant, we apply it to two problems: Régnier’s problem [12] and Zahn’s problem [15]. These problems arise in the field of the aggregation and the approximation of symmetric relations. They are modelled by a problem arising in graph theory as a clique partitioning of a weighted graph (though it is not the classic problem usually called “clique partitioning problem”, see below). We intended first to compare it with a pure simulated annealing, but our experiments led us to conclude that such a method is poorly efficient with respect to DWM on the considered problems (notice that de Amorim et alii [1] compared the applications of tabu search and of simulated annealing to this problem: their experiments show that the results provided by these two metaheuristics are qualitatively quite similar: sometimes the first is better, sometimes the second). So, we preferred to compare DWM with a hybridized method with the probability of accepting an elementary transformation issued from simulated annealing but with other improving ingredients coming from the noising methods (as we did in [5] and in [6] for the Travelling Salesman Problem).

2 Principle of the descent with mutations

As the other metaheuristics, DWM is not designed to be applicable to only one combinatorial problem, but to many of them. Such a problem can be stated as follows:

$$\text{Minimize } f(s) \text{ for } s \in S,$$

where S is a finite set and f is a function defined on S; the elements s of S will be called solutions. As many other heuristics, DWM is based on elementary transformations. A transformation is any operation changing a solution into another solution. A transformation will be considered as elementary (or local) if, when applied to a solution s, it changes one feature of s without modifying its global structure much. For instance, if s is a binary string, a possible elementary transformation would be to change one bit of s into its complement.

Thanks to the elementary transformations, we may define the neighbourhood $N(s)$ of a solution s: $N(s)$ is the set of all the solutions that we can get from s by applying an elementary transformation to s. Then, we may define an iterative improvement method, or descent for a minimization problem (it is the case for the problems considered here), as follows. A descent starts with an initial solution s_0 (which can be for instance randomly computed, or found by a heuristic) and then generates a series of solution s_1, s_2, ..., s_i, ..., s_q such that:

1. for any $i \geq 1$, s_i is a neighbour of s_{i-1}: $s_i \in N(s_{i-1})$;
2. for any $i \geq 1$, s_i is better than s_{i-1} with respect to f: $f(s_i) < f(s_{i-1})$;
3. no neighbour of s_q is better than s_q: $\forall s \in N(s_q)$, $f(s) \geq f(s_q)$.

Then s_q is the solution returned by the descent. The descent is over and the final solution s_q provided by the descent is (at least) a local minimum of f with respect to the adopted elementary transformation. The whole method may stop here, or restarts a new descent from a new initial solution (to get repeated descents).

In such a descent, the process is not blind in the sense that the elementary transformations are adopted only if they fulfil the following acceptance criterion: they must involve an improvement of the value taken by f. In DWM, we also apply the basic process of a descent but, from time to time, instead of applying the previous acceptance criterion, we apply and accept the considered elementary transformation, whatever its effect on f: we say that we have a blind elementary transformation, or simply a mutation (see Figure 1, which describes DWM in general). Thus, the only thing to specify in order to apply DWM (in addition to what must be defined to apply a descent, like the elementary transformation) is when a mutation is adopted. It is what we show in the next section, for the problems studied in this paper.

- Repeat:
  - with a certain probability, apply an arbitrary elementary transformation (irrespective improvement or worsening: this is a mutation)
  - otherwise, apply an elementary transformation which brings an improvement
- until a given condition is fulfilled.

Figure 1. General description of DWM.

We said at the beginning that DWM can be considered as a variant of the noising methods. Remember that the most general scheme of the noising methods (see [7]) consists in computing a noised variation $\Delta f_{noised}(s, s')$ of f when a neighbour s' of the current solution s is considered: $\Delta f_{noised}(s, s') = f(s') - f(s) + \rho$, where ρ is a random number depending on different things (like s, s', the iteration number, the scheme of the noising method, the adopted probability law, and so on); then the acceptance criterion becomes the following: the transformation of s into s' is accepted if $\Delta f_{noised}(s, s')$ is negative. So, indeed, it is not difficult to design the characteristics of the law followed by ρ in order to show that DWM constitutes a special case of the noising methods: it is sufficient to choose a very negative value for ρ (that is, a negative value with a great absolute value) when we decide to perform a mutation, or 0 otherwise; details are left to the interested reader.

3 The clique partitioning problem and the descent with mutations

3.1 Aggregation and approximation of symmetric relations; the clique partitioning problem

The problems that we consider deal with the aggregation and the approximation of symmetric relations. The first one, set by S. Régnier [12], consists in the aggregation of equivalence relations into a unique equivalence relation summarizing the initial equivalence relations as accurately as possible; the second one, set by C.T. Zahn [15], consists in the approximation of a symmetric relation by an equivalence relation at minimum distance.

Both can be modelled by a clique partitioning problem.

Régnier's problem arises for example in cluster analysis. In this problem, we consider a set X of n objects and a set of m criteria; each criterion is assumed to define an equivalence relation on X; the aim is to find a unique equivalence relation defined on X which summarizes the m criteria as accurately as possible by minimizing the total number of disagreements with respect to the given m criteria. This problem is NP-hard when m is not fixed [2] (its complexity is unknown in the general case when m is fixed).

Zahn's problem, which arises in social sciences, consists in approximating a given symmetric relation R defined on a set X by an equivalence relation E defined also on X which is at minimum distance from R with respect to the symmetric difference distance (which measures the number of disagreements between R and E, see [3]). This problem is NP-hard too, as shown by M. Krivanek and J. Moravek [11].

These two problems can be represented (see for instance [13]) by the following clique partitioning problem (CPP in what follows). In this CPP, we consider a weighted undirected graph $G = (X, U, w)$ with n vertices; G is complete; an integer (which can be positive, or negative, or equal to 0) $w(x, y) = w(y, x)$ is associated with each edge $\{x, y\} \in U(x \neq y)$; our CPP consists in finding a partition of X into $k(G)$ disjoint cliques C_1, C_2 , ..., $C_{k(G)}$ (hence the number $k(G)$ of cliques is not fixed a priori and depends on G, or more precisely on n and w; for this reason and because of the signs of the weights, CPP is not the famous clique partitioning problem sometimes known as the k-cut problem, though the formulations of the two problems are near each other) in order to minimize the sum of the weights of the edges with their two extremities in a same clique, i.e., in order to minimize the function f defined for any partition $(C_1, C_2, ..., C_k)$ of X by:

$$f(C_1, C_2, ..., C_k) = \frac{1}{2}\sum_{i=1}^{k} \sum_{(x,y)\in C_i^2, x\neq y} w(x, y).$$

This CPP is also NP-hard [13-14]. To formulate Régnier's problem and Zahn's problem as instances of CPP, we build a weighted complete graph as follows. The vertex set will be the set X of Régnier's or Zahn's problems. For Régnier's problem, the weight of an edge $\{x, y\}$ (with $x \neq y$) is given by the difference $m - 2m_{xy}$, where m_{xy} denotes the number of equivalence relations for which x and y are together in a same class (this weight is also the difference between the number, equal to $m - m_{xy}$, of equivalence relations for which x and y are not together in a same class, and m_{xy}). For Zahn's problem, let R be the symmetric relation of the instance; the weight of an edge $\{x, y\}$ (with $x \neq y$) is -1 if x and y are in relation with respect to R and +1 otherwise. Then the search of an equivalence relation which is a solution of Régnier's or Zahn's problems consists in both cases in partitioning the weighted complete graph into disjoint cliques in order to minimize the sum of the weights of the edges with their two extremities in a same clique; hence our clique partitioning problem.

Conversely, it is trivial to associate an instance of Zahn's problem (a symmetric relation) with any instance of CPP in which all the weights belong to $\{-1, 1\}$. On the other hand, B. Debord [10] showed that it is possible to associate an instance of Régnier's problem (a set of equivalence relations) with any instance of CPP if all the edge-weights have the same parity. As it is always possible to double the weights $w(x, y)$ of the edges $\{x, y\}$ without changing the structure of an optimal solution, it appears that it is equivalent to solve Régnier's problem or to solve CPP. And similarly, it is equivalent to solve Zahn's problem or the instances of CPP in which all the weights belong to $\{-1, 1\}$. In the following, we refer to Régnier's problem for instances of CPP with any integer weights, and to Zahn's problem for instances of CPP in which all the weights belong to $\{-1, 1\}$.

3.2 Application of the descent (without mutations) to the clique partitioning problem

To define a descent for CPP, we apply the following elementary transformation (defined by S. Régnier [12]): we choose a vertex v and we move v from its current clique into another clique or alone in a new clique; in this last case, we say that we put v in the empty clique. To apply a descent, we begin from a randomly chosen partition and we consider the vertices one after the other in a cyclic way: the neighbours are ranked in a certain order (which is not necessarily always the same) and they are all considered in this order once, before being considered for a second time; a neighbour better than the current solution is accepted as soon as it has been discovered. More precisely, for each vertex v and for each clique C (including the empty one and the current clique of v) of the current solution s, we compute the variation $\Delta f(s, C)$ of f when v is moved from its current clique to C; the clique $C*$ for which the variation $\Delta f(s, C*)$ is minimum (since the current clique of v is taken into account, we get $\Delta f(s, C*) \leq 0$, with an equality if $C*$ is the current clique of v) is called the best clique for v (with respect to s); then, if moving v from its current clique to its best clique (with respect to s) involves a strict improvement (that is, if $\Delta f(s, C*) < 0$, we do move v from its current clique to its best clique (with respect to s) and thus we get a new solution s' from which we apply the same process, starting from the vertex following v in the prescribed order; otherwise, we keep v in its current clique and we consider the next vertex. When all the vertices are successively considered and that there is no vertex that can be moved towards a clique better than its current clique, the descent is over.

3.3 Application of the descent with mutations to the clique partitioning problem

In DWM, we apply almost a descent but, from time to time, instead of moving the considered vertex to its best clique, we move it to a clique chosen randomly. More precisely, when a vertex v is considered, we have two possibilities: with a probability p, we choose a clique randomly, with a uniform probability on the cliques (including the empty one) of the current solution; or, with a probability $1 - p$, we compute the best clique for v; in both cases, we move v to the chosen clique.

The method is utterly described by Figure 2, in which the vertices are assumed to be 1, 2, ..., n. It requires to choose only two parameters: a real number, called $initialRate$, between 0 and 1 and an integer called $totNbCycles$, which gives the total number of performed cycles. We call cycle the operations that must be performed in order to compute the best clique for each vertex of the graph (then $totNbCycles$ is linked to the total number of applied elementary transformations that are performed during the method, but the explicit relation is not simple because the number of disjoint cliques is not fixed and so may change during the process; it involves that $totNbCycles$ is directly related to the CPU time that the user wishes to spend to solve his or her problem: the higher $totNbCycles$, the longer the method). The parameter $initialRate$ gives the probability to perform a mutation at the beginning of the whole process.

```
• Choose a partition randomly: it is the current partition $P$;
• $bestPartition$ = $P$;
• $numCycle$ = 0;
• while $numCycle < totNbCycles$, do:
    – choose a real number $r$ between 0 and 1 randomly, with a uniform probability;
    – $p = r \times initialRate \times \frac{totNbCycles-numCycle}{totNbCycles}$;
    – $v$ = 1;
    – while $v \leq n$, do
        ∗ choose a real number $q$ between 0 and 1 randomly, with a uniform probability;
        ∗ if $q < p$, then choose a clique $C*$ (which can be the empty clique) in $P$ randomly, with a uniform probability;
        ∗ else compute the best (with respect to $P$) clique $C*$ for $v$;
        ∗ update $P$ by moving $v$ to $C*$;
        ∗ if necessary, update $bestPartition$;
        ∗ $v = v + 1$;
    – $numCycle = numCycle + 1$;
• apply a descent to $P$;
• if necessary, update $bestPartition$;
• return $bestPartition$ and $f(bestPartition)$.
```

Figure 2. Scheme of DWM for the clique partitioning problem.

We can see in Figure 2 that the probability $initialRate \times \frac{totNbCycles-numCycle}{totNbCycles}$ of performing a mutation is computed at the beginning of each cycle and decreases arithmetically from one cycle to the next one, down to 0 (it is also possible to apply a geometrical decrease, as in simulated annealing, but then not down to 0). To improve the performances reached by the method, we multiply this probability of applying mutations in a given cycle by a real number (r in Figure 2) chosen randomly between 0 and 1. Of course, we keep the best solution obtained during the process in memory. To be sure to get at least a local minimum, we complete the process with a descent.

4 Experiments

Experiments are not detailed here. They show that DWM may provide good results, with about the same quality than the ones got by an improved and tuned version of simulated annealing, within the same CPU time (or less), while, above all, it is very easy to design and to apply DWM to problems like CPP, and usually easier to tune than SA.

Indeed, the main advantage of DWM with respect to standard metaheuristics like SA is that, aside the CPU time, there is only one parameter to tune: $initialRate$. The sensibility analysis shows that the tuning of $initialRate$ is not a crucial point if the value of $initialRate$ is not too low. It is even possible to choose $initialRate = 1$, so that there is no parameter to tune !

Because of its simplicity (to design and then to tune), we hope that DWM deserves interest.

5 References

[1] S.G. de Amorim, J.-P. Barthélemy and C.C. Ribeiro, Clustering and clique partitioning: simulated annealing and tabu search approaches, Journal of Classification 9, 1992, 17-42.
[2] J.-P. Barthélemy and B. Leclerc, The median procedure for partitions, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 19, 1995, 3-34.
[3] J.-P. Barthélemy and B. Monjardet, The median procedure in cluster analysis and social choice theory, Mathematical Social Sciences 1, 1981, 235-267.
[4] I. Charon and O. Hudry, The noising method: a new combinatorial optimization method, Operations Research Letters 14, 1993, 133-137.
[5] I. Charon and O. Hudry, Mixing different components of metaheuristics, in: I.H. Osman and J.P. Kelly (eds), Metaheuristics: Theory and Applications, Kluwer Academic Publishers, Boston, 1996, 589-603.
[6] I. Charon and O. Hudry, Application of the noising methods to the Travelling Salesman Problem, European Journal of Operational Research 125 (2), 2000, 266-277.
[7] I. Charon and O. Hudry, The noising methods: a generalization of some metaheuristics, European Journal of Operational Research 135 (1), 2001, 86-101.
[8] I. Charon and O. Hudry, Noising methods for a clique partitioning problem, Discrete Applied Mathematics 154 (5), 2006, 754-769.
[9] I. Charon and O. Hudry, Application of the "descent with mutations" metaheuristic to a clique partitioning problem, Proceedings of the 2007 IEEE International Conference on Research, Innovation and Vision for the Future, Proceedings of the Institute of Electrical and Electronics Engineers, Vietnam, 2007, 29-35.
[10] B. Debord, Axiomatisation de procédures d'agrégation de préférences, PhD thesis, Grenoble, 1987.
[11] M. Krivanek and J. Moravek, NP-hard problems in hierarchical-tree clustering, Acta Informatica 23, 1986, 311-323.
[12] S. Régnier, Sur quelques aspects mathématiques des problèmes de classification automatique, I.C.C. Bulletin 4, Rome, 1965.
[13] Y. Wakabayashi, Aggregation of binary relations: algorithmic and polyhedral investigations, PhD thesis, Augsbourg, 1986.
[14] Y. Wakabayashi, The Complexity of Computing Medians of Relations. Resenhas 3 (3), 1998, 323-349.
[15] C.T. Zahn, Approximating symmetric relations by equivalence relations, SIAM Journal on Applied Mathematics 12, 1964, 840-847.