On Complexity of Optimal Recombination for Binary Representations of Solutions

Anton V. Eremeev * eremeev@ofim.oscsbras.ru
Laboratory of Discrete Optimization, Omsk Branch of Sobolev Institute of Mathematics
13 Pevtsov str., Omsk, 644099, Russia

Abstract

We consider the optimization problem of finding the best possible offspring as a result of a recombination operator in an evolutionary algorithm, given two parent solutions. The optimal recombination is studied in the case, where a vector of binary variables is used as a solution encoding. By means of efficient reductions of the optimal recombination problems (ORPs) we show the polynomial solvability of the ORPs for the maximum weight set packing problem, the minimum weight set partition problem and the linear Boolean programming problems with at most two variables per inequality and some other problems. Also we identify several NP-hard cases of optimal recombination: the Boolean linear programming problems with three variables per inequality, the knapsack, the set covering, the p-median and some other problems.

Keywords

Complexity, Evolutionary Algorithm, Optimal Recombination, Optimized Crossover

1 Introduction

An important task in design of an evolutionary algorithm for solving an optimization problem is to develop a recombination operator that efficiently combines the given *parent* solutions, producing "good" *offspring* solutions (see e.g. Jansen and Wegener (2002)). In this paper, we study the optimization problem of finding the best possible offspring as a result of a recombination operator, given two feasible parent solutions to an NP optimization problem.

We consider the optimal recombination with respect to the main principles of the random crossover operators which are widely used in evolutionary algorithms. The results of Aggarwal et al. (1997); Balas and Niehaus (1998); Borisovsky et al. (2007); Cotta and Troya (2003); Glover et al. (2000); Hohn and Reeves (1996); Meyers and Orlin (2006); Yagiura and Ibaraki (1996) and other authors provide an experimental support to this approach. Some of the algorithms developed in these works are based on binary solution encodings and some are based on more general representations. The notion of optimality in recombination is also somewhat variable. In the present paper, we use a notion of optimal recombination motivated by the principles of (Radcliffe, 1991), applied to the binary solution encodings.

The first examples of polynomially solvable optimal recombination problems for NP-hard optimization problems may be found in the works of Aggarwal et al. (1997) and Balas and Niehaus (1998). There, efficient *optimized crossover* operators were developed and implemented in genetic algorithms for the maximum independent set and

* Partially supported by Russian Foundation for Basic Research grant 07-01-00410.

the maximum clique problems. Throughout the paper we use the term *efficient algorithm* as a synonym for polynomial-time algorithm i.e. algorithm with running time bounded by a polynomial in length of the problem input. A problem which is solved by such an algorithm is *polynomially solvable.* The time complexity is measured in terms of RAM machine (Aho et al., 1974).

In this paper, we formulate the *optimal recombination problem* (ORP) for an NP-optimization problem and show the polynomial solvability or NP-hardness of ORPs for several well-known combinatorial optimization problems. It is supposed that all problem-specific input data are available to the algorithm (i.e. we do not assume the black-box optimization scenario). For the proof of polynomial solvability of the ORPs we take the operators of Aggarwal et al. (1997) and Balas and Niehaus (1998) as a starting point and move to other problems by means of efficient reductions form one ORP to another.

One of the central objects of analysis in the present paper is the Boolean linear programming problem:

$$\max \ f(x) = \sum_{j=1}^{n} c_j x_j, \tag{1}$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m, \tag{2}$$

$$x_j \in \{0, 1\}, \quad j = 1, \ldots, n. \tag{3}$$

Here $x \in \{0,1\}^n$ is the vector of Boolean variables, and the input data c_j, a_{ij} and b_i belong to the set of rational numbers Q. The similar problems where instead of "$\leq$" in (2) we have "$\geq$" or "$=$" for some indices i (or for all i) can be easily transformed to formulation (1)–(3). The number of constraints would increase at most twice. The minimization problems can be considered, using the goal function with coefficients c_j of opposite sign. Where appropriate, we will use a more compact notation for problem (1)–(3):

$$\max \{cx : Ax \leq b, x \in \{0,1\}^n\}.$$

The paper is structured as follows. The formal definition of the ORP for NP optimization problems is introduced in Section 2. Then, using efficient reductions between the ORPs we show in Section 3 that the optimal recombination is computable in polynomial time for the maximum weight set packing problem, the minimum weight set partition problem and for one of the versions of the simple plant location problem. Here we also propose an efficient optimal recombination operator for the Boolean linear programming problems with at most two variables per inequality, extending results from (Aggarwal et al., 1997) and (Balas and Niehaus, 1998).

In Section 4 we consider several NP-hard cases of optimal recombination. In particular, the ORP for the Boolean linear programming problems with 3 variables per inequality is shown to be NP-hard. The ORPs for the one-dimensional knapsack problem, the set covering problem and p-median problem, as well as the maximum satisfiability problem in Boolean linear programming formulation are shown to be NP-hard.

Section 5 is devoted to the concluding remarks and issues for further research.

2 Principles of Optimal Recombination

In some cases below it will be appropriate to use a more general notion of optimization problem, rather than the Boolean linear programming problem. For this purpose we

will employ the standard definition of an NP optimization problem (see e.g. (Ausiello et al., 1999)). By $\{0,1\}^*$ we denote the set of all strings with symbols from $\{0,1\}$ and arbitrary string length.

Definition 1 *An NP optimization problem* Π *is a triple* $\Pi = (I, Sol, f_X)$*, where* $I \subseteq \{0,1\}^*$ *is the set of instances of* Π *and:*

1. I *is recognizable in polynomial time.*

2. Given an instance $X \in I$*,* $Sol(X) \subseteq \{0,1\}^{n(X)}$ *is the set of feasible solutions of* X*. Given* X *and* x*, the decision whether* $x \in Sol(X)$ *may be done in polynomial time, and* $n(X) \leq h(|X|)$ *for some polynomial* h*.*

3. Given an instance $X \in I$*,* $f_X : Sol(X) \rightarrow \mathbf{R}$ *is the objective function (computable in polynomial time) to be maximized if* Π *is an NP maximization problem or to be minimized if* Π *is an NP minimization problem.*

In this definition $n(X)$ stands for the dimension of the Boolean space of solutions of problem instance X. In case different solutions have different length of encoding, $n(X)$ equals the size of the longest solution. If some solutions are shorter than $n(X)$, then the remaining positions are assumed to have zero values.

Formally, the problem instance X may have no feasible solutions. Nevertheless, such cases are meaningless with respect to optimal recombination, which requires two feasible parent solutions at the input of recombination operator. Due to this reason, we will always assume that for the NP optimization problems under consideration $Sol(X) \neq \emptyset$.

According to Definition 1, any NP optimization problem admits a binary representation of solutions. In this sense the results related to the ORPs for binary representations are very general. However, a binary encoding may not be a "natural one", e. g. for the problems involving numerical values of variables or permutations, the binary encodings are "unnatural". In such cases the recombination operators, manipulating the solutions in binary encoding may be of little use for practice. Besides this, there may be a number of NP optimization problems, essentially corresponding to the same problem in practice. Such formulations are usually easy to transform to each other but the solution representations may be quite different in degree of degeneracy, raggedness of fitness landscape, number of local optima, length of encoding string and other parameters important for heuristic algorithms. Since the method of solutions representation is crucial for many properties of recombination operators, in what follows we will always explicitly indicate what representation is used in the ORP.

The following definition of optimal recombination problem is motivated by the principles of recombination formulated by Radcliffe (1991).

Definition 2 *For an NP maximization problem* $\Pi_{\max}$ *the optimal recombination problem is formulated as follows.*

Given an instance X *of* $\Pi_{\max}$ *and two* parent solutions $p^1, p^2 \in Sol(X)$*, find an* offspring solution $x \in Sol(X)$ *with properties*

(a) $x_j = p^1_j$ *or* $x_j = p^2_j$ *for each* $j = 1, \ldots, n(X)$*, and*

(b) for any $x' \in Sol(X)$ *such that* $x'_j = p^1_j$ *or* $x'_j = p^2_j$ *for all* $j = 1, \ldots, n(X)$*, holds* $f_X(x) \geq f_X(x')$.

A definition of the ORP in the case of an NP minimization problem is formulated analogously, with a modification of condition (b):

(b′) for any $x' \in Sol(X)$*, such that* $x'_j = p^1_j$ *or* $x'_j = p^2_j$ *for all* $j = 1, \ldots, n(X)$*, holds* $f_X(x) \leq f_X(x')$.

In what follows, we denote the set of coordinates, where the parent solutions have different values, by $D(p^1, p^2) = \{j : p^1_j \neq p^2_j\}$.

One of the well-known approaches to the analysis of genetic algorithms is based on *schemata*, the subsets of solutions in binary search space, where certain coordinates are fixed to zero or one (Holland, 1975). The problem of optimal recombination may be formulated in a more general way, as a requirement to *respect* a given set of schemata, see e.g. (Radcliffe, 1991). In particular, Definition 2 requires to respect the set of all schemata, which is equivalent to the *gene transmission property* (Radcliffe, 1991) in the case of binary representation. The practical experience indicates that this is a fruitful approach (Aggarwal et al., 1997; Balas and Niehaus, 1998), but ORPs defined w.r.t. some other sets of schemata may be useful as well.

For example, in case of the supply management problem with lower-bounded demands (Borisovsky et al., 2007) experiments showed that it is more appropriate to fix only the variables of common value 0 in the crossover operator. In order to formulate the optimal recombination problem for this case, condition (a) in Definition 2 should be substituted by $x_j \leq p^1_j + p^2_j,\ j = 1, \ldots, n(X)$. This modification of condition (a) is equivalent to the *allelic dynastically optimal recombination* (Cotta, 2003), applied to the binary encoding case. This type of recombination has also shown a promising behavior in applications to set covering problems (Eremeev, 1999; Lourenço et al., 2001).

In what follows, we will consider only the ORPs formulated according to Definition 2.

3 Efficiently Computable Optimal Recombination

3.1 Maximum Independent Set and the Related Problems

As the first examples of efficiently solvable ORPs we will consider the following three well-known problems. Given a graph $G = (V, E)$ with vertex weights $w(v) \in Q,\ v \in V$,

- the maximum weight independent set problem asks for a subset $S \subseteq V$, such that each edge $e \in E$ has at least one endpoint outside S (i.e. S is an independent set) and the weigth $\sum_{v \in S} w(v)$ of S is maximized;
- the maximum weight clique problem asks for a maximum weight subset $Q \subseteq V$, such that any two vertices u, v in Q are adjacent;
- the minimum weight vertex cover problem asks for a minimum weight subset $C \subseteq V$, such that any edge $e \in E$ is incident at least to one of the vertices in C.

Suppose, all vertices of graph G are ordered. We will consider these three problems using the standard binary representation of solutions by the indicator vectors, assuming $n = |V|$ and $x_j = 1$ iff vertex v_j belongs to the subset represented by x. The following proposition summarizes several facts, established in (Aggarwal et al., 1997), (Balas and Niehaus, 1996) and (Balas and Niehaus, 1998).

Proposition 1 *The ORPs for the maximum weight independent set problem, the maximum weight clique problem and the minimum weight vertex cover problem are solvable in time* $O(|D(p^1, p^2)|^3 + n)$*, if the standard binary representation of solutions is used.*

Proof. Consider the minimum weight vertex cover problem on a given graph G with two parent vertex covers C_1 and C_2, represented by binary vectors p^1 and p^2. The

symmetric difference of C_1 and C_2 induces a bipartite subgraph G'. Let V_1', V_2' be the subsets of vertices in this bipartition. The minimum weight vertex cover C' for G' can be found by solving the s-t-minimum cut problem on a supplementary network $\mathcal{N}$, based on G', as described e.g. in (Hochbaum, 1997): in this network, an additional vertex s is connected by outgoing arcs with the vertices of set V_1', and the other additional vertex t is connected by incoming arcs to the subset V_2'. The capacities of the new arcs are equal to the weights of the adjacent vertices in G'. Each edge of G' is viewed as an arc, directed from its endpoint $u \in V_1'$ to the endpoint $v \in V_2'$. The arc capacity is set to $\max\{w(u), w(v)\}$. This s-t-minimum cut problem can be solved in $O(|D(p^1, p^2)|^3)$ time using the maximum-flow algorithm due to A.V. Karzanov – see e.g. Papadimitriou and Steiglitz (1982). We will assume that the s-t-minimum cut contains only the arcs outgoing from s or incoming into t, because if some arc (u, v), $u \in V_1', v \in V_2'$ enters the s-t-minimum cut, one can substitute it by (s, u) or (v, t), not increasing the weight of the cut. Finally, it is easy to verify that C' joined with $C_1 \cap C_2$ constitutes the required solution to the ORP for the minimum weight vertex cover problem on the graph G. Since the parent solutions are given by the n-dimensional indicator vectors p^1 and p^2, we get the overall time complexity $O(|D(p^1, p^2)|^3 + n)$.

The two other problems are closely related to the minimum weight vertex cover (see e.g. (Garey and Johnson, 1979)) and the proof for them is similar. Q.E.D.

To summarize, the optimal recombination algorithm for the minimum weight vertex cover problem is outlined as follows.

1. Compare the parent genotypes p^1, p^2 and compute the set of different genes $D(p^1, p^2)$, which is also the set of indices of the vertices in subgraph G'.
2. Assign $V_k' = \{i \in D(p^1, p^2) : p_i^k = 1\},\ k = 1, 2.$
3. Construct the supplementary network $\mathcal{N}$ as described in the proof of Proposition 1.
4. Solve the s-t-minimum cut problem for network $\mathcal{N}$. Let A_C denote the set of arcs of the minimum cut in $\mathcal{N}$.
5. Substitute each arc $(u, v) \in A_C$, such that $u \in V_1$, $v \in V_2$, by arc (s, u).
6. Assign $C' = \{v \in V' : (s, v) \in A_C\} \cup \{v \in V' : (v, t) \in A_C\}$.
7. Assign $x_j = 1$ iff $j \in C' \cup (C_1 \cap C_2)$ and output x as the result of recombination.

Note that if all vertex weights are equal, then the time complexity of Karzanov's algorithm for the networks of simple structure (as the one constructed in the proof of Proposition 1) reduces to $O(|D(p^1, p^2)|^{2.5})$ – see (Papadimitriou and Steiglitz, 1982).

3.2 Reductions of Optimal Recombination Problems

The usual approach to spreading a class of polynomially solvable (or intractable) problems consists in building chains of efficient problem reductions. The next proposition serves this purpose.

Proposition 2 *Let $\Pi_1 = (I_1, Sol_1, f_X)$ and $\Pi_2 = (I_2, Sol_2, g_Y)$ be both NP maximization problems and $Sol_1(X) \subseteq \{0,1\}^{n_1(X)}$ and $Sol_2(Y) \subseteq \{0,1\}^{n_2(Y)}$. Suppose the ORP is solvable in polynomial time for Π_2 and the following three polynomially computable functions exist:*

1. $\alpha : I_1 \to I_2$.
2. $\beta : Sol_1(X) \to Sol_2(\alpha(X))$, *such that given any two parent solutions from* $\beta(Sol_1(X))$, *the set of optimal solutions to ORP for* I_2 *is within* $\beta(Sol_1(X))$.
3. $\beta^{-1} : \beta(Sol_1(X)) \to Sol_1(X)$, *is the mapping inverse to* β *on the set* $\beta(Sol_1(X))$.

Besides that,

(i) For any $x, x' \in Sol_1(X)$ *such that* $f_X(x) < f_X(x')$, *holds* $g_{\alpha(X)}(\beta(x)) < g_{\alpha(X)}(\beta(x'))$.

(ii) For any $j = 1, \ldots, n_1(X)$ *there exists such* $k(j)$ *that* $\beta^{-1}(y)_j$ *is a function of* $y_{k(j)}$ *on* $\beta(Sol_1(X))$.

(iii) For any $k = 1, \ldots, n_2(Y)$ *there exists such* $j(k)$ *that* $\beta(x)_k$ *is a function of* $x_{j(k)}$ *on* $Sol_1(X)$.

Then the ORP is polynomially solvable for Π_1 *with time complexity, reqired for computing* $\alpha(X)$, $\beta(p^1)$, $\beta(p^2)$ *and* $\beta^{-1}(y)$, *plus the time of optimal recombination on* Π_2 *instance* $\alpha(X)$.

Proof. Suppose, an instance X of problem Π_1 and two parent solutions $p^1, p^2 \in Sol_1(X)$ are given. Consider two feasible solutions $q^1 = \beta(p^1)$, $q^2 = \beta(p^2)$ in $Sol_2(\alpha(X))$. Let us apply the efficient algorithm that solves the ORP for the instance $\alpha(X) \in I_2$ with parent solutions q^1, q^2. By properties of β and β^{-1}, the obtained solution y belongs to $\beta(Sol_1(X))$ and it can be transformed in polynomial time into $z = \beta^{-1}(y) \in Sol_1(X)$.

Note that for all $j \notin D(p^1, p^2)$ holds $z_j = p^1_j = p^2_j$. Indeed, by condition (ii), for any $j = 1, \ldots, n_1(X)$ there exists such $k(j)$ that (I) either $\beta^{-1}(y)_j = y_{k(j)}$ for all $y \in \beta(Sol_1(X))$, or (II) $\beta^{-1}(y)_j = 1 - y_{k(j)}$ for all $y \in \beta(Sol_1(X))$, or (III) $\beta^{-1}(y)_j$ is constant on $\beta(Sol_1(X))$. In the case (I) for all $j \notin D(p^1, p^2)$ we have $z_j = y_{k(j)}$. Now $y_{k(j)} = q^1_{k(j)}$ by the definition of the ORP, since $q^1_{k(j)} = p^1_j = p^2_j = q^2_{k(j)}$. So, $z_j = q^1_{k(j)} = p^1_j = p^2_j$. The case (II) is treated analogously. Finally, the case (III) is trivial since $z, p^1, p^2 \in \beta^{-1}(\beta(Sol_1(X)))$. So, z is a feasible solution to the ORP for Π_1.

To prove the optimality of z in the ORP for Π_1 we will assume by contradiction that there exists $\zeta \in Sol_1(X)$ such that $\zeta_j = p^1_j = p^2_j$ for all $j \notin D(p^1, p^2)$ and $f_X(\zeta) > f_X(z)$. Then $g_{\alpha(X)}(\beta(\zeta)) > g_{\alpha(X)}(\beta(z)) = g_{\alpha(X)}(y)$. But $\beta(\zeta)$ coincides with q^1 and q^2 in all coordinates $k \notin D(q^1, q^2)$ according to condition (iii), thus y is not an optimal solution to the ORP for $\alpha(X)$, which is a contradiction. Q.E.D.

Note 1 *If* Π_1 *is an NP minimization problem then the statement of Proposition 2 can be applied as well, but with a reversed sign in the first inequality of (i), i.e.* $f_X(x) > f_X(x')$.

Note 2 *Analogously, if* Π_2 *is an NP minimization problem then Proposition 2 applies with a reversed sign of the second inequality in (i), i.e.* $g_{\alpha(X)}(\beta(x)) > g_{\alpha(X)}(\beta(x'))$.

The special case of this proposition where $n_1(X) = n_2(Y)$ and $k(j) = j$, $j(k) = k$ appears to be the most applicable, as it is demonstrated in what follows.

3.3 Set Packing, Set Partition and Simple Plant Location Problems

Let us use Proposition 2 to obtain an efficient optimal recombination algorithm for the *set packing problem:*

$$\max \{ f_{\text{pack}}(x) = cx : Ax \le e, x \in \{0,1\}^n \}, \tag{4}$$

where A is a given $(m \times n)$-matrix of zeros and ones and e is an m-vector of ones. The transformation α from the set packing to the maximum weight independent set

problem (with the standard binary solutions encoding) consists in building a graph on a set of vertices $v_1, \ldots, v_n$ with weights $c_1, \ldots, c_n$. Each pair of vertices v_j, v_k is connected by an edge iff j and k both belong at least to one of the subsets $N_i = \{j : a_{ij} \neq 0\}$. In this case β is an identical mapping. Application of Proposition 2 leads to

Corollary 1 *The ORP for the maximum weight set packing problem* (4) *is solvable in time* $O(|D(p^1, p^2)|^3 + n^2 m)$*, if the solutions are represented by vectors* $x \in \{0,1\}^n$.

In many reductions of NP optimization problems the set of feasible solutions of the original instance corresponds to a subset of "high-quality" feasible solutions in the transformed formulation. In terms of Proposition 2, this subset of "high-quality" feasible solutions becomes the set $\beta(Sol_1(X))$. Let us define the set of "high-quality" feasible solutions for an NP maximization problem Π_2 as follows

$$Sol_2^X(\alpha(X)) = \left\{ y \in Sol_2(\alpha(X)) : g(y) \geq \min_{x \in Sol_1(X)} g(\beta(x)) \right\},$$

analogously, if Π_2 is an NP minimization problem, then

$$Sol_2^X(\alpha(X)) = \left\{ y \in Sol_2(\alpha(X)) : g(y) \leq \max_{x \in Sol_1(X)} g(\beta(x)) \right\}.$$

Now we can prove the polynomial solvability of the next two problems in Boolean linear programming formulations:

- The *minimum weight set partition problem:*

$$\min \{ f_{\text{part}}(x) = cx : Ax = e, x \in \{0,1\}^n \}, \tag{5}$$

  where A is a given $(m \times n)$-matrix of zeros and ones.

- The *simple plant location problem:*

$$\min \ f_{\text{splp}}(\mathbf{x}, \mathbf{y}) = \sum_{k=1}^{K} \sum_{\ell=1}^{L} c_{k\ell} x_{k\ell} + \sum_{k=1}^{K} C_k y_k, \tag{6}$$

  subject to

$$\sum_{k=1}^{K} x_{k\ell} = 1, \quad \ell = 1, \ldots, L, \tag{7}$$

$$y_k \geq x_{k\ell}, \quad k = 1, \ldots, K, \ \ell = 1, \ldots, L, \tag{8}$$

$$x_{k\ell} \in \{0,1\}, \ y_k \in \{0,1\}, \quad k = 1, \ldots, K, \ \ell = 1, \ldots, L. \tag{9}$$

  Here and below, we denote the $(K \times L)$-matrix of Boolean variables $x_{k\ell}$ by $\mathbf{x}$, and the K-dimensional vector of Boolean variables y_k is denoted by $\mathbf{y}$. The costs $c_{k\ell} \in Q$, $C_k \in Q$ are nonnegative. The motivation for this problem and some of its applications can be found in (Krarup and Pruzan, 1983) and the references provided there.

In the following corollary of Proposition 2, the image $\beta(Sol_1(X))$ will coincide with $Sol_2^X(\alpha(X))$.

Corollary 2 *(i) For the minimum weight set partition problem (5), where the solutions are represented by vectors $x \in \{0,1\}^n$, the ORP is solvable in time $O(|D(p^1,p^2)|^3 + n^2m)$.*

(ii) For the simple plant location problem, where the solutions are represented as couples $(\mathbf{x},\mathbf{y})$, the ORP is solvable in polynomial time.

Proof. For both cases we will use the well-known transformations of the corresponding NP optimization problems to the set packing problem (see e.g. the transformations T2 and T5 in (Krarup and Pruzan, 1983)).

(i) Let us denote the minimum weight set partition problem by Π_1. The input of its ORP consists of an instance $X \in I_1$ and two parent solutions, thus $Sol_1(X) \neq \emptyset$ and X is equivalent to

$$\min \ \sum_{j=1}^{n} c_j x_j + \lambda \sum_{i=1}^{m} w_i,$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j + w_i = 1, \ i = 1,\dots,m,$$

$$x_j \in \{0,1\}, \ j = 1,\dots,n; \ \ w_i \geq 0, \ i = 1,\dots,m,$$

where $\lambda > 2\sum_{j=1}^{n} |c_j|$ is a penalty factor which assures that all "artificial" slack variables w_i become zeros in the optimal solution. By substitution of w_i into the objective function, the latter model transforms into

$$\min \left\{ \lambda m + \sum_{j=1}^{n} \left(c_j - \lambda \sum_{i=1}^{m} a_{ij} \right) x_j : Ax \leq e, \ x \in \{0,1\}^n \right\},$$

which is equivalent to the following instance $\alpha(X)$ of the set packing problem Π_2:

$$\max \left\{ g(x) = \sum_{j=1}^{n} \left(\lambda \sum_{i=1}^{m} a_{ij} - c_j \right) x_j : Ax \leq e, \ x \in \{0,1\}^n \right\}.$$

We will assume that β is an identical mapping. Then each feasible solution x of the set partition problem becomes a "high quality" feasible solution to problem Π_2 with the objective function value $g(x) = \lambda m - f_{\text{part}}(x) > \lambda(m - 1/2)$. At the same time, if a vector x' is feasible for problem Π_2 but infeasible in the set partition problem, it will have the objective function value $g(x') = \lambda(m-k) - f_{\text{part}}(x')$, where k is the number of constraints $\sum_{j=1}^{n} a_{ij} x_j = 1$, violated by x'. So, β is a bijection from $Sol_1(X)$ to

$$Sol_2^X(\alpha(X)) = \{x \in Sol_2(\alpha(X)) : g(x) > \lambda(m - 1/2)\}.$$

The ORP for Π_2 can be solved in polynomial time by Corollary 1. Thus, application of Proposition 2 and Note 1 completes the proof of part (i).

(ii) Let Π_1' be the simple plant location problem. Analogously to the case (i) we will convert equations (7) into inequalities. To this end, we rewrite (7) as $\sum_{k=1}^{K} x_{k\ell} + w_\ell = 1, \ \ell = 1,\dots,L$, with nonnegative slack variables w_ℓ and ensure all of them turn into zero in the optimal solution, by means of a penalty term $\lambda \sum_{\ell=1}^{L} w_\ell$ added to the objective function. Here

$$\lambda > \sum_{k=1}^{K} C_k + \sum_{\ell=1}^{L} \max_{k=1,\dots,K} c_{k\ell}.$$

Eliminating variables w_ℓ we substitute (7) by $\sum_{k=1}^{K} x_{k\ell} \le 1,\ \ell = 1,\dots,L$, and change the penalty term into $\lambda L - \lambda \sum_{\ell=1}^{L}\sum_{k=1}^{K} x_{k\ell}$. Multiplying the criterion by -1 and introducing a new set of variables $\overline{y}_k = 1 - y_k,\ k = 1,\dots,K$, we obtain the following NP maximization problem Π_2':

$$\max\ g'(\mathbf{x},\overline{\mathbf{y}}) = \sum_{k=1}^{K}\sum_{\ell=1}^{L}(\lambda - c_{k\ell})x_{k\ell} + \sum_{k=1}^{K} C_k\overline{y}_k - \lambda L - \sum_{k=1}^{K} C_k, \tag{10}$$

subject to

$$\sum_{k=1}^{K} x_{k\ell} \le 1, \quad \ell = 1,\dots,L, \tag{11}$$

$$\overline{y}_k + x_{k\ell} \le 1, \quad k = 1,\dots,K,\ \ell = 1,\dots,L, \tag{12}$$

$$x_{k\ell} \in \{0,1\},\ \overline{y}_k \in \{0,1\}, \quad k = 1,\dots,K,\ \ell = 1,\dots,L, \tag{13}$$

where $\mathbf{x},\overline{\mathbf{y}}$ are the matrix $(x_{k\ell})$ and the vector $(\overline{y}_k)$ of variables. Obviously, Π_2' is a special case of the set packing problem, if one subtracts the constant $-\lambda L - \sum_{k=1}^{K} C_k$ from the objective function. Thus, we have defined the mapping $\alpha(X)$.

Assume that β maps identically all variables $x_{k\ell}$ and transforms the variables y_k as $\overline{y}_k = 1 - y_k,\ k = 1,\dots,K$. Then each feasible solution $(\mathbf{x},\mathbf{y})$ of the simple plant location problem becomes a "high quality" feasible solution to problem Π_2' with an objective function value $g'(\mathbf{x},\overline{\mathbf{y}}) = -f_{\text{splp}}\mathbf{x},\mathbf{y}) > -\lambda$. If a pair $(\mathbf{x},\overline{\mathbf{y}})$ is feasible for problem Π_2' but $(\mathbf{x},\mathbf{y})$ is infeasible in the simple plant location problem, then $g'(\mathbf{x},\overline{\mathbf{y}}) \le -f_{\text{splp}}(\mathbf{x},\mathbf{y}) - \lambda$, because one of the equalities (7) is violated by $(\mathbf{x},\mathbf{y})$.

The ORP for the problem Π_2' can be solved in polynomial time by Corollary 1, thus Proposition 2 and Note 1 give the required optimal recombination algorithm for Π_1'. Q.E.D.

Let us summarize the transformations α and β proposed in Corollary 2. In the case (i), the minimum weight set partition problem with parameters m, n, A, c is mapped by the transformation α to the set packing problem with parameters m', n', A', c', where $m' = m, n' = n$, $c' = \lambda eA - c$, $A' = A$. The mapping β is identical.

In the case (ii), the simple plant location problem with parameters K, L, C, c is mapped by the transformation α to the set packing problem with parameters m', n', A', c', where $m' = L(1+K), n' = K(L+1)$,

$$c' = (\lambda - c_{11}, \lambda - c_{12}, \dots, \lambda - c_{1L}, \dots, \lambda - c_{KL}, C_1, C_2, \dots, C_K),$$

$$A' = \begin{pmatrix} I_{L\times L} & I_{L\times L} & \dots & I_{L\times L} & 0 & 0 & \dots & 0 \\ & & & & \mathbf{1}_L & \mathbf{0}_L & \dots & \mathbf{0}_L \\ & & & & \mathbf{0}_L & \mathbf{1}_L & \dots & \mathbf{0}_L \\ & & I_{KL\times KL} & & & & \ddots & \\ & & & & \mathbf{0}_L & \mathbf{0}_L & \dots & \mathbf{1}_L \end{pmatrix}. \tag{14}$$

Here we denote an l-dimensional identity matrix by $I_{l\times l}$ and a column l-vector of zeros (or ones) by $\mathbf{0}_l$ (or $\mathbf{1}_l$). The mapping $x' = \beta(\mathbf{x},\mathbf{y})$ is as follows:

$$x'_j = \begin{cases} x_{\lceil j/L\rceil, 1+(j-1) \bmod L}, & \text{if } j \le KL \\ 1 - y_{j-KL}, & \text{if } j > KL \end{cases}, \quad j = 1,\dots,K(L+1).$$

If a vector $\mathbf{y} \in \{0,1\}^K$ is fixed, then the best possible solution to the simple plant location problem with this $\mathbf{y}$ can be easily constructed: for each ℓ one has to assign one of the variables $x_{k\ell} = 1$, so that $c_{k\ell} \leq c_{k'\ell}$ for all such k' that $y_{k'} = 1$. So, it suffices to specify just a vector $\mathbf{y}$ to represent a tentative solution to this problem in any heuristic algorithm. It is easy to see that it is impossible to construct some of the non-optimal feasible solutions to problem (6)–(9) this way, but the optimal solution remains. Strictly speaking, the representation of solutions, given by the vector $\mathbf{y}$ constitutes another NP-minimization problem with a reduced set of feasible solutions. In Section 4 it we will be shown that the ORP for this version of the simple plant location problem is NP-hard.

3.4 Generalization of the Basic Result Using Hypergraphs

The starting point of all reductions considered above was Proposition 1 which may be viewed as an efficient reduction of the three ORPs, mentioned there, to the maximum weight independent set problem in a bipartite graph. In order to generalize this approach now we will move from ordinary graphs to hypergraphs.

A hypergraph $H = (V, E)$ is given by a finite nonempty set of vertices V and a set of edges E, where each edge $e \in E$ is a subset of V. A subset $S \subseteq V$ is called *independent* if none of the edges $e \in E$ is a subset of S. The maximum weight independent set problem on hypergraph $H = (V, E)$ with rational vertex weights $w(v),\ v \in V$ asks for an independent set S with maximum weight $\sum_{v \in S} w(v)$.

A generalization of the bipartite graph is the *2-colorable hypergraph*: there exists a partition of the vertex set V into two disjoint independent subsets C_1 and C_2. The partition $V = C_1 \cup C_2$, $C_1 \cap C_2 = \emptyset$ is called a 2-coloring of H and C_1, C_2 are the color classes.

Like in the case of the set packing problem, let us denote by N_i the set of indices of non-zero elements in constraint i of the Boolean linear programming problem (1)-(3). In the sequel we will assume that at least one of the subsets N_i contains two or more elements (otherwise the problem is solved trivially). The following proposition gives an extension of Proposition 1 to the context of hypergraphs.

Proposition 3 *The ORP for Boolean linear programming problem (1)-(3) reduces to the maximum weight independent set problem on a 2-colorable hypergraph with a 2-coloring given in the input. The time complexity of this reduction is $O(m(2^{N_{\max}} + n))$ and each edge in the resulting hypergraph contains no more than $N_{\max}$ vertices, where $N_{\max} = \max_{i=1,\ldots,m} |N_i|$ is the maximal number of non-zero elements per linear constraint.*

Proof. Given an instance of the Boolean linear programming problem with parent solutions p^1 and p^2, let us denote the number of distinct coordinates $|D(p^1, p^2)|$ by d and construct a hypergraph H on $2d$ vertices, assigning each variable $x_j, j \in D(p^1, p^2)$ a couple of vertices v_j, v_{n+j}. In order to model each of the linear constraints for $i = 1, \ldots, m$ we will look through of all possible combinations of the Boolean variables from $D(p^1, p^2)$, involved in this constraint:

$$\{x \in \{0,1\}^n : x_j = 0\ \forall j \notin N_i \cap D(p^1, p^2)\}.$$

Let $x^{ik}, k = 1, \ldots, 2^{|N_i \cap D(p^1,p^2)|}$ denote the k-th element in this set. For each combination k violating a constraint i from (2), i.e. if

$$\sum_{j \in N_i \cap D(p^1,p^2)} a_{ij} x_j^{ik} + \sum_{j \in N_i \setminus D(p^1,p^2)} a_{ij} p_j^1 > b_i$$

holds, we add an edge

$$e_{ik} = \{v_j : x_j^{ik} = 1,\ j \in N_i \cap D(p^1, p^2)\} \cup \{v_{j+n} : x_j^{ik} = 0,\ j \in N_i \cap D(p^1, p^2)\}$$

into the hypergraph. (Note that the edge e_{ik} contains not more than $|N_i|$ elements.) Besides that, we add d edges $\{v_j, v_{n+j}\}, j \in D(p_1, p_2)$, to guarantee that both v_j and v_{n+j} can not enter into an independent set together.

If x is a feasible solution to the ORP for (1)-(3), then the set of vertices

$$S(x) = \{v_j : x_j = 1, j \in D(p_1, p_2)\} \cup \{v_{j+n} : x_j = 0, j \in D(p_1, p_2)\}$$

is independent in H. Given a set of vertices S, we can construct the corresponding vector $x(S)$, setting $x(S)_j = 1$ iff $v_j \in S, j \in D(p^1, p^2)$ or if $p_j^1 = p_j^2 = 1$. Then for each independent set S of d vertices, $x(S)$ is feasible in the Boolean linear programming problem.

The hypergraph vertices are given the following weights:

$$w(v_j) = c_j + \lambda,\ w(v_{n+j}) = \lambda,\ j \in D(p^1, p^2),$$

where $\lambda > 2\sum_{j \in D(p_1, p_2)} |c_j|$.

Now each maximum weight independent set S^* contains either v_j or v_{n+j} for any $j \in D(p^1, p^2)$. Indeed, there must exist a feasible solution to the ORP and it corresponds to an independent set of weight at least λd. However, if an independent set neither contains v_j nor v_{n+j} then its weight is below $\lambda d - \lambda/2$.

So, optimal independent set S^* corresponds to a feasible vector $x(S^*)$ with the goal function value

$$cx(S^*) = \sum_{j \in S^*,\ j \le n} c_j + \sum_{j \notin D(p^1, p^2)} c_j p_j^1 = w(S^*) - \lambda d + \sum_{j \notin D(p^1, p^2)} c_j p_j^1.$$

Under the inverse mapping $S(x)$, any feasible vector x yields an independent set of weight

$$w(S(x)) = cx + \lambda d - \sum_{j \notin D(p^1, p^2)} c_j p_j^1,$$

therefore $x(S^*)$ must be an optimal solution to the ORP as well. Q.E.D.

Note that if an edge $e \in H$ consists of a single vertex, $e = \{v\}$, then the vertex v can not enter into the independent sets. All of such vertices may be excluded from H, constructed in Proposition 3, if one wants to find a maximum independent set in H. Let us denote the resulting hypergraph by H'. If $N_{\max} \le 2$, then the hypergraph H' is just an ordinary graph with at most $2d$ vertices. Thus, by Proposition 3 the ORP reduces to solving the maximum weight independent set problem in a bipartite graph H', which is solvable in $O(d^3)$ operations (see e.g. the proof of Proposition 1). Using this fact, we can extend the result from (Balas and Niehaus, 1998) as follows:

Corollary 3 *The ORP for linear Boolean programming problem with at most two variables per inequality is solvable in time $O(|D(p^1, p^2)|^3 + mn)$, if the solutions are represented by vectors $x \in \{0, 1\}^n$.*

The class of linear Boolean programming problems with at most two variables per inequality includes e.g. the vertex cover problem and the *minimum 2-satisfiability problem* – see e.g (Hochbaum, 1997). The latter problem is formulated as follows: given

a set of logical variables $\xi_j \in \{\text{true}, \text{false}\}, j = 1, \ldots, n$ with non-negative weights and a collection of clauses in conjunctive normal form, where each clause contains at most two logical variables or their negations (2-CNF), find a satisfying truth assignment such that the total weight of true variables is minimal. The previous corollary implies

Corollary 4 *The ORP for the minimum 2-satisfiability problem is solvable in time $O(n^3)$, if the truth assignment $\xi \in \{true, false\}^n$ is used for representation of solutions.*

Corollaries 3 and 4 could be also obtained through Propositions 1,2, using the transformation of the minimum 2-satisfiability problem to the vertex cover problem proposed by Seffi Naor (Hochbaum, 1997), but that would give a more complex reduction.

An appropriate question is whether it is really important to indicate that together with the hypergraph we supply a 2-coloring of in Proposition 3. Providing a 2-coloring together with the hypergraph may be helpful in the cases, where the 2-coloring is used for finding the maximum weight independent set. For instance, in Proposition 1 the bi-partitioning of the vertices of subgraph G' is used for finding the ORP solution, and this is equivalent to usage of a 2-coloring for finding the maximal independent set in H. Of course, in the case of ordinary bipartite graphs the bi-partitioning could be easily obtained even if we only new all values of $p^1_j = p^2_j,\ j \notin D(p^1, p^2)$, and none of the values $p^1_j, p^2_j,\ j \in D(p^1, p^2)$. However, in the general case, a straightforward search for a 2-coloring may be computationally expensive. For example, it is known that when each edge of a hypergraph consists of 4 vertices, finding a 2-coloring for a 2-colorable hypergraph is NP-hard (Guruswami et al., 2002). Thus, whenever a 2-coloring is available for a hypergraph, we will indicate this.

Finally, note that on the class of Boolean linear programming problems with $O(\ln(n))$ non-zero elements per inequality, the ORP is polynomially reducible to the maximum weight independent set problem on a hypergraph with a given 2-coloring, as it follows from Proposition 3.

4 NP-hard Cases of Optimal Recombination

4.1 Boolean Linear Programming Problems with Bounded Number of Non-zeros per Inequality

We have seen that the optimal recombination on the class of Boolean linear programming problems is related to the maximum weight independent set problem on hypergraphs with a given 2-coloring. The next proposition indicates that, unfortunately, in the general case the latter problem is NP-hard.

Proposition 4 *Finding a maximum size independent set in a hypergraph with all edges of size 3 is NP-hard even if a 2-coloring is given.*

Proof. Let us construct a reduction from the NP-hard maximum size independent set problem on an ordinary graph to our problem. Given a graph $G = (V, E)$ with the set of vertices $V = \{v_1, \ldots, v_n\}$, consider a hypergraph $H = (V', E')$ on the set of vertices $V' = \{v_1, \ldots, v_{2n}\}$, where for each edge $e = \{v_i, v_j\} \in E$ there are n edges of the form $\{v_i, v_j, v_{n+k}\},\ k = 1, \ldots, n$ in E'. A 2-coloring for this hypergraph can be composed of color classes $C_1 = V$ and $C_2 = \{v_{n+1}, \ldots, v_{2n}\}$. Any maximum size independent set in this hypergraph consists of the set of vertices $\{v_{n+1}, \ldots, v_{2n}\}$ joined with a maximum size independent set S^* on G. Therefore, any maximum size independent set for H immediately induces a maximum size independent set for G, which is NP hard to obtain. Q.E.D.

The maximum size independent set problem in a hypergraph $H = (V, E)$ may be formulated as a Boolean linear programming problem

$$\max \left\{ \sum_{j=1}^{n} x_j : Ax \leq b, x \in \{0,1\}^n \right\} \tag{15}$$

with $m = |E|, n = |V|$, $b_i = |e_i| - 1,\ i = 1, \ldots, m$ and $a_{ij} = 1$ iff $v_j \in e_i$, otherwise $a_{ij} = 0$. In the special case where H is 2-colorable, we can take p^1 and p^2 as the indicator vectors for the color classes C_1 and C_2 of any 2-coloring. Then $D(p^1, p^2) = \{1, \ldots, n\}$ and the ORP for the Boolean linear programming problem (15) becomes equivalent to solving the maximum size independent set in a hypergraph H with a given 2-coloring. In view of Proposition 4, this leads to the following

Corollary 5 *The optimal recombination for Boolean linear programming problem is NP-hard in the strong sense even in the case where $|N_i| = 3$ for all $i = 1, \ldots, m$; $c_j = 1$ for all $j = 1, \ldots, n$ and matrix A is Boolean.*

In the rest of this section we will discuss NP-hardness of the ORP for several well-known Boolean programming problems.

4.2 One-Dimensional Knapsack Problem

Let us consider the complexity of the ORP for the one-dimensional knapsack problem

$$\max \left\{ cx : \sum_{j=1}^{n} a_j x_j \leq b, x \in \{0,1\}^n \right\}, \tag{16}$$

$a_j \geq 0, c_j \geq 0, j = 1, \ldots, n$ are integer.

Proposition 5 *The ORP for the one-dimensional knapsack problem (16) is NP-hard, assuming the binary representation of solutions by vector x.*

Proof. Consider the following NP-complete case of the partition problem (see e.g. (Schuurman and Woeginger, 2006)): the input consists of $2m$ positive integers $\alpha_j, j = 1, \ldots, 2m$, such that

$$\frac{B}{m+1} < \alpha_j < \frac{B}{m-1},\ \ j = 1, \ldots, 2m, \tag{17}$$

where $B = \sum_{j=1}^{2m} \alpha_j / 2$. The problem is to decide if there exists such $x \in \{0,1\}^{2m}$ that

$$\sum_{j=1}^{2m} \alpha_j x_j = B. \tag{18}$$

The NP-completeness of this special case of the partition problem can be shown by reduction from the following NP-complete modification of the partition problem (Garey and Johnson, 1979): given a set of $2m$ positive integers $\alpha'_j,\ j = 1, \ldots, 2m$, recognize the existence of such $x \in \{0,1\}^{2m}$, that

$$\sum_{j=1}^{2m} x_j = m \ \text{ and } \ \sum_{j=1}^{2m} \alpha'_j x_j = \frac{1}{2} \sum_{j=1}^{2m} \alpha'_j. \tag{19}$$

The reduction consists in setting $\alpha_i = \alpha'_i + M,\ i = 1, \ldots, 2m$ with a sufficiently large integer M, e.g. $M = 2m \max\{\alpha'_j : j = 1, \ldots, 2m\}$. It is straightforward to verify (17) and to prove the equivalence of (19) to (18) with this set of α_i.

Now we will construct a polynomial-time Turing reduction from the NP-complete case of the partition problem, satisfying (17), to the ORP for knapsack problem (16) with $n = 2m$, $b = B$ and $c_j = a_j = \alpha_j,\ j = 1, \ldots, n$.

Note that $a_j > b/(m+1),\ j = 1, \ldots, n$ implies that any feasible solution to the knapsack problem contains at most m ones. It is sufficient for us that the optimal solution contains less than $2m-2$ ones (when $m > 2$). This means that if we try to fix equal to zero each of $\binom{2m}{2}$ couples of variables with the indices $\{i_k, j_k\},\ k = 1, \ldots, \binom{2m}{2}$ and solve each of the resulting knapsack problems for the remaining $2m-2$ variables, then the objective value for the best outcome will be equal to b iff the answer to the partition problem is "yes". But one can solve each of these supplementary problems as an ORP problem with appropriate parent solutions. To do this one can set $p^1_{i_k} = p^2_{j_k} = 0$ and fill the remaining positions $j \notin \{i_k, j_k\}$ so that $p^1_j + p^2_j = 1$ and there are $m-1$ ones in each of the parents. Both of such parent solutions are feasible since $a_j < b/(m-1),\ j = 1, \ldots, n$. Q.E.D.

4.3 Set Covering and Location Problems

The next example of an NP-hard ORP is that for the set covering problem, which may be considered as a special case of (1)-(3):

$$\min \{cx : Ax \geq e,\ x \in \{0,1\}^n\}, \tag{20}$$

where A is a Boolean $(m \times n)$-matrix; $c_j \geq 0,\ j = 1, \ldots, n$. Let us assume the binary representation of solutions by the vector x. Given an instance of the set covering problem, one may construct a new instance with a doubled set of columns in the matrix $A' = (AA)$ and a doubled vector $c' = (c_1, \ldots, c_n, c_1, \ldots, c_n)$. Then an instance of the NP-hard set covering problem (20) is equivalent to the ORP for the modified set covering instance where the input consists of $(m \times 2n)$-matrix A', $2n$-vector c' and the feasible parent solutions p^1, p^2, with $p^1_j = 1, p^2_j = 0$ for $j = 1, \ldots, n$ and $p^1_j = 0, p^2_j = 1$ for $j = n+1, \ldots, 2n$. So, the ORP for the set covering problem is also NP-hard.

It is interesting that in some cases the ORP may be even harder than the original problem (assuming P$\neq$ NP). This can be illustrated on the example of the set covering problem. A special case of this problem, defined by the restriction $a_{i,1} = 1, i = 1, \ldots, m;\ c_1 = 0$ is trivially solvable: $x = (1, 0, 0, \ldots, 0)$ is the optimal solution. However, in the case $p^1_1 = p^2_1 = 0$, the ORP becomes NP-hard on this restriction as well.

The set covering problem may be efficiently transformed to the simple plant location problem (6)-(9) – see e.g. transformation T3 in (Krarup and Pruzan, 1983). To do this, it suffices to take $K = n,\ L = m,\ C_k = c_k$ for $k = 1, \ldots, n$ and to set

$$c_{k\ell} = \begin{cases} \sum_{j=1}^n c_j + 1, & \text{if } a_{\ell k} = 0, \\ 0, & \text{if } a_{\ell k} = 1, \end{cases} \quad \text{for all } k = 1, \ldots, n,\ \ell = 1, \ldots, m.$$

It is easy to verify that a vector $\mathbf{y}$ in the optimal solution to this instance of the simple plant location problem will be an optimal set covering solution as well.

Thus, if the solution representation in problem (6)-(9) is given only by the vector $\mathbf{y}$, then this reduction meets the conditions of Proposition 2, in view of Notes 1, 2. The set of "high quality" solutions to the simple plant location problem is characterized in this case by the threshold on objective function $f_{\text{splp}}(\mathbf{y}) < \sum_{j=1}^n c_j + 1$, which ensures that all constraints of the set covering problem are met. If the ORP for (6)-(9) with this representation was polynomially solvable, this would imply P=NP, therefore, we have

Proposition 6 *The ORP for the simple plant location problem (6)-(9), where the solutions are represented only by the vector* $\mathbf{y}$*, is NP-hard.*

The well-known p-median problem may be defined using the formulation of the simple plant location problem (6)-(9): it suffices to assume that all $C_k = 0$, adding a single constraint $\sum_{k=1}^{K} y_k = p$, where $1 \leq p \leq n$ is a given parameter. For this problem the following proposition holds.

Proposition 7 *The ORP for the p-median problem, where the solutions are represented by the vector $\mathbf{y}$, is NP-hard.*

Proof. The authors of (Alekseeva et al., 2007) propose a reduction of an NP-hard graph partitioning problem to the p-median problem with $n = |V|$ and $p = |V|/2$, where V is the set of the graph vertices and $|V|$ is even. Thus, this special case of the p-median problem is NP-hard as well. Now if the vector $\mathbf{y}$ is used for the solutions representation, we can consider an ORP for this case of the p-median problem with parent solutions $p^1 = (1, \ldots, 1, 0, \ldots, 0)$ and $p^2 = (0, \ldots, 0, 1, \ldots, 1)$ of $n/2$ ones. Obviously, such ORP instance is equivalent to the given p-median instance, which implies the NP-hardness of the ORP. Q.E.D.

4.4 Maximum Satisfiability Problem

Another example of an NP-hard ORP is the maximum satisfiability problem (MAX-SAT): given a CNF formula with clauses $\mathbf{c}_1, \ldots, \mathbf{c}_M$, where each clause is a disjunction of logical variables or their negations, it is required to maximize the number of satisfied clauses. The vector $\xi \in \{\text{true,false}\}^N$ of N logical variables is the most natural and compact representation of solutions of MAX-SAT. Optimal recombination for MAX-SAT with this encoding is NP-hard: just consider the parent solutions where $p_j^1 + p_j^2 = 1,\ j = 1, \ldots, N$, which make the ORP equivalent to the original MAX-SAT problem and it is NP-hard. Notice that in this formulation MAX-SAT does not belong to the class of Boolean integer programming problems.

In the Boolean integer programming formulation the MAX-SAT remains NP-hard. To see this, let us introduce the Boolean variables $y_1, \ldots, y_N$, assuming for $j = 1, \ldots, N$ that $y_j = 1$ iff $\xi_j = \text{true}$. Introduce additionally a set of Boolean variables $z_1, \ldots, z_M$, corresponding to the clauses of the CNF. Let C_i^- denote the set of indices of logical variables that belong to a clause i in negated form and let C_i^+ correspond to the non-negated variables of the clause. Then the MAX-SAT problem can be written as follows (Cheriyan et al., 1996):

$$\max\ f_{\text{sat}}(y) = \sum_{i=1}^{M} z_i, \tag{21}$$

$$\sum_{j \in C_i^-} y_j - \sum_{j \in C_i^+} y_j + z_i \leq |C_i^-|,\ \ i = 1, \ldots, M, \tag{22}$$

$$y_j \in \{0, 1\},\ j = 1, \ldots, N,\ \ z_i \in \{0, 1\},\ i = 1, \ldots, M. \tag{23}$$

If a variable z_i equals 1, this implies that the vector y gives a true assignment to clause $\mathbf{c}_i$.

Proposition 8 *The optimal recombination is NP-hard for the MAX-SAT problem in Boolean linear programming formulation (21)–(23), assuming solutions representation (y, z).*

Proof. Suppose, an arbitrary CNF with clauses $\mathbf{c}'_1, \ldots, \mathbf{c}'_{M'}$ and variables $\xi_1, \ldots, \xi_{N'}$ is given. Consider a modified set of clauses $\mathbf{c}_1, \ldots, \mathbf{c}_{M'}$, where each clause $\mathbf{c}_i\ i = 1, \ldots, M'$ contains all literals of clause $\mathbf{c}'_i$, appended with a new logical variable $\xi_{N'+i}$. Let us construct one new clause $\mathbf{c}_{M'+i} = (\overline{\xi_{N'+i}})$ for each $i = 1, \ldots, M'$.

Here $\bar{\xi}$ denotes the negation of a variable ξ. Note that any assignment of variables, where clause $\mathbf{c}_{M'+i}$ evaluates to "true", will also satisfy the clause $\mathbf{c}_i$ iff the original clause $\mathbf{c}'_i$ is satisfied.

Now let us consider the ORP for the constructed instance $\mathbf{c}_1, \ldots, \mathbf{c}_M$ of MAX-SAT, using formulation (21)–(23) with $N = N' + M'$ and $M = 2M'$. Consider the parent solutions $p^1 = (y^1, z^1)$ and $p^2 = (y^2, z^2)$, where $y^1 = (0, \ldots, 0)$, $y^2 = (1, \ldots, 1)$ and

$$z_i^k = \begin{cases} 1, & \text{if } \sum\limits_{j \in C_i^-} y_j^k - \sum\limits_{j \in C_i^+} y_j^k < |C_i^-| \\ 0, & \text{otherwise,} \end{cases}, \quad i = 1, \ldots, M, \ k = 1, 2.$$

By the definition of this instance, $z^1_{M'+1} = \ldots = z^1_M = 1$ and $z^2_1 = \ldots = z^2_{M'} = 1$, but $z^2_{M'+1} = \ldots = z^2_M = 0$. Thus, in the optimal recombination problem we should only fix equal to 1 the variables z_i, $i \leq M'$, for which $z_i^1 = 1$. The rest of the variables are set free.

By means of the described reduction, the NP-complete satisfiability recognition problem (SAT) can be solved on the basis of the value $f_{\text{sat}}(y)$, where y is the optimal solution to the constructed ORP instance. Indeed, let us give the positive answer for SAT iff $f_{\text{sat}}(y) = 2M'$. On the one hand, whenever a satisfying assignment $\xi_1, \ldots, \xi_{N'}$ exists, the corresponding vector $y = (y_1, \ldots, y_{N'}, 0, 0, \ldots, 0)$ yields $f_{\text{sat}}(y) = 2M'$. On the other hand, when $f_{\text{sat}}(y) = 2M'$, this implies that all clauses $\mathbf{c}_i$ are satisfied, as well as all clauses $\mathbf{c}'_i$ of the original CNF. Q.E.D.

As it was demonstrated in Corollary 2 (ii) and Proposition 6, even in the cases where the most natural representation of solutions induces an NP-hard ORP, additional redundancy in the representation can make the ORP polynomially solvable.

Another example of this type may be observed on the unit-cost maximum 3-satisfiability problem (MAX-3-SAT), a restriction of the unit-cost MAX-SAT where each clause involves only three logical variables or their negations. By the same reasoning as in the general case of MAX-SAT it follows that when the vector of logical variables is taken as solution representation, the optimal recombination problem for the unit-cost MAX-3-SAT is NP-hard.

Instead, we can move to a formulation of the MAX-3-SAT with a graph-based representation. This representation is motivated by a reduction of the MAX-3-SAT to the maximum independent set problem, using the graph construction described in (Garey and Johnson, 1979). Given a set of 3-element clauses $\mathbf{c}_1, \ldots, \mathbf{c}_M$ in N variables, one constructs a graph $G = (V, E)$. Here a clause $\mathbf{c}_i, i = 1, \ldots, M$ is represented with a triple of vertices $v^i_{\ell_{i1}}, v^i_{\ell_{i2}}, v^i_{\ell_{i3}}$, one vertex for each literal $\ell_{i1}, \ell_{i2}, \ell_{i3}$ of the clause. All vertices of a triple are connected to form a triangle, a *satisfaction testing component*. Additionally, for each variable there is a *truth-setting component* which consists of two vertices, connected by an edge. Two literals of variable $\xi_j, j = 1, \ldots, N$ are associated with vertices $v_{\xi_j}, v_{\bar{\xi}_j}$ in j-th truth-setting component. Finally, all vertices identified with a literal ℓ in the satisfaction testing components, are connected to the truth-setting vertex, corresponding to $\bar{\ell}$.

Here, the main difference from the textbook reduction is that in our case all vertices of the truth-setting components in graph $G = (V, E)$ are given weight M, the rest of the weights are equal to 1.

Now any truth assignment ξ for a MAX-3-SAT instance induces an independent set $S(\xi)$ of weight $NM + n_{\text{sat}}(\xi)$, where $n_{\text{sat}}(\xi)$ is the number of clauses satisfied by ξ. To obtain $S(\xi)$ one can start with a set $\{v_{\xi_j} : \xi_j = \text{true}\} \cup \{v_{\bar{\xi}_j} : \xi_j = \text{false}\}$ and add

an appropriate vertex from each satisfaction testing component that corresponds to a satisfied clause.

At the same time, any independent set with weight $NM + k,\ k \geq 0$ yields a truth assignment ξ with $n_{sat}(\xi) \geq k$. Obviously, all maximum-weight independent sets in G have a weight at least NM. So, solving the maximum-weight independent set problem on G is equivalent to solving the original MAX-3-SAT problem.

We can treat the independent sets of weight at least NM as feasible solutions to the MAX-3-SAT problem in the graph-based representation of solutions. Then the ORP for this NP maximization problem is efficiently solvable by Proposition 1. The general maximum satisfiability problem with arbitrary costs of the clauses may be reduced to the maximum weight independent set, using the same idea.

Note that in the framework of Proposition 2, the above transformation is not suitable for an efficient reduction from the ORP of the MAX-3-SAT with the standard representation ξ to the ORP of maximum independent set problem. This is consistent with the fact that the first ORP is NP-hard and the second one is polynomially solvable.

5 Conclusion

We have shown that optimal recombination may be efficiently carried out for a number of NP-hard optimization problems. The well-known reductions between the NP optimization problems turned out to be useful in development of polynomial-time optimal recombination procedures. We have observed that the choice of solutions encoding has a significant influence upon the complexity of the optimal recombination problems and often introduction of additional variables can simplify the task. The question of practical utility of such simplifications remains open, since the additional redundancy in representation increases the size of the search space. This trade-off may be studied in further research.

Another open question is related to the trade-off between the complexity of optimal recombination and its impact on the efficiency of an evolutionary algorithm. As it is indicated by Cotta (2003) and Borisovsky et al. (2007), the evolutionary algorithm may benefit from solving allelic dynastically optimal recombination problems, where only 0-valued genes are fixed, in spite of greater complexity of such problems.

All of the polynomially solvable cases of the optimal recombination problems considered above rely upon the efficient deterministic algorithms for the max-flow/min-cut problem (or the maximum matching problem in the unweighted case). However, the crossover operator was initially introduced as a randomized operator in genetic algorithms (Holland, 1975). As a compromise approach one can solve the optimal recombination problem approximately or solve it optimally but not in all occasions. Examples of the genetic algorithms using this approach may be found in (Borisovsky et al., 2007; Glover et al., 2000; Hohn and Reeves, 1996; Reeves, 1994).

In this paper, we did not discuss the issues of convergence of evolutionary algorithms when the optimal recombination is used. Due to fast localization of the search process in such heuristics it is often important to provide a sufficiently large initial population and employ some mechanism for adaptation of the mutation strength. Interesting techniques that maintain the diversity of population by constructing the second child, as different from the optimal offspring as possible, can be found in (Aggarwal et al., 1997) and (Balas and Niehaus, 1998). It is likely that the general schemes of the evolutionary algorithms and the procedures of parameter adaptation also require some revision when the optimal recombination is used.

References

Aggarwal, C., Orlin, J., and Tai, R. (1997). An optimized crossover for maximum independent set. *Operations Research*, 45:225–234.

Aho, A., Hopcroft, J., and Ullman, J. D. (1974). *The design and analysis of computer algorithms*. Addison-Wesley, Reading.

Alekseeva, E., Kochetov, Y., and Plyasunov, A. (2007). Complexity of local search for the p-median problem. *European Journal of Operational Research*. In Press.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., and Protasi, M. (1999). *Complexity and Approximation: Combinatorial optimization problems and their approximability properties*. Springer Verlag.

Balas, E. and Niehaus, W. (1996). Finding large cliques in arbitrary graphs by bipartite matching. In Johnson, D. and Trick., M., editors, *Series in Discrete Mathematics and Theoretical Computer Science*, volume 26, pages 29–49. AMS.

Balas, E. and Niehaus, W. (1998). Optimized crossover-based genetic algorithms for the maximum cardinality and maximum weight clique problems. *Journal of Heuristics*, 4(2):107–122.

Borisovsky, P., Dolgui, A., and Eremeev, A. (2007). Genetic algorithms for a supply management problem: MIP-recombination vs greedy decoder. *European Journal of Operational Research*. In Press.

Cheriyan, J., Cunningham, W., Tuncel, L., and Wang, Y. (1996). A linear programming and rounding approach to max 2-sat. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 26, pages 395–414.

Cotta, C. (2003). A study on allelic recombination. In *Proc. of 2003 Congress on Evolutionary Computation*, pages 1406–1413. IEEE Press, Canberra.

Cotta, C. and Troya, J. M. (2003). Embedding branch and bound within evolutionary algorithms. *Applied Intelligence*, 18:137–153.

Eremeev, A. (1999). A genetic algorithm with a non-binary representation for the set covering problem. In *Proc. of Operations Research (OR'98)*, pages 175–181. Springer Verlag.

Garey, M. and Johnson, D. (1979). *Computers and intractability. A guide to the theory of NP-completeness*. W.H. Freeman and Company.

Glover, F., Laguna, M., and Marti, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 29(3):653–684.

Guruswami, V., Hastad, J., and Sudan, M. (2002). Hardness of approximate hypergraph coloring. *SIAM Journal of Computing*, 31(6):1663–1686.

Hochbaum, D. S. (1997). Approximating covering and packing problems: Set cover, vertex cover, independent set, and related problems. In Hochbaum, D., editor, *Approximation Algorithms for NP-Hard Problems*, pages 94–143. PWS Publishing Company, Boston.

Hohn, C. and Reeves, C. R. (1996). Graph partitioning using genetic algorithms. In G.R.Sechi, editor, *Proc. of the 2nd International Conference on Massively Parallel Computing Systems*, pages 31–38. IEEE Computer Society Press, Los Alamitos, CA.

Holland, J. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.

Jansen, T. and Wegener, I. (2002). On the analysis of evolutionary algorithms - a proof that crossover really can help. *Algorithmica*, 34(1):47–66.

Krarup, J. and Pruzan, P. (1983). The simple plant location problem: survey and synthesis. *European Journal of Operational Research*, 12:36–81.

Lourenço, H., Paixão, J., and Portugal, R. (2001). Multiobjective metaheuristics for the bus driver scheduling problem. *Transportation Science*, 35:331–343.

Meyers, C. and Orlin, J. B. (2006). Very large-scale neighborhood search techniques in timetabling problems. In *Proc. of The 6th International Conference on the Practice and Theory of Automated Timetabling PATAT-2006*, pages 36–52.

Papadimitriou, C. H. and Steiglitz, K. (1982). *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall.

Radcliffe, N. J. (1991). Forma analysis and random respectful recombination. In *Proc. of the Fourth International Conference on Genetic Algorithms*, pages 31–38. Morgan Kaufmann.

Reeves, C. R. (1994). Genetic algorithms and neighbourhood search. In Fogarty, T., editor, *Evolutionary Computing, AISB Workshop. Selected Papers*, pages 115–130. Springer-Verlag, Berlin.

Schuurman, P. and Woeginger, G. (2006). Approximation schemes – a tutorial. Manuscript, to appear in *Lectures on Scheduling*, edited by R.H. Möhring, C.N. Potts, A.S. Schulz, G.J. Woeginger and L.A. Wolsey.

Yagiura, M. and Ibaraki, T. (1996). The use of dynamic programming in genetic algorithms for permutation problems. *European Journal of Operational Research*, 92:387–401.