Efficient algorithms for decomposing graphs under degree constraints

Cristina Bazgan * Zsolt Tuza † Daniel Vanderpooten *

Abstract

Stiebitz (1996) proved that if every vertex v in a graph G has degree $d(v) \geq a(v) + b(v) + 1$ (where a and b are arbitrarily given nonnegative integer-valued functions) then G has a nontrivial vertex partition (A, B) such that $d_A(v) \geq a(v)$ for every $v \in A$ and $d_B(v) \geq b(v)$ for every $v \in B$. Kaneko (1998) and Diwan (2000) strengthened this result, proving that it suffices to assume $d(v) \geq a+b$ $(a, b \geq 1)$ or just $d(v) \geq a+b-1$ $(a, b \geq 2)$ if G contains no cycles shorter than 4 or 5, respectively.

The original proofs contain nonconstructive steps. In this paper we give polynomial-time algorithms that find such partitions. Constructive generalizations for k-partitions are also presented.

Keywords: graph decomposition, vertex partition, polynomial algorithm, vertex degree.

1 Introduction

In this paper we give constructive proofs of the results published in [9, 7, 4, 6] concerning the vertex decomposability of graphs where some degree constraints are satisfied.

For a graph G, we denote by $V(G)$ and $E(G)$ the vertex set and the edge set, respectively; or simply by V and E if G is understood. Given a set $S \subseteq V(G)$, the subgraph of G induced by S is denoted by $G[S]$; and we write $d_S(v)$ for the degree of a vertex v in $G[S \cup \{v\}]$ (i.e., $v \in S$ may or may not hold). Then the degree of v (in G) is $d(v) = d_V(v)$. Moreover, a k-cycle is a cycle on k vertices.

Our paper is motivated by the following three existence theorems.

* LAMSADE, Université Paris-Dauphine, Place du Marechal de Lattre de Tassigny, 75775 Paris Cedex 16, France. Email: {bazgan,vdp}@lamsade.dauphine.fr

† Computer and Automation Institute, Hungarian Academy of Sciences, Budapest; and Department of Computer Science, University of Veszprém, Hungary. Email: tuza@sztaki.hu

Theorem 1 ([9]) *Let G be a graph, and $a, b : V(G) \to \mathbb{N}$ two functions such that $d(v) \geq a(v) + b(v) + 1$ for every $v \in V(G)$. Then, there is a nontrivial vertex partition (A, B) of G such that*

$$d_A(v) \geq a(v) \ \ \forall\ v \in A \qquad \text{and} \qquad d_B(v) \geq b(v) \ \ \forall\ v \in B \tag{$\star$}$$

Theorem 2 ([7]) *Let G be a graph, and $a, b : V(G) \to \mathbb{N} \setminus \{0\}$ two functions such that $d(v) \geq a(v)+b(v)$ for every $v \in V(G)$. If G contains no 3-cycles, then there is a nontrivial vertex partition (A, B) of G satisfying $(\star)$.*

Theorem 3 ([4]) *Let G be a graph without 3-cycles and 4-cycles, and $a, b : V(G) \to \mathbb{N} \setminus \{0, 1\}$ two functions such that $d(v) \geq a(v) + b(v) - 1$ for every $v \in V(G)$. Then, there is a nontrivial vertex partition (A, B) of G satisfying $(\star)$.*

Theorems 2 and 3 were originally stated just for constants a, b instead of functions $a(v)$, $b(v)$; but in fact the proofs work for the general case without any changes. This has been observed for Theorem 3 by Gerber and Kobler in [6].

The original proofs of Theorems 1, 2 and 3 contain nonconstructive steps. The goal of the present paper is to show that in all the three cases a vertex partition can be found in polynomial time. More explicitly, the following results will be proved. For short, a nontrivial vertex partition (A, B) of $V(G)$ with the property $(\star)$ is called an (a, b)-*decomposition.*

Theorem 4 *Under the conditions of Theorem 1, an (a, b)-decomposition can be found in polynomial time.*

Theorem 5 *Under the conditions of Theorem 2, an (a, b)-decomposition can be found in polynomial time.*

Theorem 6 *Under the conditions of Theorem 3, an (a, b)-decomposition can be found in polynomial time.*

These results will be proved in Section 3. The next section introduces basic concepts and elementary algorithmic procedures that will be useful in subsequent algorithms. In Section 4, a constructive generalization for k-partitions is also given.

In order to avoid trivialities, it will be assumed throughout that the functions $a, b : V \to \mathbb{N}$ satisfy $a(v) \leq d(v)$ and $b(v) \leq d(v)$ for all vertices $v \in V$. Partitions (A, B) of V will be assumed nontrivial (i.e., $A \neq \emptyset$ and $B \neq \emptyset$) without explicitly mentioning this condition at each occurrence.

2 Preliminaries

In order to find an (a, b)-decomposition in a given graph efficiently, we shall need some more notions.

Let G be a graph and $f : V(G) \to \mathbb{N}$ be a function. Graph G is said to be *f-degenerate* if every nonempty (induced) subgraph H of G contains a vertex $v \in V(H)$ such that $d_H(v) \leq f(v)$. Thus, if G is not $(f-1)$-degenerate, there exists a subset A such that $d_A(v) \geq f(v)$ for each vertex $v \in A$. Such a set A will be called an *f-satisfactory subset*.

Given G and the functions a and b, we shall say that a partition (A, B) of V is an *(a^-, b^-)-partition* if $G[A]$ is $(a-1)$-degenerate and $G[B]$ is $(b-1)$-degenerate; and a partition (A, B) of V is an *(a^+, b^+)-partition* if $G[A]$ is not $(a-1)$-degenerate and $G[B]$ is not $(b-1)$-degenerate.

The next observation is a slight generalization of the known fact that the "coloring number" of a graph can be determined in polynomial time (see [5, 8], that corresponds to $f =$ constant).

Proposition 7 *It is decidable in polynomial time if the subgraph induced by a given set $A \subseteq V$ is f-degenerate, for any given function f. Moreover, if $G[A]$ is not f-degenerate, then a satisfactory $(f+1)$-subset $A' \subseteq A$ can be found in polynomial time.*

Proof: The algorithm consists of iteratively removing vertices v from $G[A]$ of degree less than or equal to $f(v)$ while it is possible. If at the end we obtain a nonempty subgraph $G[A']$, then $G[A]$ is not f-degenerate since $d_{A'}(v) \geq f(v) + 1$, for all $v \in A'$. Thus A' is a $(f+1)$-satisfactory subset.

Conversely, suppose that the algorithm removes all vertices from A. If $G[A]$ were not f-degenerate, then it would contain a $(f+1)$-satisfactory subset A'. The first vertex v of A' considered by the algorithm cannot be removed, since its current degree is greater than or equal to $d_{A'}(v) \geq f(v) + 1$. In this way no vertex of A' could be removed. Thus, if the entire set A gets deleted, then G is f-degenerate.

Since the selection of removable vertices can be performed in polynomial time, the algorithm clearly is polynomial. □

A *minimal f-satisfactory subset* A is an f-satisfactory subset such that, for every proper subset $A' \subset A$, there exists a vertex $v \in A'$ with $d_{A'}(v) \leq f(v) - 1$.

Proposition 8 *If a graph G contains an f-satisfactory subset, then a minimal f-satisfactory subset can be found in polynomial time.*

Proof: Let A_1 be an f-satisfactory subset of G. We construct a sequence $G[A_1], ..., G[A_t]$ of subgraphs of G, such that $A_{i+1} \subset A_i$ and each $G[A_i]$ has the property that $d_{A_i}(v) \geq f(v)$, for all $v \in A_i$.

In step i ($i \geq 1$), we select a vertex $v \in A_i$, tentatively remove it from A_i, and iteratively remove from $A_i \setminus \{v\}$ the vertices v' whose degree in the current induced subgraph is less than $f(v')$, until we obtain a set X which is either empty or an f-satisfactory subset. If

$X = \emptyset$, then we iterate the previous procedure for vertices $v \in A_i$ until a set $X \neq \emptyset$ is obtained, and in this case we continue the construction with $A_{i+1} = X$. In the other case, i.e. when all the sets X obtained are empty, we stop the algorithm with $t = i$.

We claim that A_t is a minimal f-satisfactory subset. Since A_t is clearly an f-satisfactory subset, it remains to prove that it is minimal. Suppose on the contrary, that there exists a proper subset $A' \subset A_t$ such that $d_{A'}(v) \geq f(v)$ for all $v \in A'$. Choose any $v' \in A_t \setminus A'$. Continuing the procedure from A_t by removing v', a nonempty set $A_{t+1} \supseteq A'$ would be generated, which contradicts the previous assumption that $A_{t+1} = \emptyset$ holds for all $v' \in A_t$. □

Now, we establish a constructive proof of sufficiency of a necessary and sufficient condition given by Stiebitz in order to obtain an (a, b)-decomposition. (In [9] the next proposition was formulated with the stronger condition $d(v) \geq a(v) + b(v) + 1$ but its proof works for the weaker condition as well). It refers to the concept of *feasible pair* which is a pair (A, B) of disjoint, nonempty vertex subsets $A, B \subseteq V(G)$ such that $d_A(v) \geq a(v)$ for all $v \in A$ and $d_B(v) \geq b(v)$ for all $v \in B$.

Proposition 9 *Let $G = (V, E)$ be a graph and $a, b : V \to \mathbb{N}$ integer-valued functions such that $d(v) \geq a(v) + b(v) - 1$ for every $v \in V(G)$. If a feasible pair (A, B) is given, then an (a, b)-decomposition of G exists and can be found in polynomial time.*

Proof: Let $V_1 = A$ and $V_2 = B$. While there is a vertex v in $V \setminus (V_1 \cup V_2)$ such that $d_{V_1}(v) \geq a(v)$, add v in V_1. While there is a vertex v in $V \setminus (V_1 \cup V_2)$ such that $d_{V_2}(v) \geq b(v)$, add v in V_2. At the end, if $C = V \setminus (V_1 \cup V_2) \neq \emptyset$, then $d_{V_1}(v) < a(v)$ and $d_{V_2}(v) < b(v)$ for any $v \in C$. Recalling the condition $d(v) \geq a(v) + b(v) - 1$, for each $v \in C$ before the last step we have $d_{V_1 \cup C}(v) \geq a(v)$ and $d_{V_2 \cup C}(v) \geq b(v)$. Thus, it is indeed feasible to add all vertices of C either to V_1 or to V_2, and hence an (a, b)-decomposition is obtained. □

The importance of (a^+, b^+)-partitions is demonstrated by the following observation that is a direct consequence of Propositions 7 and 9.

Corollary 10 *If an (a^+, b^+)-partition is given, then an (a, b)-decomposition of G exists and can be found in polynomial time.*

This gives rise to the procedure DECOMPOSE described in Algorithm 1.

3 Finding a decomposition efficiently

By Corollary 10, the problem of efficiently finding an (a, b)-decomposition is reduced to that of finding an (a^+, b^+)-partition. This fact applies to all the three theorems to be proved. Despite several similarities, however, there are remarkable differences among the three algorithms. For this reason, they will be presented in separate subsections. Before that, we describe some general principles applied throughout.

Algorithm 1 DECOMPOSE(A, B)

Require: an (a^+, b^+)-partition (A, B) of V
Ensure: an (a, b)-decomposition (V_1, V_2)
  while there is a vertex $v \in A$ such that $d_A(v) \leq a(v) - 1$ **do**
    $A \leftarrow A \setminus \{v\}$
  while there is a vertex $v \in B$ with $d_B(v) \leq b(v) - 1$ **do**
    $B \leftarrow B \setminus \{v\}$
  // (A, B) is a feasible pair //
  $V_1 \leftarrow A$; $V_2 \leftarrow B$
  while there is a vertex v in $V \setminus (V_1 \cup V_2)$ such that $d_{V_1}(v) \geq a(v)$ **do**
    $V_1 \leftarrow V_1 \cup \{v\}$
  while there is a vertex v in $V \setminus (V_1 \cup V_2)$ such that $d_{V_2}(v) \geq b(v)$ **do**
    $V_2 \leftarrow V_2 \cup \{v\}$
  $V_1 \leftarrow V \setminus V_2$ (or $V_2 \leftarrow V \setminus V_1$)

After some preprocessing, each of the algorithms iteratively maintains a vertex partition (A, B), whose value will be measured by the function

$$w(A, B) = |E(G[A])| + |E(G[B])| + \sum_{v \in A} b(v) + \sum_{v \in B} a(v)\,.$$

After an iteration, either an (a, b)-decomposition is found, or the value of w is increased. Since $|E(G[A])| + |E(G[B])| \leq |E(G)|$, and $\max(a(v), b(v)) \leq d(v)$ for all $v \in V(G)$ by assumption, we always have $w(A, B) \leq 3\,|E(G)|$, so that the algorithms terminate after a linear number of iterations (in terms of the input size). Thus, it will suffice to prove that the algorithms are sound and that each of their steps is implementable in polynomial time.

3.1 Unrestricted graphs with $d(v) \geq a(v) + b(v) + 1$

In this subsection, we consider the algorithmic version of the theorem of Stiebitz.

Proof of Theorem 4 : We are going to prove that Algorithm 2 determines an (a, b)-decomposition in polynomial time, whenever the functions a, b satisfy $d(v) \geq a(v)+b(v)+1$ for all vertices v.

Using V as an initial a-satisfactory subset and applying Proposition 8, the first step is computable in polynomial time. Also, using Proposition 7, the **while** conditions are polynomial-time decidable. We justify that in the **while** loop the selection of a vertex v is always possible. For this, we show that at the beginning of each iteration in the **while** loop, $G[A]$ and $G[B]$ are a- and b-degenerate, respectively. Before entering the **while** loop, $G[A]$ is not $(a-1)$-degenerate, but it is a-degenerate since A is a minimal a-satisfactory subset. We enter the **while** loop only if $G[B]$ is $(b-1)$-degenerate, which means that each of its subgraphs contains a vertex v' of degree at most $b(v') - 1$. At the end of an iteration of the **while** loop, after moving v from A to B for example, the degree of vertices of B

Algorithm 2 Determination of a decomposition; $d(v) \geq a(v) + b(v) + 1$

Require: a graph G such that $d(v) \geq a(v) + b(v) + 1$ for every $v \in V(G)$
Ensure: an (a, b)-decomposition (V_1, V_2)
1: Find $A \subseteq V$, a minimal a-satisfactory subset
2: $B \leftarrow V \setminus A$
3: **while** $G[A]$ is $(a-1)$-degenerate or $G[B]$ is $(b-1)$-degenerate **do**
4: **if** $G[B]$ is $(b-1)$-degenerate **then**
5: Let $v \in A$ such that $d_A(v) \leq a(v)$
6: $A \leftarrow A \setminus \{v\}$; $B \leftarrow B \cup \{v\}$
7: **else**
8: **if** $G[A]$ is $(a-1)$-degenerate **then**
9: Let $v \in B$ such that $d_B(v) \leq b(v)$
10: $A \leftarrow A \cup \{v\}$; $B \leftarrow B \setminus \{v\}$
11: DECOMPOSE(A, B)

in $G[B \cup \{v\}]$ increases with at most one, so in each subgraph of $G[B \cup \{v\}]$ still there is a vertex v' of degree at most $b(v')$. Therefore $G[A \setminus \{v\}]$ and $G[B \cup \{v\}]$ remain a- and b-degenerate, respectively. Hence, the operations inside the loop can always be performed.

Suppose that $A \neq \emptyset$, $B \neq \emptyset$ at the beginning of an iteration of the **while** loop. This is certainly valid when we first enter the loop, right after Step 2. Say B is $(b-1)$-degenerate. Then some $v' \in B$ has $d_A(v') \geq a(v') + 2$, hence $|A| \geq 2$ and after the move of v from A to B, both A and B remain nonempty. It follows that the partition generated by Algorithm 2 is nontrivial.

What remains to show is that the value of $w(A, B)$ increases after each execution of the **while** loop. Consider any iteration. Assume, without loss of generality, that $G[B]$ is $(b-1)$-degenerate. By the choice of v, since $d_A(v) \leq a(v)$, we have $d_B(v) \geq b(v)+1$. Thus,

$$w(A \setminus \{v\}, B \cup \{v\}) - w(A, B) = d_B(v) - d_A(v) + a(v) - b(v) \geq 1$$

and we obtain a partition $(A \setminus \{v\}, B \cup \{v\})$ with a larger value of w. □

3.2 Triangle-free graphs with $d(v) \geq a(v) + b(v)$

From now on, we assume that the input graph contains no 3-cycles.

Proof of Theorem 5 : We present Algorithm 3 that finds the required decomposition. The algorithm maintains a vertex partition (A, B) of the input graph $G = (V, E)$ with the following properties:

(1) A is an a-satisfactory subset,

(2) A contains a vertex v_1 such that $d_A(v_1) = a(v_1)$,

(3) $G[A \setminus \{v_1\}]$ is $(a-1)$-degenerate.

Algorithm 3 Determination of a decomposition; triangle-free, $d(v) \geq a(v) + b(v)$

Require: a triangle-free graph G such that $d(v) \geq a(v) + b(v)$ for every $v \in V(G)$
Ensure: an (a, b)-decomposition (V_1, V_2)
1: Find $A \subseteq V$, a minimal a-satisfactory subset
2: $B \leftarrow V \setminus A$
3: **while** $G[B]$ is $(b-1)$-degenerate **do**
4: Let $x \in B$ such that $d_B(x) < b(x)$
5: $A \leftarrow A \cup \{x\}$; $B \leftarrow B \setminus \{x\}$
6: **while** there is $y \in A$ such that $d_A(y) \leq a(y)$ and $G[A \setminus \{y\}]$ is not $(a-1)$-degenerate **do**
7: $A \leftarrow A \setminus \{y\}$; $B \leftarrow B \cup \{y\}$
8: DECOMPOSE(A, B)

First observe that (1)–(3) imply the existence of a vertex $v_2 \in A$ such that $v_1v_2 \in E$, $d_A(v_2) = a(v_2)$ and $G[A \setminus \{v_2\}]$ is $(a-1)$-degenerate. Indeed, (3) implies the existence of a vertex $v_2 \in A \setminus \{v_1\}$ of degree at most $a(v_2) - 1$ in $A \setminus \{v_1\}$. Moreover, because of (1) we know that $d_A(v_2) \geq a(v_2)$. Thus, v_1v_2 must be an edge and $d_A(v_2) = a(v_2)$. In order to prove that $G[A \setminus \{v_2\}]$ is $(a-1)$-degenerate, consider any subset $A' \subseteq A \setminus \{v_2\}$. If $v_1 \in A'$, then $d_{A'}(v_1) \leq a(v_1) - 1$ since $d_A(v_1) = a(v_1)$ from (2) and $v_1v_2 \in E$. Otherwise, since $A' \subset A \setminus \{v_1\}$ and $G[A \setminus \{v_1\}]$ is $(a-1)$-degenerate, there exists $v \in A'$ such that $d_{A'}(v) \leq a(v) - 1$.

We establish now that maintaining properties (1)–(3) ensures that (A, B) is a nontrivial partition. We know that A contains at least two vertices v_1 and v_2. Moreover, $|B| \geq 2$ because v_1 and v_2 together have at least $b(v_1) + b(v_2) \geq 2$ neighbors in B and they do not have a common neighbor since G is triangle-free.

Clearly, all steps of the algorithm are feasible and can be performed in polynomial time. We show now that conditions (1)–(3) are satisfied after the preprocessing stage (Steps 1,2) and are maintained after each iteration of the main **while** loop.

In the preprocessing stage, the selection of a *minimal* a-satisfactory subset ensures that there exists at least one vertex v_1 with $d_A(v_1) = a(v_1)$, for otherwise removing any one vertex, the subset would still be a-satisfactory. Minimality of A also ensures that $G[A \setminus \{v_1\}]$ is $(a-1)$-degenerate. Thus, conditions (1)–(3) are satisfied.

Assuming now that conditions (1)–(3) are satisfied at the beginning of an iteration of the main **while** loop, we show that they remain satisfied at the end of this iteration.

Regarding (1), A clearly remains a-satisfactory after Step 5. At the end of a possible execution of the internal **while** loop (Steps 6-7), $G[A]$ is not $(a-1)$-degenerate, which means that it contains a subset A' which is a-satisfactory. Moreover, the existence of a vertex $y \in A \setminus A'$ such that $d_A(y) < a(y)$ is not possible, for otherwise it would be removed in a new iteration of this **while** loop. Therefore, A is a-satisfactory.

Regarding (2), vertex x selected at Step 4 cannot be adjacent both to v_1 and v_2, G being triangle-free. Assuming $v_1x \notin E$, (3) implies that $G[(A \setminus \{v_1\}) \cup \{x\}]$ is a-degenerate, that implies in turn that $G[A \cup \{x\}]$ is a-degenerate, since $d_{A \cup \{x\}}(v_1) = a(v_1)$. After Step

5, $G[A]$ is thus a-degenerate, and remains a-degenerate after a possible execution of the internal **while** loop. Considering that A is a-satisfactory, this implies the existence of a vertex v'_1 with $d_A(v'_1) = a(v'_1)$.

Finally, condition (3) is satisfied since otherwise v_1 would have been removed from A at an iteration of the internal **while** loop.

In order to complete the proof, we show that $w(A, B)$ increases at each iteration of the main **while** loop. We need to investigate those steps where (A, B) is or may be modified, namely Steps 5 and 7. When x is deleted from B, $|E(G[B])|$ decreases by at most $b(x) - 1$ and $\sum_{v\in B} a(v)$ by exactly $a(x)$. Inserting x into A increases $|E(G[A])|$ by at least $a(x) + 1$ and $\sum_{v\in A} b(v)$ by exactly $b(x)$. Thus, in Step 5, $w(A, B)$ increases by at least 2. Moreover, moving y from A to B (Step 7) does not decrease $w(A, B)$. Therefore, at each iteration of the main **while** loop, $w(A, B)$ increases by at least 2, which ensures that the number of iterations within this loop is polynomial. □

3.3 Graphs of girth at least 5 with $d(v) \geq a(v) + b(v) - 1$

Here we consider graphs without 3-cycles and 4-cycles.

Before presenting the algorithm, it should be noted that if a graph admits an (a, b)-decomposition, then so does any graph obtained from it by edge insertions. This observation leads to the notion of *tight vertex*, that is a vertex v such that $d(v) = a(v) + b(v) - 1$.

Proof of Theorem 6 : Algorithm 4 operates on (a^-, b^-)-partition rather than just on general subsets or partitions. Namely, having an (a^-, b^-)-partition at hand, the main loop of the algorithm will either find an (a^+, b^+)-partition or generate an (a^-, b^-)-partition with a larger value of w.

Observe that for any (a^-, b^-)-partition (A, B), we have $|A| \geq 2$ and $|B| \geq 2$, since assuming that A is nonempty, it contains a vertex x with $d_A(x) \leq a(x) - 1$ which has at least $b(x) \geq 2$ neighbors in B; B being also nonempty, it contains a vertex y with at least $a(y) \geq 2$ neighbors in A.

In the preprocessing stage, we first reduce the input graph to a spanning subgraph minimal with respect to the degree constraints $d(v) = a(v) + b(v) - 1$ (Steps 2-4). If (A, B) defined in Steps 5-6 is not an (a^+, b^+)-partition, an (a^-, b^-)-partition is created in Steps 10-11. In Step 10 the required vertex u exists since, as proved for the preprocessing stage in Section 3.2, a minimal a-satisfactory subset contains two adjacent vertices, say u and v, with $d_A(u) = a(u)$ and $d_A(v) = a(v)$. Because of the preliminary graph reduction, at least one of u and v is tight. In Step 11 an (a^-, b^-)-partition is created because, $G[B]$ being $(b - 1)$-degenerate, $G[B \cup \{u\}]$ remains $(b - 1)$-degenerate, since $d_B(u) = b(u) - 1$.

In the main part of the algorithm (Steps 12-38), sets A^- and B^- defined in Steps 13 and 20 are both non-empty since they are always defined from an (a^-, b^-)-partition (A, B). After the execution of the **while** loop (Steps 14-20), for every $u \in A^-$, $G[B \cup \{u\}]$ is not $(b - 1)$-degenerate since $G[A \setminus \{u\}]$ is necessarily $(a - 1)$-degenerate. Similarly, for every $v \in B^-$, $G[A \cup \{v\}]$ is not $(a - 1)$-degenerate. Thus, as expressed by Steps 21-22, if $uv \notin E$ for some $u \in A^-$ and $v \in B^-$, then partition $((A \cup \{v\}) \setminus \{u\}, (B \cup \{u\}) \setminus \{v\})$ is an

Algorithm 4 Determination of a decomposition; girth ≥ 5, $d(v) \geq a(v) + b(v) - 1$

Require: a graph G of girth ≥ 5 such that $d(v) \geq a(v) + b(v) - 1$ for every $v \in V(G)$
Ensure: an (a, b)-decomposition (V_1, V_2)

found $\leftarrow$ false //When true indicates that (A, B) is an (a^+, b^+)-partition.//
for all edges $uv \in E$ **do**
  if neither u nor v is tight **then**
    $E \leftarrow E \setminus \{uv\}$
Find $A \subseteq V$, a minimal a-satisfactory subset
$B \leftarrow V \setminus A$
if $G[B]$ is not $(b-1)$-degenerate **then**
  found $\leftarrow$ true
else
  Select a tight vertex $u \in A$ such that $d_A(u) = a(u)$
  $A \leftarrow A \setminus \{u\}$; $B \leftarrow B \cup \{u\}$
  repeat
    $A^- \leftarrow \{u \in A : d_A(u) < a(u)\}$; $B^- \leftarrow \{v \in B : d_B(v) < b(v)\}$
    while there exists $u \in A^-$ or $v \in B^-$ such that $(A \setminus \{u\}, B \cup \{u\})$ or $(A \cup \{v\}, B \setminus \{v\})$ is an (a^-, b^-)-partition **do**
      Select one such u or v
      if the selected vertex is u **then**
        $A \leftarrow A \setminus \{u\}$; $B \leftarrow B \cup \{u\}$
      else
        $A \leftarrow A \cup \{v\}$; $B \leftarrow B \setminus \{v\}$
      $A^- \leftarrow \{u \in A : d_A(u) < a(u)\}$; $B^- \leftarrow \{v \in B : d_B(v) < b(v)\}$
    if $uv \notin E$ for some $u \in A^-$ and $v \in B^-$ **then**
      $A \leftarrow (A \setminus \{u\}) \cup \{v\}$; $B \leftarrow (B \setminus \{v\}) \cup \{u\}$; *found* $\leftarrow$ true
    else
      Select $u' \in A \setminus A^-$ such that $d_A(u') = d_{A \setminus A^-}(u') + 1 = a(u')$
      Select $v' \in B \setminus B^-$ such that $d_B(v') = d_{B \setminus B^-}(v') + 1 = b(v')$
      Select u the unique neighbor of u' in A^-
      Select v the unique neighbor of v' in B^-
      $A \leftarrow (A \setminus \{u\}) \cup \{v\}$; $B \leftarrow (B \setminus \{v\}) \cup \{u\}$
      if $G[A]$ is $(a-1)$-degenerate **then**
        if u' is tight **then**
          $A \leftarrow (A \setminus \{u'\}) \cup \{u\}$; $B \leftarrow (B \setminus \{u\}) \cup \{u'\}$
      else
        if $G[B]$ is $(b-1)$-degenerate **then**
          if v' is tight **then**
            $A \leftarrow (A \setminus \{v\}) \cup \{v'\}$; $B \leftarrow (B \setminus \{v'\}) \cup \{v\}$
        else
          found $\leftarrow$ true
  until *found* = true
DECOMPOSE(A, B)

(a^+, b^+)-partition. If we cannot find such a non-edge, $G[A^- \cup B^-]$ is a complete bipartite induced subgraph of G, because A^- and B^- are independent sets since G is triangle-free. Moreover, each $u' \in A \setminus A^-$ and $v' \in B \setminus B^-$ has at most one neighbor in A^- and B^-, respectively, as G does not contain 4-cycles. (Also, $\min(|A^-|, |B^-|) = 1$ holds, but this fact will not be used.)

We justify now the feasibility of Steps 24-27. We first establish that $A \neq A^-$. After the execution of the **while** loop (Steps 14-20), we know that for every $v \in B^-$, $G[A \cup \{v\}]$ is not $(a-1)$-degenerate, and thus contains at least one a-satisfactory subset. Assuming that $A = A^-$, A would be an independent set and any a-satisfactory subset would be a star centered on v, which is impossible because $a \geq 2$. Similarly, we have $B \neq B^-$.

Concerning the feasibility of Steps 24 and 26, $G[A]$ being $(a-1)$-degenerate, every subset of A, and in particular subset $A \setminus A^-$, contains a vertex u' such that $d_{A \setminus A^-}(u') \leq a(u') - 1$. But, since $u' \in A \setminus A^-$, we have $d_A(u') \geq a(u')$, hence, recalling that u' has at most one neighbor in A^-, we deduce that $d_A(u') = a(u')$ and u' has exactly one neighbor in A^-. Similarly, we can justify the existence of vertices v' and v in Steps 25 and 27, which together with u' and u form an induced P_4 $u'uvv'$.

We consider now the situation after Step 27. The remaining steps before the end of the **repeat** loop (Steps 28-37) aim at constructing either an (a^+, b^+)-partition or an (a^-, b^-)-partition (A, B) with the additional property that there exists a non-edge uv with $u \in A^-$ and $v \in A^-$. The reason for imposing this property is that, in the next iteration of the **repeat** loop, if the **while** loop (Steps 14-20) is not executed, then Steps 21-22 will successfully terminate the algorithm.

Before examining the different cases, we point out that assignments performed at Step 28 are temporary assignments that will remain valid in all cases except two, where A and B will be modified again taking into account the temporary assignments (in Steps 31 and 35).

The first case is the successful one when $G[(A \setminus \{u\}) \cup \{v\}]$ and $G[(B \setminus \{v\}) \cup \{u\}]$ are respectively not $(a-1)$- and not $(b-1)$-degenerate, creating an (a^+, b^+)-partition. This case is taken into account implicitly in Steps 28 and 37.

Consider now the case where $G[(A \setminus \{u\}) \cup \{v\}]$ is $(a-1)$-degenerate and, knowing that $G[A \cup \{v\}]$ is not $(a-1)$-degenerate, we have $d_{A \cup \{v\}}(u) \geq a(u)$, and thus $d_A(u) \geq a(u) - 1$, which implies $d_A(u) = a(u) - 1$, since $u \in A^-$. Assuming now that u is tight, we get $d_B(u) = b(u)$ and thus $d_{(B \setminus \{v\}) \cup \{u\}}(u) = b(u) - 1$. Since $G[B \cup \{u\}]$ is not $(b-1)$-degenerate, whereas $G[B]$ is $(b-1)$-degenerate, u belongs to all satisfactory subsets of $B \cup \{u\}$. Considering that $d_{(B \setminus \{v\}) \cup \{u\}}(u) = b(u) - 1$, we deduce that $G[(B \setminus \{v\}) \cup \{u\}]$ is $(b-1)$-degenerate. Symmetrically, if we assume that $G[(B \setminus \{v\}) \cup \{u\}]$ is $(b-1)$-degenerate and v is tight, we can prove that $G[(A \setminus \{u\}) \cup \{v\}]$ is $(a-1)$-degenerate.

In the two above cases, $((A \setminus \{u\}) \cup \{v\}, (B \setminus \{v\}) \cup \{u\})$ is an (a^-, b^-)-partition, and we can proceed to the next iteration of the **repeat** loop, observing that after swapping u and v we create a non-edge $u'v'$ where u' and v' belong respectively to the updated sets A^- and B^-, since after the swap both u' and v' lose one neighbor. These two cases are taken into account implicitly in Step 28, since A and B are not modified afterwards if we are in one of these two cases.

Finally, we consider the case where $G[(A \setminus \{u\}) \cup \{v\}]$ is $(a-1)$-degenerate but u is not tight. In this case, corresponding to Steps 29-31, u' is tight. Observe that if we modify set $(A \setminus \{u\}) \cup \{v\}$ by adding vertex u and removing vertex u', the resulting induced graph $G[(A \setminus \{u'\}) \cup \{v\}]$ is also $(a-1)$-degenerate since $d_{(A\setminus\{u'\})\cup\{v\}}(u) \leq a(u) - 1$. Moreover, u' being tight, $d_{B\setminus\{v\}}(u') = b(u') - 1$, hence $G[(B \setminus \{v\}) \cup \{u'\}]$ is also $(b-1)$-degenerate. Therefore $((A\setminus\{u'\})\cup\{v\}, (B\setminus\{v\})\cup\{u'\})$, which corresponds to the assignments performed in Step 31, is an (a^-, b^-)-partition with the non-edge uv'.

The symmetrical case where $G[(B \setminus \{v\}) \cup \{u\}]$ is $(b-1)$-degenerate and v' is tight is handled in Steps 33-35. Here, we can prove as before that partition $((A \setminus \{u\}) \cup \{v'\}, (B \setminus \{v'\}) \cup \{u\})$ is an (a^-, b^-)-partition with the non-edge $u'v$.

In order to complete the proof, we need to establish that $w(A, B)$ increases at each iteration of the **repeat** loop in order to guarantee a polynomial number of iterations. Actually, if A and B are modified in the **while** loop (Steps 14-20) it is easy to show that $w(A, B)$ increases by at least one. However, when A and B are modified in Steps 28, 31 and 35, we can only ensure that $w(A, B)$ does not decrease. This shows that if the **while** loop (Steps 14-20) is not executed then $w(A, B)$ may not increase. But, except for the first iteration of the **repeat** loop, if the previous **while** loop is not executed, we proved previously that Steps 21-22 will apply, terminating the algorithm. Therefore, the algorithm is polynomial since each of its steps can be obviously performed in polynomial time. □

4 Partitions into more than two classes

Stiebitz has observed that Theorem 1 implies the following result by induction:

Corollary 11 ([9]) *Let G be a graph, and $f_1, \ldots, f_k : V(G) \to \mathbb{N}$ be $k \geq 2$ functions. Assume that $d_G(v) \geq f_1(v) + \ldots + f_k(v) + k - 1$ for every vertex $v \in V(G)$. Then there is a partition $(A_1, \ldots, A_k)$ of $V(G)$ into k nonempty subsets such that*

$$d_{A_i}(v) \geq f_i(v) \qquad \forall\ 1 \leq i \leq k, \quad \forall\ v \in A_i$$

A partition $(A_1, \ldots, A_k)$ of $V(G)$ into k nonempty subsets such that $d_{A_i}(v) \geq f_i(v)$ for all $1 \leq i \leq k$ and all $v \in A_i$ is called an $(f_1, \ldots, f_k)$-*decomposition.*

We can also make a constructive proof for this result.

Theorem 12 *Given an input graph G, an integer $k = k(n) \geq 2$ (i.e., possibly depending on the number n of vertices), and k functions $f_1, \ldots, f_k : V(G) \to \mathbb{N}$ such that $d_G(v) \geq f_1(v) + \ldots + f_k(v) + k - 1$ for all $v \in V(G)$, an $(f_1, \ldots, f_k)$-decomposition can be found in polynomial time.*

Proof: The required k-partition can be obtained by applying Algorithm 5. Observe that at each of the $k-1$ iterations of Algorithm 5, $G[B]$ and the functions a and b satisfy the conditions of Theorem 1, thus allowing the execution of Algorithm 2 that runs in polynomial time. □

In a similar way, the following results can also be derived:

Algorithm 5 Determination of a k-partition

Require: a graph G such that $d_G(v) \geq f_1(v)+\ldots+f_k(v)+k-1$ for every vertex $v \in V(G)$.
Ensure: an $(f_1, \ldots, f_k)$-decomposition $(V_1, \ldots, V_k)$
  $B \leftarrow V(G)$
  for $i \leftarrow 1$ to $k-1$ **do**
    Define $a(v) = f_i(v)$ and $b(v) = f_{i+1}(v) + \ldots + f_k(v) + k - (i+1)$ for all $v \in B$
    Use Algorithm 2 on $G[B]$ to determine an (a, b)-decomposition (V_i, B')
    $B \leftarrow B'$
  $V_k \leftarrow B'$

Theorem 13 *Given a triangle-free input graph G, an integer $k = k(n) \geq 2$, and k functions $f_1, \ldots, f_k : V(G) \to \mathbb{N} \setminus \{0\}$ such that $d_G(v) \geq f_1(v) + \ldots + f_k(v)$ for all $v \in V(G)$, an $(f_1, \ldots, f_k)$-decomposition can be found in polynomial time.*

Theorem 14 *Given an input graph G of girth at least five, an integer $k = k(n) \geq 2$, and k functions $f_1, \ldots, f_k : V(G) \to \mathbb{N} \setminus \{0, 1\}$ such that $d_G(v) \geq f_1(v) + \ldots + f_k(v) - k + 1$ for all $v \in V(G)$, an $(f_1, \ldots, f_k)$-decomposition can be found in polynomial time.*

5 Conclusions

It remains an open problem to determine tight asymptotics for the running time of a fastest algorithm that determines an (a, b)-decomposition (with or without assumptions on the girth of the input graph). In particular, it is not known so far whether all such algorithms run in superlinear time in the worst case.

The strength and applicability of the three Algorithms 2, 3 and 4 are different. For example, with some modifications, the scheme of Algorithm 3 can be adjusted to obtain an alternative solution for Theorem 4. We omit the details, however, since it is not the goal of the present paper to describe more than one algorithm for any of these problems.

For the particular functions $a(v) = \lceil \frac{d(v)-1}{2} \rceil$ and $b(v) = \lceil \frac{d(v)-2}{2} \rceil$, Theorem 4 yields a polynomial-time search algorithm. These functions a, b are essentially the largest possible ones in the sense that — as proved in [2] — it is NP-complete to decide whether there exists an (a, b)-decomposition for $a(v) = b(v) = \lceil \frac{d(v)}{2} \rceil$ in an unrestricted input graph. On the other hand, imposing further conditions, the situation may become different. For instance, on 4-regular graphs, for $a = b = 2$ a linear-time solution can be given [1], while for $a = b = 3$ the problem is NP-complete [3].

Acknowledgements: This research was supported by the bilateral research cooperation Balaton between EGIDE (France) and Ministry of Education (Hungary) under grant numbers 82/221580 and F-29/2003. The second author was also supported in part by the Hungarian Scientific Research Fund, grant OTKA T-042710.

References

[1] C. Bazgan, Zs. Tuza and D. Vanderpooten, *On the existence and determination of satisfactory partitions in a graph*, Proceedings of the 14th Annual International Symposium on Algorithms and Computation (ISAAC 2003), LNCS 2906, 444–453.

[2] C. Bazgan, Zs. Tuza and D. Vanderpooten, *The satisfactory partition problem*, submitted.

[3] V. Chvátal, *Recognizing decomposable graphs*, Journal of Graph Theory 8 (1984), 51–53.

[4] A. Diwan, *Decomposing graphs with girth at least five under degree constraints*, Journal of Graph Theory 33 (2000), 237–239.

[5] H.-J. Finck and H. Sachs, *Über eine von H. S. Wilf angegebene Schranke für die chromatische Zahl endlicher Graphen*, Mathematische Nachrichten 39 (1969), 373–386.

[6] M. Gerber and D. Kobler, *Classes of graphs that can be partitioned to satisfy all their vertices*, Australasian Journal of Combinatorics, 29 (2004), 201–214.

[7] A. Kaneko, *On decomposition of triangle-free graphs under degree constraints*, Journal of Graph Theory 27 (1998), 7–9.

[8] D. W. Matula, *A min-max theorem for graphs with application to graph coloring*, SIAM Review 10 (1968), 481–482.

[9] M. Stiebitz, *Decomposing graphs under degree constraints*, Journal of Graph Theory 23 (1996), 321–324.