Packing Triangles in Bounded Degree Graphs

Alberto Caprara[+] Romeo Rizzi[o]

[+] DEIS, University of Bologna
Viale Risorgimento 2, I-40136 Bologna, Italy
e-mail: acaprara@deis.unibo.it

[o] BRICS[*], Department of Computer Science, University of Aarhus
Ny Munkegade, DK-8000 Aarhus C, Denmark
e-mail: rrizzi@science.unitn.it

Abstract

We consider the two problems of finding the maximum number of node disjoint triangles and edge disjoint triangles in an undirected graph. We show that the first (resp. second) problem is polynomially solvable if the maximum degree of the input graph is at most 3 (resp. 4), whereas it is APX-hard for general graphs and NP-hard for planar graphs if the maximum degree is 4 (resp. 5) or more.

Key words: packing triangles, polynomial algorithm, APX-hardness, planar graphs, NP-hardness.

1 Introduction

The problem of finding the maximum number of node or edge disjoint cycles in an undirected graph G has several applications, for instance in computational biology [2]. It is often the case that the maximum degree of G and/or the length of the cycles to be found are bounded (e.g. both do not exceed 4) [2]. In this paper, we consider the problem of finding the simplest type of cycles, namely *triangles*, in graphs with bounded degree, considering both the requirement that the triangles be node disjoint and the requirement that they be edge disjoint. Addressing both the planar and nonplanar case, we determine the exact border between polynomial solvability and NP-hardness (APX-hardness for the nonplanar case) depending on the maximum degree of G.

Let $G = (V, E)$ be a simple graph. A *triangle* of G is any induced subgraph of G having precisely 3 edges and 3 nodes. A triangle $T = (\{a, b, c\}, \{ab, bc, ca\})$ will be often denoted by $[a, b, c]$. A family of triangles $T_1, \dots, T_k$ of G is called a *node-packing of triangles* if $T_1, \dots, T_k$ are node-disjoint and is called an *edge-packing of triangles* if $T_1, \dots, T_k$ are edge-disjoint. The *size* of the packing is equal to k. In this paper, we study the following two problems:

Problem 1 Node-Disjoint Triangle Packing (NTP) *Given a graph G, find a maximum size node-packing of triangles in G.*

Problem 2 Edge-Disjoint Triangle Packing (ETP) *Given a graph G, find a maximum size edge-packing of triangles in G.*

[*]Basic Research in Computer Science,
Centre of the Danish National Research Foundation.

It is known that both NTP [7] and ETP [8] are NP-hard. In particular, NTP is known to be NP-hard also in the planar case. Moreover, NTP is known to be APX-hard [10] even for graphs with maximum degree 4 [1]. As to the approximability of the two problems, a general result of [9] leads to a polynomial time $(3/2+\varepsilon)$-approximation algorithm (for any $\varepsilon > 0$) for both problems, which is the best approximation guarantee known so far. For the planar case, [4] presents a polynomial time approximation scheme for NTP, which can be extended to handle ETP as well [5].

As customary, we let $\delta(v)$ denote the set of edges incident with $v \in V$ and $\Delta(G) := \max_{v\in V} |\delta(v)|$. Given a maximization problem P, let $opt_P(I)$ denote the optimal solution value for some instance I of P and, for a solution S of I, let $val_P(I,S)$ denote the associated value. Given a constant $\varepsilon \in (0,1)$, a $\frac{1}{1-\varepsilon}$-*approximation algorithm* for P is an algorithm that, applied to any instance I of P, runs in time polynomial in the size of I and produces a solution whose value is at least $(1-\varepsilon)\cdot opt_P(I)$. If such an algorithm exists, P belongs to APX. Moreover, P is said to be APX-*hard* if the existence of a $\frac{1}{1-\varepsilon}$-approximation algorithm for P for *any* $\varepsilon \in (0,1)$ would imply the existence of a $\frac{1}{1-\delta}$-approximation algorithm for any $\delta \in (0,1)$ for *all* problems in APX. To show that P is APX-hard, it sufficies to show a special type of polynomial time reduction from some problem Q already known to be APX-hard to P. The type of reduction used most frequently is the L-reduction [1]. An *L-reduction* from Q to P consists of a pair of polynomial-time computable functions (f,g) such that, for two fixed constants α and β: (a) f maps input instances of Q into input instances of P; (b) given a Q-instance I, the corresponding P-instance $f(I)$, and any feasible solution S for $f(I)$, $g(I,S)$ is a feasible solution for the Q-instance I; (c) $|opt_P(f(I))| \leq \alpha |opt_Q(I)|$ for all I, and (d) $|opt_Q(I) - val_Q(I, g(I,S))| \leq \beta |opt_P(f(I)) - val_P(f(I),S)|$ for each I and for every feasible solution S for $f(I)$. From this definition it follows that the relative errors are linearly related, i.e.

$$\frac{|opt_Q(I) - val_Q(I,g(I,S))|}{opt_Q(I)} \leq \alpha\,\beta\,\frac{|opt_P(f(I)) - val_P(f(I),S)|}{opt_P(f(I))}.$$

Hence, if both Q and P are maximization problems, the existence of a $\frac{1}{1-\varepsilon}$-approximation algorithm for P implies the existence of a $\frac{1}{1-\alpha\beta\varepsilon}$-approximation algorithm for Q.

In this paper, we will show the complexity of NTP and ETP depending on $\Delta(G)$. Specifically, in Section 2 we prove that NTP (resp. ETP) can be solved in polynomial time if $\Delta(G) \leq 3$ (resp. $\Delta(G) \leq 4$), whereas in Section 3 we show that it is APX-hard for $\Delta(G) = 4$ (resp. $\Delta(G) = 5$), and in Section 4 we show that it is NP-hard if G is planar and $\Delta(G) = 4$ (resp. $\Delta(G) = 5$). As mentioned above, the APX-hardness of NTP for $\Delta(G) = 4$ was already known. Generally, the results for ETP in this paper are slightly more elaborated than the corresponding results for NTP. Therefore, for the sake of presentation, we will show the results for NTP first.

2 Polynomial solvability

In general, when G is not connected, both problems naturally decompose onto the connected components of G. Another natural reduction for both NTP and ETP consists of deleting the edges which do not belong to any triangle. Hence, we restrict our attention to *reduced* graphs G, that are connected and where every edge belongs to some triangle.

Proposition 2.1 NTP *is polynomially solvable for* $\Delta(G) \leq 3$.

Proof: Consider a reduced graph G. If $\Delta(G) = 2$, then G is a triangle. We next show that, if $\Delta(G) = 3$, then G contains 4 nodes, being either a K_4 or a diamond (a K_4 without an edge), completing the proof.

Let v be a node of degree 3 and let v_1, v_2, v_3 be the neighbors of v. Since vv_1, vv_2 and vv_3 are all contained into some triangle, and since every triangle containing vv_i $(i = 1, 2, 3)$ must contain precisely one of the edges in $\delta(v) \setminus \{vv_i\}$, then we can assume w.l.o.g. that v_1v_2 and v_2v_3 both belong to $E(G)$. Now two cases are possible. If v_1v_3 also belongs to $E(G)$, then G is a K_4, since no more edges can depart from the four nodes considered. Otherwise, v_1 (and, by symmetry, v_3), has degree 2 in G. Indeed, for any further node z of G, if v_1z were in $E(G)$, then no triangle of G could contain it, since nodes v and v_2 already have three neighbors other than z. $\square$

In order to extend the above result for ETP, we need one more reduction that is valid only for this problem. Consider a node v and assume $\delta(v)$ can be partitioned as $L \cup R$ so that no triangle of G contains both an edge in L and an edge in R. The reduction consists of replacing node v with two new nodes v_L and v_R, with $\delta(v_L) := \{v_Lu : vu \in L\}$ and $\delta(v_R) := \{v_Rw : vw \in R\}$. We call this reduction *splitting on* v, and call *edge reduced* a graph G which is reduced and for which no splitting can be performed.

Proposition 2.2 ETP *is polynomially solvable for* $\Delta(G) \leq 4$.

Proof: Consider an edge reduced graph G. The case $\Delta(G) \leq 3$ is identical to the node disjoint case addressed in the proof of Proposition 2.1. Assume therefore $\Delta(G) = 4$.

We define the *auxiliary graph* $A_G = (\mathcal{T}, F)$, containing one node for each triangle of G, and where two nodes are adjacent if and only if the corresponding triangles share an edge in G. Finding a maximum independent set in A_G is the same as solving ETP on G. We next show that A_G is either *claw free* or contains a constant number of nodes. A *claw* is a node induced subgraph with nodes T, T_1, T_2, T_3 and edges TT_i for $i = 1, 2, 3$. Since maximum independent set can be solved in polynomial time for claw-free graphs [12], the proof follows.

Let node $T \in \mathcal{T}$ and its neighbors T_1, T_2 and T_3 induce a claw in A_G. The corresponding situation in G is the following: $T = [v_1, v_2, v_3]$, $T_1 = [w_1, v_2, v_3]$, $T_2 = [v_1, w_2, v_3]$, $T_3 = [v_1, v_2, w_3]$. We claim that, G contains only nodes $v_1, v_2, v_3, w_1, w_2, w_3$. Indeed, nodes v_1, v_2, v_3 already have degree 4 in $\bar{G}$ and any possible edge w_1x with $x \notin \{v_1, v_2, v_3, w_1, w_2, w_3\}$ can not be in a triangle along with w_1v_2 or along with w_1v_3. $\square$

3 APX-hardness for nonplanar graphs

For the APX-hardness proofs, the reductions are from Max-2-Sat-3, and their structure is similar to the reductions in [7] and [10] to prove the NP- and APX-hardness of NTP (see also [6]). In particular, in this section and the next one the general idea of the reductions from a satisfiability problem is fairly simple and always the same. The only things that change and have to be defined with some care are the gadgets associated with clauses and boolean variables. As already mentioned, Proposition 3.1 below is already proved in [10]. However, for the sake of presentation, we give an explicit proof here, since this proof is similar to (and simpler than) those of the other results in this section and the next one.

The input to MAX-2-SAT-3 is a boolean formula φ in conjunctive normal form with clauses $C := \{c_1, \ldots, c_m\}$, in which each clause is the OR of at most 2 literals. Each literal is a variable or the negation of a variable taken from a ground set of boolean variables $X := \{x_1, \ldots, x_n\}$, with the additional restriction that each variable appears in at most 3 of the clauses, counting together both positive and negative occurrences. MAX-2-SAT-3 calls for a truth assignment that satisfies as many clauses as possible. It is known that MAX-2-SAT-3 is APX-hard [1, 3]. We let m_i denote the number of occurrences of x_i. W.l.o.g. we can assume $2 \le m_i \le 3$, for if x_i appears only in one clause we can set it to the value which satisfies the clause.

Proposition 3.1 NTP *is APX-hard, even for* $\Delta(G) = 4$.

Proof: We will show how to transform a MAX-2-SAT-3 instance φ into a graph $G(\varphi)$ in such a way that every truth assignment for φ that satisfies k clauses can be transformed into a packing of $G(\varphi)$ of value $\sum_{i=1}^{n} m_i + k$, and viceversa. Noting that $\sum_{i=1}^{n} m_i = 2m$ and the optimal MAX-2-SAT-3 value is at least $\frac{m}{2}$, since at least half of the clauses can be satisfied by a simple greedy approach, shows that this is an L-reduction with $\alpha = 5$ and $\beta = 1$. This implies that any $(\frac{1}{1-\varepsilon})$-approximated solution of NTP on $G(\varphi)$ can be transformed into a $(\frac{1}{1-5\varepsilon})$-approximated solution of MAX-2-SAT-3 on φ.

To each clause c_j we associate a *test component*, shown in Fig. 1. The left-hand side shows the test component when the clause has two literals, while the right-hand side shows it when the clause has one literal. The test component of a clause with two literals consists of two *triangles* $[s_j^1, r_j^1, t_j^1]$ and $[s_j^2, r_j^1, t_j^2]$ with node r_j^1 in common. The test component associated with a clause with one literal consists of a single triangle $[s_j^1, r_j^1, t_j^1]$.

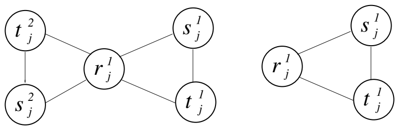

Figure 1: To the left, the test component associated with the clause c_j when it has two literals and, to the right, when it has one literal, in the reduction of Proposition 3.1.

To each variable x_i is associated a *truth setting component* depending on m_i, shown in Fig. 2. The component consists of $2m_i$ triangles T_j, where, for $j = 1, \ldots, m_i$, $T_{2j-1} = [a_i^j, v_i^{j-1}, u_i^j]$ and $T_{2j} = [b_i^j, u_i^j, v_i^j]$, all indices being modulo m_i. The *parity* of T_j is simply the parity of j. Clearly, at most half of the triangles in this component can belong to a same packing, and this is possible only if they have all the same parity.

The graph $G(\varphi)$ is obtained by connecting test and truth setting components as follows. Let c_j be a clause with two literals and let x_1, x_2 be the variables which occur in c_j. If x_i occurs positive (resp. negated) in c_j, then identify node t_j^i of the test component with a node a_i^k (resp. b_i^k) of the truth setting component of x_i which has not yet been involved in any identification. Similarly, let c_j be a clause with one literal x_1. If x_1 occurs positive (resp. negated) in c_j, then identify node t_j^1 of the test component with node a_1^k (resp. b_1^k) of the

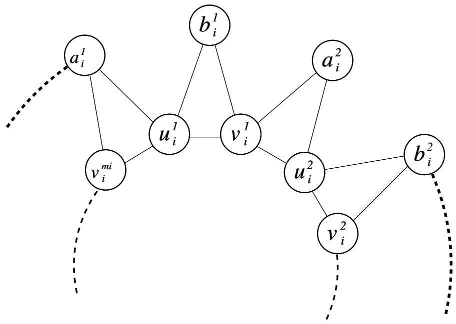

Figure 2: The truth setting component associated with variable x_i in the reduction of Proposition 3.1.

truth setting component of x_1 which has not yet been involved in any identification. This completes the construction of $G(\varphi)$. Note that $\Delta(G(\varphi)) = 4$.

A maximal packing $\mathcal{P}$ is called *canonical* if, for each truth setting component, it contains either all even triangles or all odd triangles of the component. A canonical packing $\mathcal{P}$ naturally corresponds to a truth assignment in the following way. If $\mathcal{P}$ contains a triangle of the test component of c_j this means that c_j is "satisfied" by $\mathcal{P}$. If $\mathcal{P}$ contains all even (resp. odd) triangles of the truth setting component of x_i this means that x_i is "set to true (resp. false)" by $\mathcal{P}$. Therefore a canonical packing of $G(\varphi)$ containing $\Sigma_{i=1}^{n} m_i + k$ triangles corresponds to a truth assignment of φ satisfying exactly k clauses, and viceversa.

The proof is concluded by showing that, given a packing $\mathcal{P}$, we can find in polynomial time a canonical packing which is at least as large. Indeed, focus on a truth setting component C of $\mathcal{P}$ associated with variable x_i. There are two cases to consider. First, if all triangles of C that are in the packing have the same parity, say even, then adding to the packing all even triangles in C that are missing, and possibly removing the triangles corresponding to x_i in the test components in which x_i appears negated, yields a packing which is at least as large as $\mathcal{P}$. Otherwise, the packing contains both even and odd triangles. We consider the case $m_i = 3$ (the case $m_i = 2$ being analogous and simpler). Suppose w.l.o.g. x_i appears negated in clause c_1 and positive in clauses c_2 and c_3. Adding to the packing all even triangles in C that are missing and removing all odd triangles and, possibly, the triangle corresponding to x_i in the test component of clause c_1, yields a packing which is at least as large as $\mathcal{P}$. Note that in this case it is crucial that $m_i \leq 3$ for all i. $\square$

Proposition 3.2 ETP *is APX-hard, even for* $\Delta(G) = 5$.

Proof: Analogous to the proof of Proposition 3.1, changing the definition of G. In particular, we will show an L-reduction with $\alpha = 21$ and $\beta = 1$. The test component associated with each clause c_j is shown in Fig. 3. Specifically, the test component of a clause with two literals consists of two *triangles* $[t_j^1, s_j^1, s_j^2]$ and $[t_j^2, s_j^1, s_j^2]$ with edge $s_j^1 s_j^2$ in common, whereas the test component associated with a clause with one literal consists of a single triangle $[t_j^1, s_j^1, r_j^1]$.

The truth setting component associated with each variable x_i is shown in Fig. 4. This component could be slightly simplified, however we present this version as it can be used

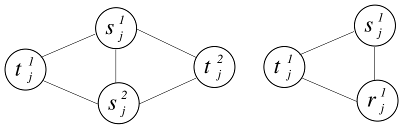

Figure 3: To the left, the test component associated with the clause c_j when it has two literals and, to the right, when it has one literal, in the reduction of Proposition 3.2.

also for the proof of Proposition 4.2. The component consists of $10m_i$ triangles T_j, where, for $j = 1, \ldots, m_i$, $T_{10j-9} = [a_i^j, z_i^{j-1}, u_i^j]$, $T_{10j-8} = [a_i^j, b_i^j, u_i^j]$, $T_{10j-7} = [b_i^j, c_i^j, u_i^j]$, $T_{10j-6} = [c_i^j, u_i^j, v_i^j]$, $T_{10j-5} = [c_i^j, d_i^j, v_i^j]$, $T_{10j-4} = [d_i^j, v_i^j, y_i^j]$, $T_{10j-3} = [d_i^j, e_i^j, y_i^j]$, $T_{10j-2} = [e_i^j, f_i^j, y_i^j]$, $T_{10j-1} = [f_i^j, y_i^j, z_i^j]$, $T_{10j} = [f_i^j, a_i^{j+1}, z_i^j]$, all indices being modulo m_i. As before, the parity of T_j is simply the parity of j. Moreover, at most half of the triangles in this component can belong to a same packing, and this is possible only if they have all the same parity.

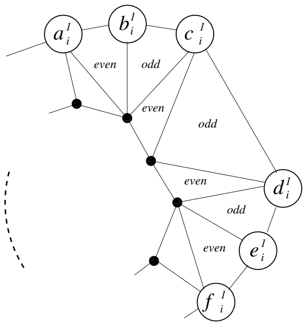

Figure 4: The truth setting component associated with variable x_i in the reduction of Proposition 3.2.

The graph $G(\varphi)$ is obtained by connecting test and truth setting components as follows. Let c_j be a clause with two literals and let x_1, x_2 be the variables which occur in c_j. If x_i occurs positive (resp. negated) in c_j, then identify nodes s_j^i and t_j^i of the test component with nodes b_i^k and c_i^k (resp. e_i^k and f_i^k) of the truth setting component of x_i which have not yet been involved in any identification. Similarly, let c_j be a clause with one literal x_1. If x_1 occurs positive (resp. negated) in c_j, then identify nodes s_j^1 and t_j^1 of the test component with nodes b_1^k and c_1^k (resp. e_1^k and f_1^k) of the truth setting component of x_1 which have not yet been involved in any identification. This completes the construction of $G(\varphi)$. Note that $\Delta(G(\varphi)) = 5$.

The proof follows by observing that every truth assignment for φ that satisfies k clauses can be transformed into a packing of $G(\varphi)$ of value $\sum_{i=1}^{n} 5m_i + k$, and viceversa. $\square$

4 NP-hardness for planar graphs

As in the previous section, let $X = \{x_1, \ldots, x_n\}$ and $C = \{c_1, \ldots, c_m\}$ denote, respectively, the set of variables and clauses in a boolean formula φ in conjunctive normal form, where now each clause has *exactly* 3 literals. Consider the bipartite graph $B_\varphi = (X \cup C, E_\varphi)$, with color classes X and C and edge set $E_\varphi = \{xc$: variable x occurs in clause $c\}$. The boolean formula φ is called *planar* when B_φ is planar. PLANAR 3-SAT is the problem of finding, if any, a truth assignment that satisfies all clauses in a planar boolean formula, where each clause has exactly three literals. It is known that PLANAR 3-SAT is NP-complete [11].

Proposition 4.1 NTP *is NP-hard for planar graphs, even for* $\Delta(G) = 4$.

Proof: Follows the same lines as that of Proposition 3.1, transforming a PLANAR 3-SAT instance φ to a graph $G(\varphi)$ in polynomial time by connecting certain gadgets together. The truth setting component associated with variable x_i is the same as in the reduction of Proposition 3.2, displayed in Fig. 2, noting that in this case m_i may be arbitrarily large. The test component associated with a clause c_j is displayed in Fig. 5. The property of this component is that any maximal packing of triangles contains either one or two triangles in the component, and in the latter case it contains at least one triangle with a node among t_j^1, t_j^2, t_j^3. This component is connected to truth setting components by identifying the nodes t_j^1, t_j^2, t_j^3 with nodes a_i^q or b_i^q, as in the reduction of Proposition 3.1.

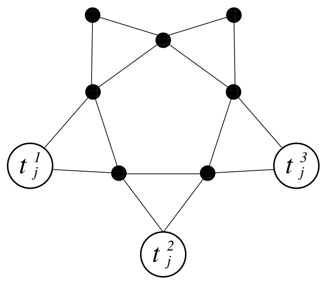

Figure 5: The test component associated with the clause c_j in the reduction of Proposition 4.1.

In this fashion we obtain a graph $G(\varphi)$ in polynomial time, with $\Delta(G(\varphi)) = 4$. Because of the freedom in making the connections, $G(\varphi)$ is not uniquely defined. However, it is always possible to perform the above edge identifications in such a way that $G(\varphi)$ is planar. Considerations analogous to those in the proof of Proposition 3.1 show that $G(\varphi)$ has a packing of triangles of size $\sum_{i=1}^n m_i + 2m$ if and only if φ is satisfiable, yielding the proof. □

Proposition 4.2 ETP *is NP-hard for planar graphs, even for* $\Delta(G) = 5$.

Proof: Analogous to that of Proposition 4.1. The truth setting component associated with variable x_i is the same as in the reduction of Proposition 3.2, see Fig. 4, noting again that m_i may be arbitrarily large. The test component associated with a clause c_j is displayed in Fig. 6. It is not difficult to check that there is an optimal packing of triangles that, for each test component C_j, contains one triangle visiting node s^i_j for $i = 1, 2, 3$. If at least one of these triangles visits also node t^i_j, then the packing contains 5 additional triangles in C_j, otherwise the packing contains 4 additional triangles in C_j. This component is connected to truth setting components by identifying nodes s^1_j, s^2_j, s^3_j with nodes b^k_i or e^k_i, and nodes t^1_j, t^2_j, t^3_j with nodes c^k_i or f^k_i, as in the reduction of Proposition 3.2.

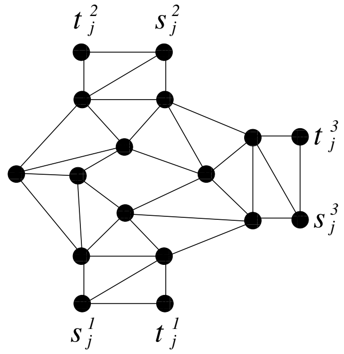

Figure 6: The test component associated with the clause c_j in the reduction of Proposition 4.2.

Note that we can guarantee that $G(\varphi)$ is planar (this motivates the structure of the component in Fig. 4), and that $\Delta(G(\varphi)) = 5$. $G(\varphi)$ has a packing of triangles of size $\sum_{i=1}^{n} 5m_i + 8m$ if and only if φ is satisfiable, yielding the proof. $\square$

Acknowledgments

Part of this work was done while the first author was visiting BRICS. The first author was also partially supported by CNR and MURST, Italy. We are grateful to Brenda Baker, Viggo Kann and Alessandro Panconesi for helpful discussions on the subject.

References

[1] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protasi, *Combinatorial Optimization Problems and their Approximability Properties*, Springer-Verlag, Berlin (1999).

[2] V. Bafna and P.A. Pevzner, Genome rearrangements and sorting by reversals. SIAM Journal on Computing, 25 (1996) 272–289.

[3] P. Berman and M. Karpinski, On some tighter inapproximability results. ECCC Report No. 29 (1998), University of Trier, 1998.

[4] B.S. Baker, Approximation algorithms for NP-complete problems on planar graphs. Journal of the ACM, 41 (1994) 153–180.

[5] B.S. Baker, Personal communication (2001).

[6] A. Caprara, A. Panconesi and R. Rizzi, Packing Cycles in Undirected Graphs. Working Paper, University of Bologna (2001).

[7] M.R. Garey and D.S. Johnson, Computers and intractability: a guide to the theory of NP-completeness. Freeman, San Francisco (1979).

[8] I. Holyer, The NP-completeness of some edge-partition problems. SIAM Journal on Computing, 10 (1981) 713–717.

[9] C.A.J. Hurkens and A. Schrijver, On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. SIAM Journal on Discrete Mathematics, 2 (1989) 68–72.

[10] V. Kann, Maximum bounded 3-dimensional matching is MAX SNP-complete. Information Processing Letters 37 (1991) 27–35.

[11] D. Lichtenstein, Planar formulae and their uses. SIAM Journal on Computing, 11 (1982) 329–343.

[12] G.J. Minty, On maximal independent sets of vertices in claw-free graphs. Journal of Combinatorial Theory Series B, 28 (1980) 284–304.

[13] T.J. Schaefer, The complexity of satisfiability problems. Proceedings of the 10th Annual ACM Symposium on Theory of Computing, ACM, New York (1978) 216–226.