Electronic Colloquium on Computational Complexity, Report No. 16 (2006)

Partition into k-vertex subgraphs of k-partite graphs

Tomás Feder
268 Waverley St., Palo Alto, CA 94301, USA
tomas@theory.stanford.edu
and
Carlos Subi
27-024 Okana Place, Kaneohe, HI 96744, USA
carlos_subi@hotmail.com

Abstract

The H-matching problem asks to partition the vertices of an input graph G into sets of size $k = |V(H)|$, each of which induces a subgraph of G isomorphic to H. The H-matching problem has been classified as polynomial or NP-complete depending on whether $k \leq 2$ or not. We consider a variant of the problem, in which a homomorphism from G to H is given, so that G is k-partite, and each chosen set of size k must contain exactly one vertex from each of the k parts. We classify the problem as polynomial or NP-complete depending on whether H is a forest or not.

The polynomial case with H a forest generalizes to the case where each set of size k must contain a subgraph satisfying certain degree constraints, provided that the skip between consecutive allowed degrees is at most two; a skip of at least three gives NP-completeness. More generally, one may specify which sets of incident edges for a vertex in the subgraph are allowed, and the problem has complexity related to delta-matroids. Several of the polynomial cases extend to a weighted version.

An optimization variant of the problem asks to maximize the number of chosen sets of size k that induce a subgraph isomorphic to H. This problem is shown polynomial if every component of H is a path, and NP-complete otherwise. The polynomial cases extend to a weighted version, while the NP-complete cases are hard to approximate within a constant even for bounded degree instances, allowing a $\frac{k}{2} + \epsilon$ approximation. We similarly classify the problem of asking that each chosen set of size k contain at least r edges forming a connected

ISSN 1433-8092

subgraph, for all H and r, and nearly classify the case where the r edges are not required to form a connected subgraph.

1 Introduction

Kirkpatrick and Hell [16] considered the problem of partitioning a graph into isomorphic subgraphs. For a fixed graph H with $|V(H)| = k$, and an input graph G with $|V(G)| = kl$, the problem asks to partition $V(G)$ into l disjoint sets S_i with $|S_i| = k$, such that the subgraph of G induced by S_i is isomorphic to H. They showed that the problem is NP-complete for any H with $k = |V(H)| \geq 3$, where H is not necessarily connected. The analogous problem in which the subgraph induced by S_i need only have $k = |V(H)|$ vertices and contain a subgraph isomorphic to H is also NP-complete for any H that contains a connected component of three or more vertices. Both problems can be solved in polynomial time by matching for any H not meeting the stated restrictions.

An optimization version of the problem asks to maximize the number of sets S_i that induce a subgraph isomorphic to H. While the polynomial cases remain polynomial, the NP-complete cases are hard to approximate within some constant $c > 1$, even for instances degree bounded by some constant d, as shown by Kann [15]; the degree bound for hardness of approximation is $d = 4$ when H is a triangle, as shown by Berman and Fujito [4]. The best known approximation bounds are $\frac{k}{2} + \epsilon$ for all $\epsilon > 0$, as shown by Hurkens and Schrijver [13], and the bounds for a weighted version of the problem are $k - 1 + \epsilon$, as shown by Arkin and Hassin [1], and $\frac{2(k+1)}{3}$, as shown by Chandra and Halldórsson [5]. A weighted version with H a path with three edges has a $\frac{4}{3}$ approximation, see Hassin and Rubinstein [12]. By contrast, the cases where H and G are required to be planar admit a polynomial time approximation scheme, as shown by Baker [2], but not a fully polynomial approximation scheme, as shown by Berman et al [3].

We shall study here a restricted version of this problem, in which we are given in addition a homomorphism f from G to H, and the problem asks to partition $V(G)$ into l disjoint sets S_i with $|S_i| = k$ such that the homomorphism f restricted to each S_i is a one-to-one correspondence from S_i to H. We study again the basic problem where each S_i must induce a subgraph of G isomorphic to H; note that in that case an isomorphism will be given by the restriction of f to S_i, and that without the isomorphism condition f is always isomorphism from each S_i to a subgraph of H. We show that this problem can be solved in polynomial time if H is a forest, and is NP-complete otherwise. The case where H is a forest remains polynomial when edges have weights and we wish to maximize the sum of the weights in the copies S_i of H over valid partitions.

The polynomial case with H a forest generalizes to the case where each set of size k must contain a subgraph satisfying certain degree constraints, provided that the skip between consecutive allowed degrees is at most two; a skip of at least three gives NP-completeness. More generally, one may specify which sets of incident edges for a vertex in the subgraph are allowed, and the problem is polynomial or NP-complete for upward closed allowed edge sets depending on whether the specified edge sets are delta-matroids or not. The case of upward closed delta-matroids remains polynomial when edges have weights and we wish to maximize the sum of the weights in the subsets S_i over valid partitions.

The optimization variant of the problem asks to maximize the number of chosen sets of size k that induce a subgraph isomorphic to H.This problem is shown polynomial if every component of H is a path, and NP-complete otherwise. The polynomial cases extend to a weighted version where vertices and edges have weights, while as before the NP-complete cases are hard to approximate within some constant $c > 1$ even for bounded degree instances, allowing a $\frac{k}{2} + \epsilon$ approximation in the unweighted case, and both a $k - 1 + \epsilon$ and a $\frac{2(k+1)}{3}$ approximation in the weighted case.

We finally classify the problem of asking that each chosen set of size k contain at least r edges forming a connected subgraph, for all H and r, and nearly classify the case where the r edges are not required to form a connected subgraph. In both cases, for H connected and $1 < r < |E(H)|$, the problem is polynomial is H is a star and NP-complete otherwise. When H is not connected, the possible polynomial cases for both problems involve components that are paths or stars.

2 The Basic Problem

We first characterize the basic problem. Recall that we are given a homomorphism f from the input graph G to the fixed graph H with $|V(H)| = k$, and wish to partition the vertices of G into l sets S_i with $|S_i| = k$ such that f is a one-to-one correspondence from S_i to H.

Theorem 2.1 *The basic problem where each S_i must induce a subgraph of G isomorphic to H can be solved in polynomial time by matching if H is a forest, and is NP-complete otherwise. The polynomial case with H a forest* *remains polynomial when edges have weights and the aim is to find a valid partition that maximizes the total weight of edges with endpoints in the same* S_i*.*

Proof. Suppose H is a forest. For every edge uu' in H, there are l edges vv' in G with $f(v) = u$, $f(v') = u'$ that must be chosen, one for each S_i. Thus these edges vv' must form a perfect matching of the bipartite subgraph of G induced by the vertices w such that $f(w) = u$ or $f(w) = u'$. One can find these perfect matchings in G for each edge uu' in H, if they exist, in polynomial time. The subgraph of G corresponding to the union of these perfect matchings has as connected components l copies of each connected component K of H, where K is a tree, and where f is an isomorphism from each of these l copies to K. We may then combine the l copies of each connected component K to obtain l copies of H, where f is an isomorphism from each of these l copies S_i to H. The weighted version of the problem can also be solved in polynomial time by a weighted matching algorithm that finds perfect matchings of maximum edge weight.

Suppose H is not a forest. Let H' be the shortest cycle in H. We reduce the problem for H' to the problem for H. Given an instance G' for H' with $k'l$ vertices, add the remaining $(k - k')l$ vertices to G' to obtain an instance G for H, including also for every edge uu' in H and not in R all the edges vv' in G for v, v' such that $f(v) = u$, $f(v') = u'$. Any decomposition of G into l copies S_i of H such that f is an isomorphism from S_i to H will give l copies S'_i of the cycle H' such that f is an isomorphism from S'_i to H'. Conversely, any decomposition of G' into l copies S'_i of the cycle H' such that f is an isomorphism from S'_i to H' can be extended by arbitrarily assigning the l vertices v such that $f(v) = u$ with u not in the cycle H' to the l copies S'_i of H' to obtain l copies S_i of H such that f is an isomorphism from S_i to H.

Given a cycle H' of length k', let H'' be a triangle. We reduce the problem for H'' to the problem for H'. Given an instance G'' for H'' with $3l$ vertices, select a vertex u in H'', and replace u with a path p of length $k' - 3$ to obtain the cycle H' of length k'. Replace correspondingly each of the l vertices v in G'' such that $f(v) = u$ with a path p_v of length $k' - 3$ that maps to p by an extension of f, to obtain a graph G' with $k'l$ vertices that maps to the cycle H' by f. Any decomposition of G' into l copies S'_i of H' such that f is an isomorphism from S'_i to H' will have the l paths p_v in the l copies S'_i, so after contracting the l paths p_v we obtain l triangles S''_i that map to H'' by an isomorphism. Conversely, any decomposition of G'' into l triangles S''_i such that f is an isomorphism from S''_i to H'' gives a decomposition of G' into l cycles S'_i with an isomorphism f from S'_i to H'

after replacing each vertex v with $f(v) = u$ by the corresponding path p_v.

It remains to show NP-completeness for the case where H is a triangle. An instance of the 3-dimensional matching problem has three sets V_1, V_2, V_3 of size l and a collection E of triples (v_1, v_2, v_3) with v_i in V_i. The aim is to partition $V_1 \cup V_2 \cup V_3$ into l triples (v_1, v_2, v_3) from E. The 3-dimensional matching problem is NP-complete [11]. Replace each triple (v_1, v_2, v_3) from E with a graph R consisting of four triangles $a_1a_2a_3$, $b_1b_2b_3$, $c_1c_2c_3$, $a_1b_2c_3$, plus the edges $v_1a_2, v_1a_3, v_2b_1, v_2b_3, v_3c_1, v_3c_2$. In the resulting graph G, each copy of R is connected to the rest of the graph only through v_1, v_2, v_3. Define a mapping f from G to the triangle H with vertices u_1, u_2, u_3 by mapping all vertices in G indexed by i to u_i. All triangles in G are within each of the copies of R, and to cover all the vertices a_i, b_i, c_i in R we must either select the three triangles $a_1a_2a_3$, $b_1b_2b_3$, $c_1c_2c_3$, or the four triangles $a_1b_2c_3$, $v_1a_2a_3$, $b_1v_2b_3$, $c_1c_2v_3$. Selecting the three triangles corresponds to not selecting the triple (v_1, v_2, v_3) in E since the vertices v_i do not appear in the three triangles, while selecting the four triangles corresponds to selecting the triple (v_1, v_2, v_3) in E since the three vertices v_i do appear in the four triangles, completing the reduction. □

3 Degree Constraints, Optimization, Matroids

We now weaken the condition that each S_i must induce a subgraph of G isomorphic to H, and consider a degree constrained problem. We assume that each vertex v of an instance G is given a nonempty set of integers $D_v \subseteq \{0, 1, \ldots, r_v\}$, where r_v is the degree of $f(v)$ in H. We shall require that the subgraph of G induced by each S_i contain a subgraph T_i with $V(T_i) = S_i$ such that the degree in T_i of a vertex v belongs to D_v.

The *skip* of D_v is the largest integer $s \geq 0$ such that D_v contains two integers t and $t + s$ but no integer x with $t < x < t + s$.

Theorem 3.1 *If H is a forest, and the skip of each D_v is at most 2, then this degree constrained problem can be solved in polynomial time via a general factor algorithm from [6].*

Proof. An instance of the general factor problem is a graph G' in which each vertex v is assigned a set $D'_v \subseteq \{0, 1, \ldots, d_v\}$ with skip at most 2, where d_v is the degree of v in G'. The aim is to select a set of edges $F' \subseteq E(G')$ such that each vertex v in G' is incident to f_v edges in F' for some $f_v \in D'_v$. Cornuejols [6] gave a polynomial time algorithm for the general factor problem.

Given an instance G of the degree constrained problem for a forest H, define an instance G' of the general factor problem by replacing each vertex v in G such that $f(v) = u$ with $r_v + 1$ vertices $a_v, b_{v,u'}$ for each neighbor u' of u in H, with r_v edges $a_v b_{v,u'}$, and an edge $b_{v,u'}, b_{v',u}$ for each edge vv' in G such that $f(v) = u$, $f(v') = u'$. Let $D'_{a_v} = D_v$, and let $D'_{b_{v,u'}} = \{d_{v,u'} - 1\}$, where $d_{v,u'}$ is the degree of $b_{v,u'}$.

Solve the general factor problem for G'. If G' has the edges $a_v b_{v,u'}$, $a_{v'}, b_{v',u}$, $b_{v,u'}, b_{v',u}$, with the first two edges in F' and the third edge not in F', then include the edge vv' in F. This gives a set of edges $F \subseteq E(G)$ such that the degree of v in F is the same as the degree of a_v in F', and this degree is in $D_v = D'_{a_v}$. Furthermore, if $f(v) = u$ and uu' is an edge of H, then F contains at most one edge vv' with $f(v') = u'$.

Select now a vertex u of H as the root, and place the l vertices v with $f(v) = u$ in the l copies S_i. If v is thus placed in some S_i, then place any v' that is in the same connected component of F as v in the same S_i. Repeat this similarly for each child u' of u in H, placing the vertices v' with $f(v') = u'$ in the copies S_i into which no v' with $f(v') = u'$ was previously placed. Repeat again for each child u'' of each child u', and so on, until every vertex u in H has been considered and every vertex v in G has been placed in some S_i. As a result, all the edges in F join two vertices in the same S_i, and we may let T_i be the subgraph with vertices S_i and having the edges from F joining two vertices in S_i, thus giving a solution to the degree constrained problem.

Note that any solution for the degree constrained problem for G consisting of sets S_i and subgraphs T_i corresponds in the way just described to a solution to the general factor problem for G', by letting F be the edges of all the T_i, and letting F' contain all edges $a_v b_{v,u'}$ such that F has an edge vv' with $f(v') = u'$, and all edges $b_{v,u'} b_{v',u}$ such that vv' is not in F'. □

Theorem 3.2 *If H is a fixed forest containing a vertex u of degree at least 3, each vertex v with $f(v) = u$ is assigned the same fixed set $D = D_v$ of skip at least 3, and for all other vertices w the set D_w is either $\{0\}$ or $\{1\}$, then the degree constrained problem is NP-complete.*

Proof. An instance of the r-dimensional matching problem has r sets $V_1, \ldots, V_r$ of size l and a collection E of tuples $(v_1, \ldots, v_r)$ with v_i in V_i. The aim is to partition $V_1 \cup \cdots \cup V_r$ into r tuples $(v_1, \ldots, v_r)$ from E. The r-dimensional matching problem is NP-complete for $r \geq 3$ [11].

If D has skip $r \geq 3$, consider an instance of the r-dimensional matching. Note that D contains two integers t and $t+r$ but no integer x with $t < x < t+r$. Let $u_1, \ldots, u_r$ be r neighbors of u, and let the set of $|E|$ vertices v_i with $f(v_i) = u_i$ be the set V_i augmented with some additional $|E| - l$ vertices. The vertices v_i from V_i have $D_{v_i} = \{1\}$, and the remaining $|E| - l$ vertices v_i have $D_{v_i} = \{0\}$. Let the set of vertices v with $f(v) = u$ correspond to the $|E|$ tuples in E, and if v corresponds to $(v_1, \ldots, v_r)$ then include the edges vv_i in G. Let $u'_1, \ldots, u'_t$ be t neighbors of u other than the r neighbors u_i, and let the set of $|E|$ vertices v'_i with $f(v'_i) = u'_i$ for each u'_i be joined by a matching to the $|E|$ vertices v with $f(v) = u$ in G. These vertices v'_i have $D_{v'_i} = \{1\}$. All other vertices w in G have no incident edges and $D_w = \{0\}$, for a total of $|E| \cdot |V(H)|$ vertices in G.

Note that each v with $f(v) = u$ has $t + r$ incident edges and that the t edges joining v to vertices v'_i must be chosen since $D_{v'_i} = \{1\}$. Thus the only choice for each v is to select or not select all the r edges joining v to vertices v_i, which corresponds to choosing or not choosing the associated tuple $(v_1, \ldots, v_r)$ in $|E|$. Furthermore, each element v_i of V_i must be in exactly one chosen tuple from E since $D_{v_i} = \{1\}$. Thus solutions to the degree constrained problem for H are in correspondence with solutions to the r-dimensional matching problem. $\square$

We now consider the optimization problem that asks to maximize the number of S_i that induce a subgraph isomorphic to H. The argument from the preceding two theorems gives the following.

Theorem 3.3 *The optimization problem can be solved in polynomial time if H is a forest such that each connected component of H is a path, and is NP-complete otherwise.*

Proof. Suppose H is a path. Let u be the first vertex of H. Add to H a vertex u' adjacent to u to obtain a path H'. Given an instance G of the optimization problem for H with kl vertices and an integer r, we construct an instance G' of the degree constrained problem for H', by adding l vertices v' with $f(v') = u'$ and all the edges joining some r of these vertices v' to all vertices v with $f(v) = u$. These r vertices v' have $D_{v'} = \{1\}$, and the remaining $l - r$ vertices v' have $D_{v'} = \{0\}$. If u'' is the last vertex of the path H, then all vertices v'' with $f(v'') = u''$ have $D_{v''} = \{0, 1\}$. All other vertices v have $D_v = \{0, 2\}$. It follows that a solution to the degree constrained problem will select r paths T_i, and the optimization problem can be solved by choosing the largest r for which a solution to the degree constrained problem exists.

If H is a forest and each connected component of H is a path, we can solve the optimization problem for each connected component of H and obtain the largest r for each such component, and then the solution for H is the minimum of these r over the various components of H.

If H is not a forest then the optimization problem is NP-complete by Theorem 2.1. If H is a forest that has a component that is not a path, then H has a vertex u with at least three neighbors u_1, u_2, u_3. Consider an instance of the 3-dimensional matching problem as in Theorem 2.1. As in the proof of Theorem 3.2, we may represent each of the $|E|$ tuples (v_1, v_2, v_3) by a vertex v with $f(v) = u$ adjacent to the three vertices v_i, which have $f(v_i) = u_i$, with additional $|E| - l$ vertices v_i having $f(v_i) = u_i$. For each of the edges $u'u''$ in H such other than the edges uu_i, include all edges $v'v''$ with $f(v') = u'$ and $f(v'') = u''$ in G. It is then possible to choose l copies S_i of H if and only if it is possible to choose l vertices v with $f(v) = u$ such that the corresponding tuples (v_1, v_2, v_3) form a solution to the 3-dimensional matching instance, so that for each selected tuple the vertices v, v_1, v_2, v_3 are in a copy S_i; all other vertices v' for S_i can be chosen arbitrarily. Thus maximizing the number of copies S_i such that f is an isomorphism from S_i to H is NP-complete. □

We strengthen this result.

Theorem 3.4 *The optimization problem in the case where H is a forest for which each component is a path remains polynomial in a weighted version where vertices and edges have weights, and we ask to maximize the sum of the weights of vertices and edges over components forming copies of H. The remaining NP-complete cases are hard to approximate within some constant $c > 1$ even for bounded degree instances, allowing a $\frac{k}{2} + \epsilon$ approximation in the unweighted case, and both a $k - 1 + \epsilon$ and a $\frac{2(k+1)}{3}$ approximation in the weighted case where each candidate set of size k is given a weight and we wish to maximize the sum of weights of chosen sets S_i forming copies of H.*

Proof. Suppose H is a path with k vertices, and we wish to obtain r copies of H of maximum weight. We express the problem as a min-cost flow problem having levels i for $0 \leq i \leq 2k+1$, and edges of capacity 1 from level i to level $i+1$. At level 0 is a single vertex with supply r joined by edges to all vertices at level 1, and at level $2k+1$ is a single vertex with demand r joined by edges from all vertices at level $2k$. Choose a large quantity M. The l vertices in G corresponding to a vertex in position i in the path H are represented in the min-cost flow problem by a matching of size l joining level

$2i-1$ to level $2i$, where a vertex of weight w corresponds to an edge in the matching of cost $M-w$. The edges joining two vertices in G corresponding to an edge joining position i to position $i+1$ in H are represented in the min-cost flow problem by edges joining two matched vertices at levels $2i$ (matched to level $2i-1$) and $2i+1$ (matched to level $2i+2$), where if the edge in G had weight w, then the corresponding edge in the flow problem has cost $M-w$. The min-cost flow problem has a polynomial time algorithm, see Edmonds and Karp [7]. The optimal solution of minimum cost can here be assumed to be integral. Thus a path of total weight W corresponds to a unit of flow with total cost $(2k-1)M-W$, and a solution consisting of r such paths with total weight W consists to a solution to the min-cost flow problem with total cost $r(2k-1)M-W$. This completes the construction when H is a path; when H is a forest with each component consisting of a path, we choose r paths for each component of H by a separate min-cost flow.

To show hardness of approximation, we note that when H is not a forest, then the proof from Theorem 2.1 selects a component that is not a tree, finds the shortest cycle in this component, and reduces the problem for H a cycle to the problem for H a triangle. The problem of selecting the maximum number of disjoint triangles in a 3-partite graph is shown hard to approximate within some constant $c>1$ by Kann [14], even if the input graph G has degrees at most 6. The same result with degrees at most 6 thus applies to a cycle of any length $k\geq 3$ as in Theorem 2.1. If H contains a cycle of length k, then we may include l copies of each component of H minus the cycle, each connected to a corresponding vertex v mapping to a vertex u in the cycle, and also to other vertices v' mapping to other vertices u' in the cycle, while taking into consideration that only at most 6 vertices v' may be in the same cycle as v. This bounds the degree in the instance showing hardness of approximation.

If H is a forest containing a component that is not a path, then as in the proof from Theorem 3.3 we are left with the case of a star with three edges, which reduces from 3-dimensional matching. The 3-dimensional matching problem is shown hard to approximate within some constant $c>1$ by Kann [14] in the case where every element belongs to at most 3 sets, so the hardness for H a star with 3 edges holds with maximum degree 3. As before, we may attach l copies of the subtrees involving edges not in the star with 3 edges to obtain the result when H is a forest with a component that is not a path.

Given a set S and a collection C of subsets of S of size k, the problem of selecting the maximum number of disjoint subsets from C can be approximated within $\frac{k}{2} + \epsilon$ by an algorithm of Hurkens and Schrijver [13]. If the sets in C have nonnegative weights and we wish to maximize the sum of weights of disjoint subsets selected from C, then Arkin and Hassin [1] give a $k - 1 + \epsilon$ approximation and Chandra and Halldórsson [5] give a $\frac{2(k+1)}{3}$ approximation. All three results apply to the allowed choices of subsets of size k forming a subgraph isomorphic to H in our problem. □

We now consider a set constrained problem where each vertex v of an instance G with $f(v) = u$ is given a subset R_v of the set $\mathcal{P}(N_u)$ of subsets of the set N_u of neighbors of u in H. The set R_v is *upward closed* when if $T \in R_v$ and $T \subseteq U \subseteq N_u$, then $U \in R_v$. We shall require that each S_i contain a subgraph T_i with $V(T_i) = S_i$ such that if v is a vertex in some T_i and M_v is the set of neighbors of v in T_i then the set of $f(w)$ for w in M_v is in R_v. Note that if R_v is upward closed, then we may always take T_i to be the subgraph induced by S_i.

Let M be a set of n-bit vectors. The set M can be viewed also as a set of subsets of the set N of n bit positions, with $|N| = n$, under the correspondence between an n-bit vector x and the set of bit positions equal to 1 in x. Given two n-bit vectors x, y, the distance $d(x, y)$ is the number of bit positions where x and y differ. The set M of n-bit vectors is a *delta-matroid* if for all n-bit vectors x, y in M, if z is an n-bit vector such that $d(x, z) = 1$ and $d(z, y) = d(x, y) - 1$, then either z is in M, or there exists an n-bit vector t in M such that $d(z, t) = 1$ and $d(t, y) = d(z, y) - 1$. A delta-matroid M is a *matroid* if all n-bit vectors in M have the same number of 1s, that is, all associated sets are of the same size.

A delta-matroid M *reduces* to M' if M' is obtained from M by repeatedly performing any one of the following operations: (1) removing from M all n-bit vectors that have a 0 in a given position i; (2) removing from M all n-bit vectors that have a 1 in a given position i; (3) replacing M with a set of $(n-1)$-bit vectors by removing from each n-bit vector x in M the bit in a given position i. If M reduces to M', then M' is also a delta-matroid.

Theorem 3.5 *If H is a forest, and each R_v is an upward closed delta-matroid, then this set constrained problem can be solved in polynomial time via a matroid intersection algorithm. The problem remains polynomial when edges have weights and the aim is to find a valid partition that maximizes the total weight of edges with endpoints in the same S_i, via a weighted matroid intersection algorithm.*

Proof. As in the proof of Theorem 3.1, associate with an instance G that maps to H a new graph G' by replacing each vertex v in G such that

$f(v) = u$ with $r_v + 1$ vertices $a_v, b_{v,u'}$ for each neighbor u' of u in H, with r_v edges $a_v b_{v,u'}$, and an edge $b_{v,u'}, b_{v',u}$ for each edge vv' in G such that $f(v) = u$, $f(v') = u'$.

The problem is again equivalent to selecting a subset F' of edges of G' such that the set of edges $a_v b_{v,u'}$ in F' incident to a given a_v is such that the set of corresponding vertices u' is in R_v; and each $b_{v,u'}$ is incident to $d_{v,u'} - 1$ edges of F', where $d_{v,u'}$ is the degree of $b_{v,u'}$ in G'.

Note that the constraint imposed on the edges incident to $b_{v,u'}$ in F' forms a matroid, namely all subsets of size $d_{v,u'} - 1$ of a set of size $d_{v,u'}$. The constraint imposed on the edges incident to a_v in F' forms a delta-matroid. The graph G' is bipartite, and all the edges in G' are constrained by the vertices of $G' = (U, V, E)$ in each side of G' to belong to a given delta-matroid. Therefore the problem asks to find a set of edges in the intersection of two given delta-matroids, one specified by the vertices in U and one specified by the vertices in V.

If both delta-matroids are matroids, then the problem can be solved in polynomial time by matroid intersection. Feder [8] showed that if M is a set of n-bit vectors forming a delta-matroid, and M does not reduce to either of two 2-bit vector delta-matroids given by $\{00, 11\}$ or by $\{00, 01, 11\}$, then under the correspondence between each n-bit vector x in M and all the $2n$-bit vectors y such that the first n bits of y are the same as the n bits of x, and the total number of 1s in y is n, the resulting set of $2n$-bit vectors y in M' obtained from n-bit vectors x in M forms a matroid.

The delta-matroids R_v were chosen in such a way that every superset of a set in R_v is also in R_v. This property carries over to delta matroids to which R_v reduces. In particular, if such a delta-matroid has the 2-bit vector 00, then it must also have 01 and 10. Therefore the two forbidden 2-bit vector delta-matroids do not occur, and so the two delta-matroids specified by U and by V can be described as matroids by introducing extra edges. Although the extra edges are constrained only by a vertex in U or by a vertex in V, we can include a copy of U in the side V and a copy of V in the side U, with two identical instances, and the extra edges incident to a single vertex in one side can be made incident to the same vertex in the other side. The problem is thus matroid intersection, which can be solved in polynomial time. The weighted version is a weighted matroid intersection problem that can also be solved in polynomial time [10]. □

Theorem 3.6 *If H is a fixed forest containing a vertex u, each vertex v with $f(v) = u$ is assigned the same fixed set $R = R_v$ that is upward closed* *but not a delta-matroid, and for all other vertices w the set R_w is $\mathcal{P}(N_u)$ then the set constrained problem is NP-complete.*

Proof. We reduce the problem from 3-satisfiability, under the constraint that each variable appears in at most 3 clauses. This restriction is NP-complete, since multiple equal variables can be simulated with clauses $x_1 \vee \overline{x_2}, x_2 \vee \overline{x_3}, \ldots, x_{n-1} \vee \overline{x_n}, x_n \vee \overline{x_1}$, with two occurrences per variable, leaving a third occurrence of each of the n copies to be used elsewhere.

Feder [8] showed that if a set M of n-bit vectors is not a delta-matroid, then M reduces to a set M' of 3-bit vectors containing in particular a vector $b_1b_2b_3$ and its complement $\overline{b_1b_2b_3}$, and possibly other vectors, but no vector of the form $b_1x_2x_3$ other than $b_1b_2b_3$.

Since R_v was chosen so that each superset of a subset in R_v is also in R_v, this property must also be true for the set of 3-bit vectors M', giving as the only possibility $b_1b_2b_3 = 011$, with $M' = \{011, 100, 101, 110, 111\}$, so that changing a 0 to 1 in a vector in M' stays in M'.

The M' that is not a delta-matroid is obtained from R_v by requiring that certain u' adjacent to u in H be such that no edge vv' with $f(v') = u'$ be selected, so we may just never include such vv' edges; and requiring that other u' adjacent to u in H either be such that an edge vv' with $f(v') = u'$ is always selected or allowed to be possibly selected, both choices being the same since a non-selected case can be changed to a selected case becuase supersets of subsets in R_v are also in R_v, so we may include all edges vv' with $f(v) = u$ and $f(v') = u'$ in this case to allow for the selection to take place.

There only remain three neighbors u_1, u_2, u_3 of u to consider, in the bit positions corresponding to M'. For each clause $x \vee y \vee z$ in the 3-satisfiability instance, include three vertices v_x, v_y, v_z with $f(v_x) = f(v_y) = f(v_z) = u$, and two vertices c, c' with $f(c) = f(c') = u_1$. If a variable x appears in three clauses, then we may assume that it appears sometimes positive and sometimes negative, otherwise the clauses are easily satisfied. Say we have a positive occurrence called x_1 and two negative occurrences called x_2, x_3. Include then two vertices w_2, w_2' with $f(w_2) = f(w_2') = u_2$, two vertices w_3, w_3' with $f(w_3) = f(w_3') = u_3$, and edges from x_1 to both w_2, w_3', from x_2 to both w_2, w_3, and from x_3 to both w_2', w_3'.

The condition given by M' says that for v with $f(v) = u$, either an edge to v' with $f(v') = u_1$ is selected, or edges to v'', v''' with $f(v'') = u_2$ and $f(v''') = u_3$ are selected. Since v_x, v_y, v_z are in different S_i, only two of these S_i may contain c or c', so for the remaining S_i, say the one containing v_x, we must select edges to v'' and to v'''. We view this as choosing the literal corresponding to x to satisfy the clause $x \vee y \vee z$. Notice that we may not choose x for one clause and $\overline{x}$ for another clause, because if we choose both x_1 and x_2, then w_2 would have to be in the same S_i as both x_1 and x_2, which is not possible, and similarly if we choose both x_1 and x_3, then w'_3 would have to be in the same S_i as both x_1 and x_3, which is not possible. It is however possible to choose both x_2 and x_3, with w_2, w_3 in the same S_i as x_2 and w'_2, w'_3 in the same S_i as x_3. Thus a solution satisfying M' corresponds to a solution to the 3-satisfiability problem. We may add isolated vertices v with $f(v) = u'$ for $u' \neq u$ until all families have the same number of vertices as the number of vertices v with $f(v) = u$. $\square$

Feder [8] showed that each boolean satisfiability problems with two ccurrences of each variable, is either among Schaefer's polynomial cases [17] without bound on the number of occurrences of each variable, consists only of delta-matroid constraints of constant size, or is NP-complete. The complexity of the case known as delta-matroid parity for delta-matroids that are the direct sum of delta-matroids of constant size remains open, with the exception of strictly bipartite cases [9] that are known to be polynomial.

Theorem 3.7 *If H is a forest, and each R_v is a delta-matroid, then this set constrained problem reduces to delta-matroid parity for delta-matroids that are the direct sum of delta-matroids of constant size.*

Proof. Again, as in the proof of Theorem 3.5, we associate with an instance G that maps to H the corresponding graph G', and G' gives a delta-matroid intersection problem, or equivalently a delta-matroid parity problem. We can now apply the result of Ford and obtain a polynomial time algorithm, by showing that the delta-matroid is the direct sum of delta-matroids on a constant number of elements. The delta-matroids in the direct sum are given here by the constraints on selected edges out of each vertex v in G', so it suffices here to obtain vertices of constant degree. The degree of the vertices a_v is bounded by the constant $k = |V(H)|$. The vertices $b_{v,u'}$ have unbounded degree $d_{v,u'}$, but have an associated matroid that is very simple, namely selecting any $d_{v,u'} - 1$ out of the $d_{v,u'}$ edges coming out of $b_{v,u'}$. This can be simulated with vertices of degree 2 and 3, by replacing $b_{v,u'}$ with a path $x_0, x_1, \ldots, x_{2d_{v,u'}-2}$, with the $d_{v,u'}$ edges incident to $b_{v,u'}$ now coming out of the $d_{v,u'}$ vertices x_{2i} for $0 \leq i < d_{v,u'}$, requiring that exactly one edge be selected for x_0, $x_{2d_{v,u'}-2}$, and each x_{2i+1}, and that exactly two edges be selected for the remaining x_{2i} for $1 \leq i \leq d_{v,u'} - 2$. This completes the bounding of degrees and gives the delta-matroids on a constant number of elements in the direct sum. $\square$

Theorem 3.8 *If H is a fixed forest containing a vertex u, each vertex v with $f(v) = u$ is assigned the same fixed set $R = R_v$ that is not a delta-matroid, and for all other vertices w the set R_w is either $\mathcal{P}(N_u)$ or selects the sets in $\mathcal{P}(N_u)$ containing a specific element, then the set constrained problem is NP-complete.*

Proof. As in the proof of Theorem 3.6, the set R_v reduces to a set M' of 3-bit vectors containing in particular a vector $b_1b_2b_3$ and its complement $\overline{b_1b_2b_3}$, and possibly other vectors, but no vector of the form $b_1x_2x_3$ other than $b_1b_2b_3$.

Again as in Theorem 3.6, the problem reduces to the case where there only remain three neighbors u_1, u_2, u_3 of u to consider, in the positions corresponding to M', where for the remaining position we have either included no edges vv' if the position corresponding to $u' = f(v')$ was set to 0, and included all edges vv' with $u' = f(v')$ otherwise, where if the position corresponding to u' was set to 1 then this is enforced by requiring in v' that the position corresponding to u be set to 1.

There are four cases, depending on whether $b_1b_2b_3$ is 000, 111, 011, 100, 101, or 010. The case 011 goes through as in Theorem 3.5. In fact, we can enforce ahead of time that there be exactly some r_i edges corresponding to u_iu selected for $i = 1, 2, 3$, by requiring that each v_i with $f(v_i) = u_i$ have an edge to a v with $f(v) = u$ selected, and add $l - r_i$ vertices v with $f(v) = u$ joined to all such v_i, and matched with $l - r_i$ new vertices v_j with $f(v_j) = u_j$ for $j \neq i$. This makes the case 100 go through as well. For the case, 101, exchange the first two bits to obtain 011, with setting $b_2 = 1$ forcing $b_1 = 0$ and $b_3 = 1$, so this case goes through; and similarly, for the case 010, exchange the first two bits to obtain 100, with setting $b_2 = 0$ forcing $b_1 = 1$ and $b_3 = 0$, so this case also goes through.

For the two remaining cases, 000 and 111, we can set $r_i = r$ for $i = 1, 2, 3$, so that since a 0 forces 000 in the first case, and a 1 forces 111 in the second case, we are only allowing 000 and 111, and requiring that 111 be selected r times, and this problem is NP-complete by the NP-completeness of the optimization problem for the star with three edges from Theorem 3.3. □

4 r-Edge Subgraph and Connected Subgraph

We now consider the r-edge subgraph problem, where each S_i is required to induce a subgraph of G with at least r edges, and the r-edge connected subgraph problem, where each S_i is required to induce a subgraph of G containing a connected component with at least r edges.

Theorem 4.1 *Assume H is connected. Then the r-edge subgraph and connected subgraph problems can be solved in polynomial time if $r = 1$, or if $1 < r < |E(H)|$ and H is a star, or if $r = |E(H)|$ and H is a tree. Otherwise both problems are NP-complete.*

Proof. Recall $|V(H)| = k$ and $|V(G)| = kl$. If $r = 1$, then each S_i must have at least one edge, so G must have a matching of size l. If G has a matching of size l, then the l matched edges may be included in different S_i. Therefore the case $r = 1$ can be solved in polynomial time by matching. The polynomial and NP-complete cases for $r = |E(H)|$ are covered by Theorem 2.1.

So assume $1 < r < |E(H)|$. The polynomial case when H is a star is covered by Theorem 3.1 since this can be viewed as a degree r constraint for the single vertex of the star incident to all the edges of H.

We thus assume H is not a star. If $2 \leq r \leq E(H) - 2$, then we may repeatedly remove edges of H until the remaining edges form a connected subgraph H' that is still not a star but has $2 \leq r = E(H') - 1$.

We thus assume H is not a star and $2 \leq r = E(H) - 1$. We simplify both problems to the special cases with $|E(H)| = 3$, namely a triangle or a path of length 3. If H is not a tree, then H contains a cycle, which either is a triangle or contains a path of length 3. For all edges uu' in H not in the triangle or the path of length 3, we may include all edges vv' such that $f(v) = u$ and $f(v') = u'$, so that both problems are equivalent to the 2-subgraph problem on the triangle or the path of length 3. If H is a tree but not star, then H contains a path p of length at least 3 joining two leaves. For the r-edge subgraph problem, it suffices to select a path q of length 3 contained in p, and for all uu' in H not on q include all edges vv' such that $f(v) = u$ and $f(v') = u'$, so that the problem is equivalent to the problem on the path q of length 3. For the r-edge connected subgraph problem, for all uu' in H not on p include all edges vv' such that $f(v) = u$ and $f(v') = u'$, so that the problem is equivalent to the problem on the path p of length at least 3. If the path p has length $t \geq 4$, then given the problem for a path of length 3 we may select one of the two internal vertices u on the path and replace it by a path s of length $t - 3$ to obtain a path of length t, and similarly replace vertices v with $f(v) = u$ in an instance with paths mapping to s. The connected problem for a path of length 3 thus reduces to the problem for the path p of length $t \geq 3$.

We are thus left with the two cases of a triangle and of a path of length 3, with $r = 2$. For the case of a path of length 3, given by $u_0u_1u_2u_3$, we consider instances with some number l_1 of vertices v with $f(v) = u_0$, some number l_2 of vertices v with $f(v) = u_3$, and $l_1 + l_2$ vertices v of each kind with $f(v) = u_1$ and with $f(v) = u_2$, and ask whether such a graph G can be decomposed into l_1 paths mapping to $u_0u_1u_2$ and l_2 paths mapping to $u_1u_2u_3$ under f.

If this problem can be shown NP-complete, then adding l_2 isolated vertices v with $f(v) = u_0$ and l_1 isolated vertices v with $f(v) = u_3$ gives the NP-completeness with $r = 2$ for the r-edge connected subgraph problem, since one of the two chosen edges for each S_i must here always be the middle edge corresponding to u_1u_2. Also the NP-completeness with $r = 2$ for the r-edge subgraph problem is obtained, since if we use the two edges u_0u_1 and u_2u_3 for $m > 0$ sets S_i, then we can only have $l_1 - m$ sets S_i using $u_0u_1u_2$ and $l_2 - m$ sets S_i using $u_1u_2u_3$, thus accounting for only $l_1 + l_2 - m$ sets S_i, instead of $l_1 + l_2$. So a solution must here coincide with a solution for the r-edge connected subgraph problem.

Given an instance of the 3-dimensional matching problem as in Theorem 2.1 with triples (v_1, v_2, v_3), replace each such triple with a graph R having paths $a_0a_1a_2$, $b_1b_2b_3$, $c_1c_2c_3$, and additional edges b_1a_2, a_2c_3, a_1v_2, v_1b_2, c_2v_3. Each copy of R is connected to the rest of the graph only through v_1, v_2, v_3, and all paths $x_0x_1x_2$, $y_1y_2y_3$ are contained within the graphs R. We may thus either select the three identified paths of a_i, b_i, c_i respectively, covering no v_i, or cover all three v_i by choosing four paths $v_1b_2b_3$, $a_0a_1v_2$, $c_1c_2v_3$, $b_1a_2c_3$. Thus a solution to the problem solves the 3-dimensional matching problem.

The remaining problem has $r = 2$ for a triangle H with vertices u_1, u_2, u_3. Again a reduction from the 3-dimensional matching problem with triples (v_1, v_2, v_3) replaces each such triple with a graph R having paths $a_1a_2a_3$, $b_1b_2b_3$, $c_3c_1c_2$, and additional edges b_1c_2, c_2a_3, v_1b_2, c_1v_2, a_2v_3. Each copy of R is connected to the rest of the graph only through v_1, v_2, v_3, and all paths $x_1x_2x_3$, $y_3y_1y_2$, $z_1z_3z_2$ are contained within the graphs R. We may thus either select the three identified paths of a_i, b_i, c_i respectively, covering no v_i, or cover all three v_i by choosing four paths $v_1b_2b_3$, $c_3c_1v_2$, $a_1a_2v_3$, $b_1c_2a_3$. Thus a solution to the problem solves the 3-dimensional matching problem. □

We give a complete characterization for the r-edge connected subgraph problem. The case where H has only one connected component with at least r edges is covered by the preceding theorem.

Theorem 4.2 *Assume H has at least two connected components with at least r edges. Then the r-edge connected subgraph problem can be solved in polynomial time if $r = 1$, or if $r = 2$ and all connected components of H are stars, or if $r \geq 3$ and all connected components of H with at least r edges are paths with exactly r edges. Otherwise the problem is NP-complete.*

Proof. We give first the NP-completeness proofs. We may first remove edges from H to obtain a graph H' with exactly two connected components H'_1 and H'_2 each having exactly r edges, and show NP-completeness for H'. In an instance G' for H' with l vertices v such that $f(v) = u$ for each u in H, we may include $l - t$ disjoint copies of H'_2. The problem then becomes finding t copies of H'_1 in the pre-image of H'_1 under f, and this optimization problem for H'_1 is NP-complete unless H'_1 is a path by Theorem 3.3. If $r \geq 3$ and the component H_1 in H that H'_1 comes from has at least one vertex of degree 3, then we may choose H'_1 not to be a path. Therefore the problem is NP-complete unless H_1 is a path or a cycle, but if H_1 is a cycle then the problem for H_1 is NP-complete by Theorem 4.1, and similarly if H_1 is a path of length strictly greater than r then the problem for H_1 is also NP-complete by Theorem 4.1. We have thus shown NP-completeness for $r \geq 3$ unless each component of H with at least r edges is a path of length r. If $r = 2$, then each component H_1 of H with at least r edges must be a star by Theorem 4.1, else the problem is NP-complete.

We prove polynomiality in the remaining cases. If $r \geq 3$ and all components of H with at least r edges are paths H_i of length r, then we may determine the maximum number of copies l_i of each such H_i that can be obtained from the preimage under f of H_i by Theorem 3.3. The instance then has a solution if the sum of these l_i is at least l.

If $r = 2$ and all components of H with at least r edges are stars H_j, then we can determine whether we can obtain l_j subgraphs with $r = 2$ edges from the star H_j by degree constraints from Theorem 3.1. If the star H_j consists of a root u adjacent to s leaves u_i, assign degree constraint $\{0, 2\}$ to the vertices v such that $f(v) = u$, assign degree constraint $\{1\}$ to the vertices v_i such that $f(v_i) = u_i$, and add $sl - 2l_j$ vertices v with $f(v) = u$ and adjacent to all v_i such that $f(v_i) = u_i$, with each such v having degree constraint $\{1\}$. Add also $sl - 2l_j$ isolated vertices v_i such that $f(v_i) = u_i$ for each u_i, with degree constraint $\{0\}$. The neighbors of the added $sl - 2l_j$ vertices v with $f(v) = u$ will only match $sl - 2l_j$ out of the original sl vertices v_i with $f(v_i) = u_i$, so the remaining $2l_j$ vertices v_i must be matched with degree 2 by l_j original vertices v such that $f(v) = u$. Thus the maximum l_j for each star H_j can be determined, and the algorithm ends by determining if the sum of these l_j is at least l. □

We give a partial characterization for the r-edge subgraph problem. The case $r = 1$ is the same as the r-edge connected subgraph, and the case $r = |E(H)|$ is the basic problem.

Theorem 4.3 *Assume* $1 < r < |E(H)|$. *Then the* r*-edge subgraph problem is NP-complete if* H *has a connected component that is not a star.*

Proof. Assume H has a component H_1 with s edges that is not a star. Choose $t = \min(|E(H)| - s), r - 2)$ edges uu' in H and not in H_1, and include in an instance all edges vv' such that $f(v) = u$ and $f(v') = u'$ for such a uu', and none for an edge uu' in H and not in H_1 and not among the t chosen edges. The problem is thus the same as a given problem for H_1 with corresponding $r_1 = r - t$. Note that $1 < r_1 < s = |E(H_1)|$, so by Theorem 4.1 the problem is NP-complete if H_1 is not a star. □

The classification of the r-edge subgraph problem in the cases where each connected component of H is a star remains open. We obtain partial results.

Theorem 4.4 *Suppose each connected component of* H *is a star, and* $1 < r < |E(H)|$. *(a) If* $3 \leq r \leq |E(H)| - 3$ *and* H *has at least two connected components with at least* 3 *edges, then the* r*-edge subgraph problem is NP-complete; (b) For* $r = 2$, $r = |E(H)| - 2$, *and* $r = |E(H)| - 1$, *the* r*-edge subgraph problem can be solved in polynomial time; (c) If every connected component of* H *has at most* 2 *edges, then the* r*-edge subgraph problem can be solved in polynomial time; (d) If only one connected component of* H *has at least* 3 *edges, and at most one connected component of* H *has* 2 *edges, so that all other connected components of* H *have only* 1 *edge, then the* r*-edge subgraph problem can be solved in polynomial time.*

Proof. We prove (a). We may remove edges from H to obtain a subgraph H' consisting of just two stars H_1 and H_2 such that each H_i consists of a root with three edges coming out of it. We may select $r-3$ out of the $|E(H)-6|$ remaining edges of H not in H', include in an instance all edges vv' such that $f(v) = u$ and $f(v') = u'$ with uu' among these selected $r - 3$ edges, and not include in the instance any edge vv' such that $f(v) = u$ and $f(v') = u'$ with uu' not among the selected $r - 3$ edges and not in H'. The problem then reduces from the problem for H' with $r' = 3$. In an instance for H', include $l - t$ disjoint copies of the star H_2'. The problem then becomes finding t

copies of the star H_1' in the preimage under f of H_1', and this optimization problem is NP-complete by Theorem 3.3 since H_1' is not a star.

We prove (b) for $r = 2$. Here it suffices to determine for each star component H_i of H all pairs of integers l_1, l_2 such that we may have 1 edge from H_i in l_1 sets S_i, and 2 edges from H_i in l_2 other sets S_i, with $l_1 + l_2 \leq l$. We may determine the maximum possible value z for l_2 as in Theorem 4.2, using degree constraints $\{0, 2\}$ for vertices v with $f(v) = u$, where u is adjacent to all vertices in the star H_i. Similarly, we may determine the maximum possible value s for $l_1 + l_2$ by using degree constraints $\{0, 1\}$ for vertices v with $f(v) = u$, and the maximum value t for $l_1 + 2l_2$ using degree constraints $\{0, 1, 2\}$. Since every maximum matching has the same number of edges, we may in particular obtain solutions with $l_1 + 2l_2 = t$ such that $l_2 = z$, or such that $l_1 + l_2 = s$ with some value $y = l_2$. In fact, by considering the alternating paths distinguishing these two maximum matchings and switching them one at time, we obtain solutions with $l_1 + 2l_2 = t$ for all choices of l_2 satisfying $y \leq l_2 \leq z$.

Once this has been done, if H consists of x stars H_i, we may choose for each star a combination l_1, l_2 in at most l^x possible ways, and choose where the l copies of 2 edges will come from in at most l^y possible ways for $y = x^2$ possible ways, completing an exhaustive search for a solution in the case $r = 2$.

The proof of (*b*) for $r = |E(H)| - 2$ is similar. Here each copy of H_i must select at least $|E(H_i)| - 2$ edges, and choose $|E(H_i)| - 1$ edges for l_1 copies, and all $|E(H_i)|$ edges for l_2 copies. We may again determine the maximum value z for l_2 using degree constraints $\{|E(H_i)| - 2, |E(H_i)|\}$, the maximum value s for $l_1 + l_2$ using degree constraints $\{|E(H_i)| - 1, |E(H_i)|\}$, and similarly the maximum value t for $l_1 + 2l_2$ using degree constraints $\{|E(H_i)| - 2, |E(H_i)| - 1, |E(H_i)|\}$. We may then obtain all solutions with $l_1 + 2l_2 = t$ such that $l_2 = z$, or such that $l_1 + l_2 = s$ with some value $y = l_2$. Again, by considering the alternating paths distinguishing these two maximum matchings and switching them one at time, we obtain solutions with $l_1 + 2l_2 = t$ for all choices of l_2 satisfying $y \leq l_2 \leq z$. We can then again consider the possible choices of how many edges will come from a copy of which H_i, say x possible choices, so that there are at most l^x possible combinations.

The case of (b) for $r = |E(H)| - 1$ is easier since only one of the $|E(H_i)|$ edges may be missed, and can be handled by maximizing just the number z of copies that get all $|E(H_i)|$ edges, by matching, and ensuring that the quantities $l - z$ over different H_i, counting edges missed, add up to at most l.

The proof of (c) with stars H_i having at most two edges involves the same counting for each H_i as with $r = 2$, determining the possible pairs l_1, l_2 for each H_i, and then finding all possible combinations among different H_i as to where the r edges come from, say x possible combinations, for a total of l^x possible cases.

The proof of (d) finds a maximum matching in the preimage of H_i under f for each component H_i with only one edge. If this maximum matching has l_i edges, it removes H_i and its preimage, adds an edge uu' to the star H_1 with at least 3 edges and u incident to all its edges, and adds edges from l_i vertices in the preimage of u' under f to all vertices in the preimage of u. The two problems are equivalent, since matching l_i vertices in the preimage of u to vertices in the preimage of u' adds count 1 for l sets S_i, which could have been obtained from the preimage of H_i. We are thus left with the case where H_1 has at least three edges, and there is just one other component H_2, with exactly two edges.

The algorithm considers again the possible values l_1, l_2 achievable with the preimage of H_2. For H_1, we consider combinations with all copies having at least $r-2$ edges, l_1 copies having $r-1$ edges and l_2 copies having r edges, successively using degree constraints $\{r-2, r\}$ then $\{r-2, r-1\}$, and then $\{r-2, r-1, r\}$ as before in the cases $r = 2$ and $r = |E(H)| - 2$ from (b). The final step combines the possible solutions giving l_1, l_2 for H_1 and for H_2. □

In the remaining open cases of the r-edge subgraph problem, each connected component of H is a star, $3 \leq r \leq |E(H)| - 3$, exactly one connected component has at least 3 edges, and there are at least two connected components with 2 edges, with the remaining connected components having only 1 edge.

References

[1] E.M. Arkin and R. Hassin, On local search for weighted packing probems, Math. Oper. Res. 23 (1998) 640–648.

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

[3] F. Berman, D. Johnson, T. Leighton, P.W. Shor, and L. Snyder, Generalized planar matching, J. Algorithms 11 (1990) 153–184.

[4] P. Berman and T. Fujito, Approximating independent sets in degree 3 graphs, Proc. 4th Workshop on Algorithms and Data Structures, Lecture Notes in Comput. Sci. 955, Springer-Verlag (1995) 449–460.

[5] B. Chandra and M.M. Halldórsson, Greedy local improvement and weighted set packing approximation, Proc. 10th Ann. ACM-SIAM Symp. on Discrete Algorithms (1995) 169–176.

[6] G.P. Cornuejols, General factors of graphs, J. Combinatorial Theory, Series B, 45 (1988) 185–198.

[7] J. Edmonds and R.M. Karp, Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems, J. ACM 19 (1972).

[8] T. Feder, Fanout limitations on constraint systems, Theoretical Computer Science 255 (2001) 281–293.

[9] T. Feder and D. Ford, Classification of bipartite boolean constraint satisfaction through delta-matroid intersection, manuscript.

[10] H. Gabow, A matroid approach to finding edge connectivity and packing arborescences, J. Comp. and Syst. Sci. 50 (1995) 259–273.

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

[12] R. Hassin and S. Rubinstein, An approximation algorithm for maximum packing of 3-edge paths, Inform. Process. Lett. 63 (1997) 63–67.

[13] 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 J. Disc. Math. 2 (1989), 68–72.

[14] V. Kann, Maximum bounded 3-dimensional matching is MAX SNP-complete, Inform. Process. Lett. 37 (1991) 27–35.

[15] V. Kann, Maximum bounded H-matching is MAX SNP-complete, Inform. Process. Lett. 49 (1994) 309–318.

[16] D.G. Kirkpatrick and P. Hell, On the complexity of a generalized matching problem, Proc. 10th Ann. ACM Symp. on Theory of Computing (1978) 240–245.

[17] T.J. Schaefer, The complexity of satisfiability problems, Proc. 10th Ann. ACM Symp. on Theory of Computing (1978) 216–226.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'