Balanced Network Flows. I. A Unifying Framework for Design and Analysis of Matching Algorithms

Christian Fremuth-Paeger, Dieter Jungnickel

Lehrstuhl für Diskrete Mathematik, Optimierung und Operations Research, University of Augsburg, D-86135 Augsburg, Germany

Received 8 December 1997; accepted 29 April 1998

Abstract: We discuss a wide range of matching problems in terms of a network flow model. More than this, we start up a matching theory which is very intuitive and independent from the original graph context. This first paper contains a standardized theory for the performance analysis of augmentation algorithms in a wide area of matching problems. Several optimality criteria are given which do not use cuts or barriers. As an application of our theory, the known cardinality matching algorithms of Edmonds, Kameda and Munro, and Micali and Vazirani, and the algorithm of Kocay and Stone for capacitated matching problems can be studied in their effects. From our theory a c-capacitated b-matching algorithm can be derived that behaves like the Dinic algorithm for the maximum flow problem. It will turn out that techniques for the maximum flow problem can be applied to matching problems much more explicitly than done before. A comprehensive duality theory depending on the network flow model used here will follow. Explicit algorithms for nonweighted problems will be presented in subsequent papers. Networks 33: 1–28, 1999

Keywords: capacitated matching problems; network flows; balanced flow networks; skew-symmetric graphs; antisymmetrical digraphs; augmenting a matching; blossoms; nuclei; layered auxiliary networks

1. PRELIMINARIES

The best-known and best-understood matching problem is the **cardinality matching problem:** For a given graph (V, E), a set $M \subseteq E$ of nonadjacent edges, called a **matching,** of maximum cardinality has to be found. The most popular techniques for solving this problem were established by Berge [3], Edmonds [7], Hopcroft and Karp [16], and Micali and Vazirani [23] long ago. Here, linear programming methods are not considered.

All known cardinality matching algorithms are based on the method of augmentation of a given matching M: An alternating path, called the **augmenting path,** is chosen which starts and ends with nodes not in M. Here, an **alternating path** is a path which traverses edges of M and $E \setminus M$ in alternation. Once an augmenting path p is found, we obtain a matching $\tilde{M}$ of cardinality $|\tilde{M}| = |M| + 1$ by deleting all edges from M that are also on p and adding all edges on p that are not in M. This step is called an **augmentation** of M. Any matching that

Correspondence to: D. Jungnickel; e-mail: jungnickel@math.uni-augsburg.de

The major results of this paper form part of the first author's doctoral thesis which has been written under the supervision of the second author.

AMS subject classification: 05C70, 90B10, 90C35

 CCC 0028-3045/99/010001-28

cannot be augmented was proven to be maximum by Berge [3]. We will present analogous results for the capacitated matching problems which are introduced in the next section.

Many notions used in algorithmic matching theory like *blossoms* and *blossom bases* were introduced by Edmonds [7]. He devised the method of *shrinking blossoms,* which reduces the problem of augmentation in arbitrary graphs to augmentation in bipartite graphs. Since finding an augmenting path in bipartite graphs is very easy, this resulted in the first efficient cardinality matching algorithm.

Hopcroft and Karp [16] gave upper bounds on the number of augmentation steps under the assumption that all occurring augmenting paths have minimum length. Any algorithm satisfying this assumption is called **phase-ordered.** At this point, the perception of bipartite matching problems began to differ from that of general graph matching problems:

Concerning bipartite graphs, Hopcroft and Karp [16] and Even and Tarjan [8] improved the complexity of the augmentation algorithm to $O(n^{5/2})$ and $O(\sqrt{n}m)$, respectively. The latter algorithm is still state of the art. Furthermore, Even and Tarjan recognized that the bipartite matching problems can be mapped onto the max-flow problem for a certain class of networks and that their cardinality matching algorithm was equivalent to the max-flow algorithm of Dinic running on these networks.

However, the method of explicitly shrinking blossoms is unable to obtain a minimum-length augmenting path in general, since the length of alternating paths changes within those shrinking operations. Micali and Vazirani [23] devised an algorithm that uses an appropriate search strategy and a set union mechanism instead of explicit shrinking. This algorithm runs in $O(\sqrt{n}m)$-time, exactly the bound known for the bipartite case.

Since the first comprehensive correctness proof of this algorithm was published by Vazirani [28] only recently, his ideas did not reappear in any compendium on matching theory yet. We rework Vazirani's results, but in a different logical arrangement for three reasons: to make them applicable to a wider range of problems, to obtain a standardization of the analysis and notions in algorithmic matching theory, and to get more useful and intuitive intermediate results.

Our approach is based on two recent papers by Kocay and Stone [19, 20], who described a class of integral flow networks, called *balanced* networks, that represent generalized matching problems, an adaptation of the famous max-flow min-cut theorem to balanced networks and an algorithm for finding maximum matchings in terms of these networks. As we will see, this is indeed the suitable framework for the study of matching algorithms, since it unifies the setting for nearly all important matching algorithms proposed up to now.

Throughout the paper, we will use standard terminology from network flow theory without further explanation; we refer the reader to Ahuja et al. [1] or Jungnickel [17] for background.

2. GENERALIZED MATCHING PROBLEMS

There are several directions of generalizing the cardinality matching problem. In this paper, we will not treat weighted matching problems which have been studied by many authors before. We, rather, look at multigraphs instead of simple graphs and a larger class of subgraphs than independent edge sets. Our framework includes the notions of k-factors, f-factors, and (f, g)-factors.

A **multigraph** is a triple $G := (V, E, c)$ consisting of a **node set** V, an **edge set** $E \subseteq \{\{u, v\} : u, v \in V\}$, and a capacity constraint $c : E \to \mathbf{N}$, where $c(e)$ denotes the **multiplicity** of the edge e, that is the number of edges **parallel** to e. Note that this definition allows edges of the form $\{u\} = \{u, u\} \in E$, called **loops.**

Any mapping $x : E \to \mathbf{N}_0$ with $x \leq c$ is called a **subgraph** of G. The **degree sequence** of x in G is given by

$$deg_x(v) := x(\{v, v\}) + \sum_{w \in V} x(\{v, w\}),$$

which is the **degree** of v in the subgraph x. Note that an incidence with a loop is counted twice since loops are considered to be cycles of length 1. We simply write $deg(v)$ instead of $deg_c(v)$. More generally, we call any mapping $y : V \to \mathbf{N}_0$ with $y \leq deg$ a **degree sequence** of G.

Now let G be a connected multigraph, and a, b, two degree sequences of G so that $a(v) \leq b(v)$, $a(v) > 0$, and $b(v) < deg(v)$ for all nodes $v \in V(G)$. From the **subgraph network** $\mathcal{M} := (V, E, c, a, b)$, some interesting optimization problems result. To this end, we call any subgraph x of G with the property $deg_x \leq b$ a **matching** of $\mathcal{M}$. If G is bipartite, we speak of **assignments** instead of matchings.

A matching x is said to be a **maximum** matching iff it has maximum cardinality within the set of matchings of $\mathcal{M}$. The **extended matching problem** asks for a maximum matching of any given subgraph network $\mathcal{M}$. Here, the **cardinality** of a subgraph x of G is defined by

$$|x| := \tfrac{1}{2} \sum_{v \in V} deg_x(v)$$

and is usually considered to be the number of edges in x. But one can use the notation $|x|$ also if x is a **fractional** matching, that is, a mapping $x : E \to \mathbf{R}$ with $0 \leq x \leq c$ and $0 \leq deg_x \leq b$, where also the definition of deg_x has to be extended to nonintegral matchings.

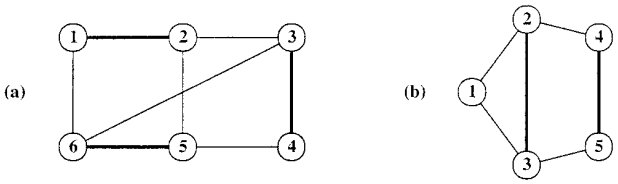

Fig. 1. Examples for the cardinality matching problem.

In what follows, let $\mathcal{M}$ denote some subgraph network and x some matching of $\mathcal{M}$. A node $v \in V$ is said to be **saturated** by x whenever $deg_x(v) \geq a(v)$ holds. The nodes which are not saturated by x are called **deficient** and collected into the set $V^-(x)$. The matching x is called **perfect** or a **factor** of $\mathcal{M}$ iff all nodes in V are saturated, that is, iff $V^-(x)$ is empty. The problem of determining a factor of $\mathcal{M}$ is called the **degree constrained subgraph problem.**

If there is no factor of $\mathcal{M}$, one could try to minimize $|V^-(x)|$, the number of deficient nodes. This has been called the **maximum saturation problem** and is known to be NP-hard [15]. Another way to measure how much the condition $a \leq deg_x$ is violated by any matching x is given by the **deficiency** of x:

$$def(x) := \sum_{v \in V^-(x)} (a(v) - deg_x(v)).$$

This **minimum-deficiency matching problem** reduces to the general matching problem if we have $a \equiv b$. We examine the case $a \equiv b$ in more detail:

A matching x is called **near-perfect** iff all nodes except v are saturated and if $deg_x(v) = a(v) - 1$. Clearly, a factor of $\mathcal{M}$ can only exist if $a(V) := \Sigma_{v \in V} a(v)$ is even, and a near-perfect matching, only if $a(V)$ is odd. In the latter case, the best we can hope is that a u-matching exists for every node $u \in V$. Then, we call $\mathcal{M}$ **factor-critical.**

For example, Figure 1 shows a 1-factor and a deficient matching within its graphs. The graph shown in Figure 1(b) is even factor-critical. We mention that factor-critical matching networks never come from bipartite graphs.

Figure 2 depicts our less trivial running example which contains some parallel edges and loops. We choose $a \equiv b \equiv 2$ and ask if the resulting subgraph network has a factor. The figure also contains a matching x consisting of all edges drawn in boldface. One can consider this matching to be found by a greedy procedure. Here, we have $V^-(x) = \{10\}$ and $def(x) = 2$. We will show how to derive from x a 2-factor, where a 2-factor is nothing but a subgraph which partitions the node set into simple cycles.

All popular matching problems are obtained by specialization of the subgraph network $\mathcal{M}$: If $a, b \equiv k \in \mathbf{N}$ holds, we speak of **k-factors** instead of factors. Thus, if we consider the cardinality matching problem, we simply set $a, b :\equiv 1$ and get 1-factors and perfect matchings as equivalent notions.

The so-called **capacitated *b*-matching problem** is the extended matching problem where loops are excluded. The ***b*-matching problem** is the case with $c \equiv b(V)$. These problems are the natural extensions of the cardinality matching problem in the context of the matching polytope.

Traditionally, factors are called (f, g)-factors, and f-factors, if we have $a \equiv b$. Since we will use f and g to denote network flows, we will use the slightly different notations introduced before.

This model is extended to weighted matching problems by adding cost labels $\gamma(e)$ and $\beta(v)$ with the arcs and, nodes respectively, of the multigraph $G(\mathcal{M})$. We will discuss nonweighted matching problems thoroughly, but we will give only a few remarks on weighted problems.

3. BALANCED FLOW NETWORKS

Searching for maximum flows of a certain class of flow networks solves bipartite matching problems very elegantly (see [17]). The class of **balanced** flow networks we introduce now yields a similar representation for the general matching problem. Related settings can be found in Tutte [27], Anstee [2], Kocay and Stone [19], and Goldberg and Karzanov [14] and in papers on the so-called **symmetric assignment problem** (see [5]).

Unfortunately, the notations of the different papers do not coincide. We will follow the terminology given in Kocay and Stone [19], the paper known to us first. Their results are the same as in Tutte [27], but the terminology is more familiar. We will point out where synonymous expressions are used among the different papers.

Let $G = (V, E)$ be a simple graph. We choose a node set V' disjoint to V of cardinality $|V'| = |V|$ and some bijection of V onto V'. The image of $v \in V$ is said to be the **complementary** (or **conjugated** [27]) node of v and is denoted by v'. The elements of V are called **outer,** and the elements of V', **inner** nodes.

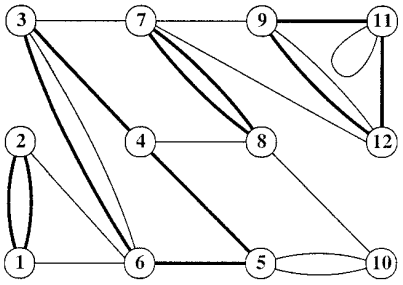

Fig. 2. A multigraph and one of its subgraphs.

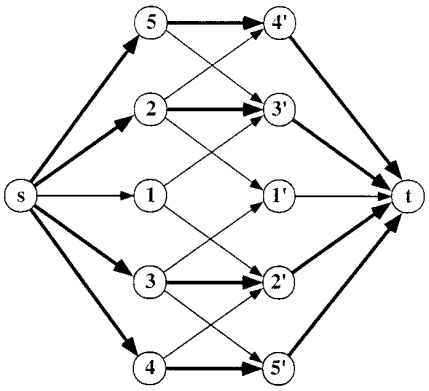

Fig. 3. The skew-symmetric digraph associated with Figure 1(b).

Moreover, we choose a source s and a sink t neither in V nor V'. By the arc sets $A_G := \{uv' : \{u, v\} \in E\}$, $A_s := \{su : u \in V\}$, and $A_t := \{v't : v \in V\}$, we define the **skew-symmetric digraph** associated with G by

$$D_G := (V \cup V' \cup \{s, t\}, A_G \cup A_s \cup A_t).$$

This graph is bipartite since $V \cup \{t\}$ and $V' \cup \{s\}$ are independent node sets. Accordingly, we consider the sink t to be an outer and the source s to be an inner node. Thus, any path in D_G visits inner nodes and outer nodes in alternation. Figure 3 shows an example, namely, the skew-symmetric digraph associated with the graph shown in Figure 1(b).

There are some symmetries inherent to D_G. To make use of these, we extend the notation of complementarity to arbitrary node sets: We define $s' := t, v' := u$, where $u' = v$ holds, and $W' := \{v' : v \in W\}$ for a given set $W \subseteq V(D_G)$. If we have $W = W'$, then W is said to be a **self-complementary set.**

Thus, the nonloop edges $\{u, v\}$ of G are represented in $N_{\mathcal{M}}$ by two arcs, namely, uv' and vu'. Again, these arcs are called **complementary,** briefly written $(uv')' = vu'$. Note that node complementarity is a graph automorphism up to the arc directions. For this reason, Goldberg and Karzanov [14] chose the term skew-symmetric.

If $B \subseteq A(D_G)$ is the sets of arcs with a given property, the **co-arcs** with respect to this property are given by the set $B' := \{a' : a \in B\}$. A loop $\{u, u\} \in E$ induces only one arc uu' in D_G, again called a **loop.** Thus, the co-loops are the loops themselves.

A network flow description of the extended matching problem is given by the digraph D_G and the capacity constraint cap which is defined by

$$cap(uv') := c(\{u, v\}) \qquad \text{for all edges } \{u, v\} \in E, u \neq v$$

$$cap(uu') := 2c(\{u, u\}) \qquad \text{for all loops } \{u, u\} \in E$$

$$cap(su), cap(u't) := b(u) \qquad \text{for all nodes } u \in V,$$

where $\mathcal{M} = (G, a, b, c)$ is a subgraph network. The network $N_{\mathcal{M}} := (D_G, cap)$ is called the **balanced flow network** associated with $\mathcal{M}$. In the special case $cap \equiv 1$, an instance of the cardinality matching problem, we write N_G instead of $N_{\mathcal{M}}$.

We are interested in those flows on $N_{\mathcal{M}}$ that are the image of some fractional matching x of $\mathcal{M}$ under the mapping

$$f(uv') := x(\{u, v\}) \qquad \text{for all non-loop edges } \{u, v\} \in E$$

$$f(uu') := 2x(\{u, u\}) \qquad \text{for all loops } \{u, u\} \in E$$

$$f(sv), f(vt) := deg_x(v) \qquad \text{for all nodes } v \in V.$$

This is the natural injection of the fractional matchings of $\mathcal{M}$ into the set of flows on $N_{\mathcal{M}}$. Hence, we say that x **corresponds** to the flow f.

For a given arc a, we denote the start node by a^-, the end node by a^+, and the reverse arc by $\bar{a}$. Let $\delta^+(s) := \{a \in A(N) : a^- = v\}$ be the set of arcs with start node s, and $\delta^-(s) := \{a \in A(N) : a^+ = v\}$. The value

$$e(v) := \sum_{a \in \delta^+(s)} f(a) - \sum_{a \in \delta^-(s)} f(a)$$

is called the **excess** of the node v with respect to f. It is obvious that we have $e(v) = 0$ for $v \neq s, t$. Accordingly, we call f an **st-flow** and

$$val(f) := e(t) = -e(s) = 2|x|$$

the **flow value.** A **maximum flow** is an *st*-flow with maximum flow value.

Maximum matchings of $\mathcal{M}$ do not correspond to maximum flows on $N_{\mathcal{M}}$ in general, and an arbitrary flow f on $N_{\mathcal{M}}$ cannot be transformed into a matching x with $2|x| = val(f)$. Indeed, all flows corresponding to some matching satisfy the following additional constraints:

(b1) $f(a)$ and $f(a')$ are equal for all arcs $a \in A(N_{\mathcal{M}})$,
(b2) $f(a)$ is integral for every arc $a \in A(N_{\mathcal{M}})$,
(b3) $f(a)$ is even for every loop $a = a' \in A(N_{\mathcal{M}})$.

Every flow on $N_{\mathcal{M}}$ and, more generally, every labeling of $A(N_{\mathcal{M}})$ with these properties is called **balanced.** The transformation described above is a bijection between the matchings of $\mathcal{M}$ and the balanced flows on $N_{\mathcal{M}}$. It is also a bijection between the fractional matchings and the

fractional balanced flows, that is, mappings which satisfy condition (b1) only.

To see that maximum matchings do not correspond to maximum *st*-flows in general, we consider the network N_G of Figure 3 and the 0–1 flow f which is supported by the arcs drawn in bold. This flow corresponds to the maximum matching shown in Figure 1(b) and, hence, is maximum balanced. However, $(s, 1, 2', 3, 1', t)$ is an augmenting path with respect to the residual network $N_{\mathcal{M}}(f)$ so that f is not a maximum flow.

On the other hand, there is an obvious way to obtain a maximum fractional balanced flow f_0 from any given maximum flow f^*. One merely has to put

$$f_0(a) := \tfrac{1}{2}(f^*(a) + f^*(a')).$$

Since f_0 and f^* have same value, f_0 is maximum again, especially maximum fractional balanced.

In ordinary network flow theory, it is standard procedure to consider the more general notion of **circulations,** which means a flow where every node v has excess $e(v) = 0$. Any *st*-flow f can be transformed into a circulation by adding the **return arc** ts and putting $f(ts) := val(f)$, eliminating the special role of the source s and the sink t. One then asks for an **admissible circulation** f which satisfies capacity bounds $0 \le lcap(a) \le f(a) \le ucap(a)$.

Not surprisingly, an analogous approach turns out also to be useful for balanced flows. There are reductions for the degree constrained subgraph problem and the minimum-deficiency matching problem to circulation problems on balanced flow networks. For this, we use D_G with the return arc ts, $ucap \equiv cap$,

$$lcap(uv') := 0 \quad \text{for all edges } \{u, v\} \in E$$

$$lcap(su), lcap(u't) := a(u) \quad \text{for all nodes } u \in V$$

and $lcap(ts) = 0$, $ucap(ts) = \infty$ and then try to find an admissible circulation. Actually, the subsequent theory does not depend on the problem transformations presented but on the following general setting:

We call any digraph (V, A) **skew-symmetric** (or **antisymmetrical** [27]) if there is a permutation π of V which has no fixpoints, only cycles of length 2, and $(\pi(v), \pi(u)) \in A$ for every arc $uv \in A$. In this setting, a **balanced flow network** is skew-symmetric where complementary nodes are specified, and the capacity labels are balanced.

For simplicity, we make the general assumption that balanced flow networks are bipartite and that neither parallel nor antiparallel arcs occur. Otherwise, we could replace the network by one of its subdivisions which is a linear reduction step again. We also assume balanced networks to be connected and the source to be an inner node. The bipartition is unique then.

Note that all of these problem transformations are linear, whereas the usual reduction mechanisms to the cardinality matching problem grow the instance size significantly (see [21]).

4. AUGMENTATION

In this section, we will prove that one can always obtain a maximum balanced *st*-flow if the choice of augmenting paths is properly restricted and augmentation is done pairwise. In what follows, let N denote a balanced flow network; s and t, a complementary pair of nodes; and f, a balanced *st*-flow on N. We begin with some basic observations:

If p is a simple vw-path in $N(f)$ and P is the set of edges on p, the edges in $P' = \{a' : a \in P\}$ define a simple $w'v'$-path p', the **complementary** path of p. Because of its meaning in the underlying graph, we sometimes call p' the **way back.**

If p is an augmenting *st*-path, also the complementary *st*-path p' is augmenting. An obvious idea is to start with the flow $f \equiv 0$ on N and to augment f step by step along pairs p, p' of *st*-paths. However, p and p' could have some arcs in common. We cannot exclude this case in general, but have to consider additional restrictions for the choice of an augmenting path p.

A simple directed path p in N is said to be **valid** (or **admissible** [27] or **regular** [14]) iff there is no complementary edge pair a, a' of residual capacity $rescap(a) = rescap(a') = 1$ on p, that is, one can augment on complementary valid paths p, p' simultaneously. We call a node $v \in V(N)$ **strictly reachable** iff there is a valid path leading from s to v.

Thus, if t is strictly reachable, there exists a pair of valid paths and we may augment the flow f. If we consider the example of $\mathcal{M}$ in Figure 2, we find

$$p := (s, 10, 8', 7, 9', 11, 11', 12, 7', 8, 4', 5, 10', t)$$

to be a valid augmenting path in $N_{\mathcal{M}}(f)$, where f is the balanced flow corresponding to the matching of Figure 2. Note that this path does not correspond to a path, but merely a walk of $G(\mathcal{M})$ since the complementary arc pair $(8', 7), (7', 8)$ is traversed. If we augment f by the paths p and p', the balanced flow corresponding to the matching of Figure 4 results.

This example is not representative for the difficulties which the balanced augmentation process presents, since the maximum balanced flows are maximum flows here. Nevertheless, reachability of t yields a necessary and also sufficient criterion for the maximality of the balanced flow f, as we shall see shortly.

If N is a flow network without antiparallel arcs, we call

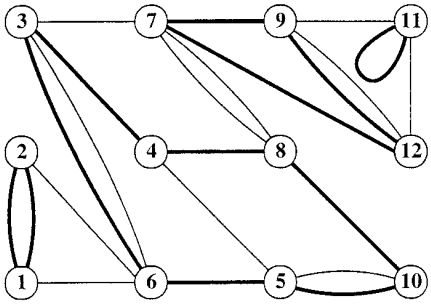

Fig. 4. A 2-factor of the graph of Figure 2.

$$Supp(f) := \{a \in A(N) : f(a) > 0\} \cup \{\bar{a} : a \in A(N), f(a) < 0\}$$

the **support** of the flow f on N. If p is a simple path p, we denote by f_p the **elementary flow** supported by p, that is, f_p is a flow with values 0, 1, -1 and $Supp(f_p)$ is the set of arcs on p.

Theorem 4.1 (Balanced Flow Decomposition). *Let f, g be different balanced circulations on the balanced flow network N. Then, there are valid cycles $p_1, p_2, \ldots, p_k$ in the residual network $N(f)$ such that*

$$g - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}). \tag{1}$$

Proof. Put $\delta := g - f$, and consider the arc sets

$$A_\delta^+ := \{a \in A(N) : \delta(a) > 0\},$$

$$A_\delta^- := \{\bar{a} : a \in A(N), \delta(a) < 0\}.$$

Note that the arcs of A_δ^+ and A_δ^- are arcs of the residual network $N(f)$. Furthermore, the set $Supp(\delta) = A_\delta^+ \cup A_\delta^-$ is self-complementary, and $\delta(a)$ is even for every loop $a \in A(N)$. If we put $\delta(a) := -\delta(\bar{a})$ for every $a \in A_\delta^-$, we see that $N(f)[Supp(\delta)]$ together with δ is a balanced network again.

Let p be a walk with respect to $N(f)[Supp(\delta)]$ such that p and p' together traverse any arc $a \in A_\delta$ at most $\delta(a)$ times. Suppose that p is a maximal walk with that property. We have to show that p is a closed walk.

From the flow conservation conditions for f and g, a ''mass balance'' equation results for every node $v \in V(N)$, namely:

$$\sum_{\substack{a \in A_\delta \\ a^+ = v}} \delta(a) = \sum_{\substack{a \in A_\delta \\ a^- = v}} \delta(a). \tag{2}$$

If v is an interior node on p, then p reaches v by an arc which is counted on the left-hand side of (2) and leaves v by an arc counted on the right-hand side.

Let u be the start node and w be the end node of p. If we would have $u \neq w$, then there would be an arc $a \in A_\delta$, $a^- = w$ which is not traversed $\delta(a)$-times by p and p' together. If a is a loop, then $\delta(a)$ is even, and a is traversed by p and p' an even number of times. In every case, p could be extended by a, contradicting the maximality of p.

Therefore, p is a closed walk. Hence, p splits into simple cycles $p_1, p_2, \ldots, p_l$ which are all valid with respect to $N(f)$. We now update g by setting

$$g :\equiv g - \sum_{i=1}^{l} (f_{p_i} + f'_{p_i}).$$

Then, g stays a balanced and admissible circulation. It is easy to see that $\Pi(\delta) := \Sigma_{a \in A_\delta} \delta(a)$ decreases by the amount $2(|p_1| + |p_2| + \cdots + |p_l|)$. Since $\Pi(\delta)$ is nonnegative and integral, there can only be a finite number of update steps on g. Therefore, we eventually achieve $f \equiv g$ which finishes the proof. ■

If we have $N = N_G$, Theorem 4.1 essentially describes the symmetric difference of two matchings of the underlying graph G. The applications of this theorem are just as widespread as in the traditional setting. In particular, one can derive optimality criteria for weighted problems also. Here, we restrict ourselves to the maximum balanced flow problem:

Theorem 4.2 (Augmenting Path Theorem for Balanced Networks). *Let f be a balanced st-flow on the balanced flow network N. Then, f is a maximum balanced flow iff there is no valid st-path in $N(f)$.*

Proof. Both directions follow by contraposition: First, let p be a valid *st*-path in $N(f)$. One obtains a balanced flow g with $val(g) = val(f) + 2$ by setting $g :\equiv f + f_p + f_{p'}$. But then f is not a maximum balanced flow.

Conversely, we assume that f is not maximum, choose some maximum balanced *st*-flow g for N, add the return arc ts to N, and consider Eq. (1). Since we have $g(ts) > f(ts)$, there is a valid cycle with respect to $N(f)$ which traverses ts. This cycle corresponds to a valid *st*-path in the former network, a contradiction. ■

If $N = N_G$ is a 0–1-balanced network, Theorem 4.2 simplifies to the well-known theorem of Berge [3], since the restriction to $cap \equiv 1$ forces valid paths not to contain complementary arc pairs. Furthermore, every valid path is **strictly simple.** This means a simple path in a balanced network which does not traverse any complementary pair of nodes (start and end node excluded).

To interpret valid paths in the underlying graph, we

delete from the residual network $N(f)$ all arcs with start node t and end node s. Then, strictly simple, directed paths are just the image of alternating paths under the transformation of (G, x) into $N_G(f)$.

In a similar way, we call a pair of valid paths p, q **strictly disjoint** iff p is node disjoint not only to q but also to q'. Thus, if p is a valid sv-path, q is a valid vw-path, and p and q are strictly disjoint; also their concatenation $p \circ q$ is valid. **Strictly edge-disjoint** paths are defined likewise.

In what follows, we need a prototype algorithm which finds a maximum balanced flow from an arbitrary balanced flow f_0, for instance, the zeroflow. This algorithm is a mere modification of the maximum-flow algorithm of Ford and Fulkerson.

As suggested by the previous discussion, we must either find a valid st-path in $N(f_i)$ in step i of the algorithm or prove that such a path does not exist. This process is called **balanced network search** (BNS). Once we have found such a path p, we can proceed from f_i to the flow

$$f_{i+1} :\equiv f_i + balcap(p)(f_p + f_{p'}), \tag{3}$$

where the **balanced capacity** of a valid path p is defined by

$$balcap(p) := \min_{a \in p} \begin{cases} \left\lfloor \dfrac{rescap(a)}{2} \right\rfloor, & \text{if } a' \in p \\ rescap(a), & \text{otherwise} \end{cases}$$

and the iteration of Eq. (3) is called an **augmentation step.** If no valid st-path is available, the algorithm halts. This is called the **balanced augmentation algorithm.**

Corollary 4.3. *The balanced augmentation algorithm halts for every balanced flow network, and returns some maximum balanced flow.* ■

At this point, we can adapt the idea of matching matroids to balanced flow networks. The following construction requires 0–1 capacities. But note that an arbitrary balanced flow network can be transformed into a 0–1 flow network by introducing explicit parallel arcs. Hence, a matroid can be associated with every balanced flow network. The resulting matroids, however, are the same. One puts

$$\mathcal{I} := \{I \subseteq \delta^+(s) : I \text{ is strictly independent}\},$$

where $I \subseteq \delta^+(s)$ is called **strictly independent** iff we have $I \subseteq Supp(f)$ for some balanced flow f on N. If N is arbitrary, there is no special structure in $\mathcal{I}$. The following theorem, however, covers more networks than just transformed networks N_G.

Theorem 4.4. *Let N be a balanced flow network with 0–1 capacities. Then, the set system $(\delta^+(s), \mathcal{I})$ forms a matroid.*

Proof. Let $\Delta \subseteq \delta^+(s)$ and $I, J \subseteq \Delta$ maximal independent with $|I| \leq |J|$. It is sufficient to show that $|I| = |J|$.

Let balanced st-flows f, g be chosen so that $I \subseteq Supp(f)$, $J \subseteq Supp(g)$. Since I and J are maximal, we have $I = Supp(f) \cap \Delta$ and $I = Supp(g) \cap \Delta$. Since a flow on the arcs in $\delta^+(s)$ cannot decrease by an augmentation step, we can assume that f and g are maximum balanced flows. Decompose

$$g - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are pairwise strictly edge disjoint and valid in $N(f)$. If we would have $|I| < |J|$, then we could find an index i so that the cycle p_i traverses an arc $a \in J \backslash I$ but not the reverse of any arc in $I \backslash J$. We would obtain a flow $h :\equiv f + f_{p_i} + f_{p_i'}$ with $Supp(h) \cap \Delta = I \cup \{a\}$, contradicting the maximality of I. ■

By this result, a maximal independent arc set with minimum costs can be found by a greedy strategy. But, then, a maximum balanced st-flow with minimum vertex costs can be found as follows:

Determine some maximum balanced st-flow. Order the nodes in $V(\mathcal{M})$ by the β-labels into a queue Q. Successively delete a node v with minimum $\beta(v)$ on Q. Run a BNS, and check if there is a strictly v-reachable node $w \in V(\mathcal{M})$ which is still on Q. In that case, expand a valid vw-path q in $N(f)$, put $p := q \circ (w, s, v)$, augment $f :\equiv f + balcap(p)(f_p + f'_p)$, and repeat the BNS.

5. LOWER-CAPACITY BOUNDS

Next, we describe how to find a balanced circulation on a balanced flow network N with given capacity bounds $lcap$ and $ucap$: We reduce the problem to the maximum balanced flow problem. The same technique was used by Ford and Fulkerson [9] for finding ordinary circulations.

We introduce an artificial source node s^*, a sink node t^*, and replace the capacities by a labeling cap which is defined by

$$cap(a) := ucap(a) - lcap(a)$$

for every arc $a \in A(N)$ and

$$cap(s^*, v) := \sum_{\substack{a \in A(N) \\ a^+ = v}} lcap(a),$$

$$cap(v, t^*) := \sum_{\substack{a \in A(N) \\ a^- = v}} lcap(a)$$

for every node $v \in V(N)$. The resulting flow network is denoted by N^*, where s^* and t^* are considered to be complementary nodes. Since we have

$$cap(s^*, v) = \sum_{\substack{a \in A(N) \\ a^+ = v}} lcap(a) = \sum_{\substack{a \in A(N) \\ a^- = v'}} lcap(a) = cap(v', t^*), \quad (4)$$

N^* is balanced. In the case of a subgraph network $\mathcal{M}$, the network $N^*_{\mathcal{M}}$ is bipartite, since s^* is adjacent with t and the nodes in $V(\mathcal{M})$ only. Moreover, if we have $a(\mathcal{M}) \equiv b(\mathcal{M})$, then $N_{\mathcal{M}}$ and $N^*_{\mathcal{M}}$ are essentially the same (t is adjacent with s and s^* only).

Let f be a nonnegative balanced circulation for N where we require that the upper-capacity constraints are satisfied but allow that the lower bounds are violated. We define a flow f^* for N^* by

$$f^*(a) := max\{0, f(a) - lcap(a)\}$$

for every arc $a \in A(N)$ and

$$f^*(s^*, v) := \sum_{\substack{a \in A(N) \\ a^+ = v}} (lcap(a) - max\{0, lcap(a) - f(a)\})$$

$$f^*(v, t^*) := \sum_{\substack{a \in A(N) \\ a^- = v}} (lcap(a) - max\{0, lcap(a) - f(a)\})$$

for every node $v \in V(N)$. We say that f^* **corresponds to** f. It is obvious that f^* is admissible and balanced. For a given $v \in V(N)$, we have

$$\begin{aligned} \sum_{\substack{a \in A(N) \\ a^+ = v}} f(a) &= \sum_{\substack{a \in A(N) \\ a^+ = v \\ f(a) < lcap(a)}} f(a) + \sum_{\substack{a \in A(N) \\ a^+ = v \\ f(a) \geq lcap(a)}} f(a) \\ &= \sum_{\substack{a \in A(N) \\ a^+ = v}} (lcap(a) \\ &\quad - max\{0, lcap(a) - f(a)\} \\ &\quad + max\{0, f(a) - lcap(a)\}) \\ &= f^*(s^*v) + \sum_{\substack{a \in A(N) \\ a^+ = v}} f^*(a) \end{aligned} \quad (5)$$

and the analogus equation

$$\sum_{\substack{a \in A(N) \\ a^- = v}} f(a) = f^*(v, t^*) + \sum_{\substack{a \in A(N) \\ a^- = v}} f^*(a), \quad (6)$$

which implies that $e(v) = 0$ for the nodes $v \in V(N)$. Hence, f^* is an s^*t^*-flow on N^*. One can now characterize the admissible balanced circulations for N by their corresponding flows on N^*:

Theorem 5.1. *Let N be a balanced flow network with capacity constraints lcap and ucap, and N^* be defined as above. Then, there is an admissible balanced circulation on N iff a balanced s^*t^*-flow f^* on N^* with $val(f^*) = \Sigma_{a \in A(N)} lcap(a)$ exists.*

Proof. ($\rightarrow$) Let f be an admissible balanced circulation on N, and f^*, the corresponding flow on N^*. Since $f(a) \geq lcap(a)$ is assumed for every arc $a \in A(N)$, we obtain

$$f^*(s^*v) = \sum_{\substack{a \in A(N) \\ a^+ = v}} lcap(a) = cap(s^*v) \quad (7)$$

for every node $v \in V(N)$ or, equivalently,

$$val(f^*) = \sum_{v \in V(N)} f^*(s^*v) = \sum_{a \in A(N)} lcap(a). \quad (8)$$

($\leftarrow$) Let f^* be a balanced s^*t^*-flow on N^* which satisfies the Eqs. (7) and (8). Put $f(a) := f^*(a) + lcap(a)$ for every arc $a \in A(N)$. It is obvious that f is admissible and balanced for N. The flow conservation conditions follow by

$$\begin{aligned} \sum_{\substack{a \in A(N) \\ a^+ = v}} f(a) &= \sum_{\substack{a \in A(N) \\ a^+ = v}} (f^*(a) + lcap(a)) \\ &= f^*(s^*v) + \sum_{\substack{a \in A(N) \\ a^+ = v}} f^*(a). \end{aligned} \quad (9)$$

■

Using this reduction mechanism, we can solve the degree-constrained subgraph problem as soon as we have efficient methods for the maximum balanced flow problem. A good characterization of minimum-deficiency matchings requires some defect form of Theorem 5.1:

Let $\mathcal{M}$ be a subgraph network; x, a matching for $\mathcal{M}$; f, the corresponding circulation on $N_{\mathcal{M}}$; and f^*, the s^*t^*-flow on $N^*_{\mathcal{M}}$ corresponding to f. Then, we say that f^* **corresponds to** x. Since $f(s, v)$ denotes the degree of v in x, we observe that

$$def(x) = cap(s^*, t) - f^*(s^*, t), \quad (10)$$

$$def(x) = \sum_{v \in V(\mathcal{M})} (cap(s^*, v) - f^*(s^*, v)), \quad (11)$$

and, hence,

$$val(f^*) + 2def(x) = 2 \sum_{v \in V(\mathcal{M})} a(v). \quad (12)$$

Note that a flow on $N^*_{\mathcal{M}}$ corresponds to a flow on $N_{\mathcal{M}}$ only in special circumstances. But we can **restrict** an arbitrary balanced flow f^* for $N^*_{\mathcal{M}}$ to a matching x for $\mathcal{M}$ by putting $x(\{u, v\}) := f(u, v')$ for every adjacent pair $u, v \in V(\mathcal{M})$. If a flow f^* corresponds to a matching x, then f^* restricts to x. Denote by $N^{\#}_{\mathcal{M}}$ the network $N^*_{\mathcal{M}}$ where

$$cap(s^*, t), cap(s, t^*) := a(V(\mathcal{M})) - def(\mathcal{M})$$

is set. This modification is not made for algorithmic purposes. But we can identify maximum balanced flows and minimum-deficiency matchings, then,

Lemma 5.2. *A maximum balanced st-flow on $N^{\#}_{\mathcal{M}}$ corresponds to a minimum-deficiency matching.*

Proof. Let f be a maximum balanced st-flow on $N^{\#}_{\mathcal{M}}$, x the restricted matching, and g, the flow corresponding to x. Assume that $f \not\equiv g$ and decompose

$$f - g \equiv \sum_{i=1}^{k} (f_{p_i} + f_{p_i}).$$

Assume that p_i traverses s^*. Note that $val(g) \leq val(f)$ and $f(s^*, t) \leq g(s^*, t)$ so that p_i would traverse an arc $s^*u, u \in V(\mathcal{M})$. But $g(s^*, u) < a(u)$ implies that $g(s, u) = 0$. Since we have $g(u, v') = f(u, v')$ for any $v \in V(\mathcal{M})$, the node u cannot not be left by p_i, a contradiction. If follows that no node in $V(\mathcal{M})$ is traversed by $p_1, \ldots, p_k$. Hence, $f \equiv g$ holds. ■

Lemma 5.3. *Let f be a maximum balanced s^*t^*-flow on $N^{\#}_{\mathcal{M}}$. Then, neither s nor t are strictly s^*-reachable in $N^{\#}_{\mathcal{M}}$.*

Proof. Suppose, otherwise, that p is a valid s^*s-path in $N^{\#}_{\mathcal{M}}(f)$ which does not traverse t. Note that $f(s^*, t) = a(V) - def(\mathcal{M})$ by Lemma 5.2. But then we have $def(\mathcal{M}) > 0$ since $f(s^*, v) < a(v)$ for at least one node $v \in V(\mathcal{M})$.

Hence, p could be extended by the arc st^* to an augmenting path in $N^*_{\mathcal{M}}(f)$. The augmented flow would correspond to a matching again, contradicting the definition of $N^{\#}_{\mathcal{M}}$.

A valid s^*t-path which does not traverse s could be extended by ts and st^* to an augmenting path in $N^*_{\mathcal{M}}(f)$. An analogous contradiction follows. ■

For algorithmic purposes, statements concerning the network $N^{\#}_{\mathcal{M}}$ are useless. But we can give another optimality criterion:

Theorem 5.4. *Let x be a matching of $\mathcal{M}$, and f, the corresponding flow on $N^*_{\mathcal{M}}$. Then, x has minimum deficiency iff every valid s^*t^*-path in $N^*_{\mathcal{M}}(f)$ starts with the arc s^*t and ends with the arc st^*.*

Proof. ($\rightarrow$) Let x be minimum deficient so that f is a maximum balanced s^*t^*-flow on $N^{\#}_{\mathcal{M}}$. If there is a valid augmenting path p in $N^*_{\mathcal{M}}(f)$, at least s^*t or st^* is on p. If p would traverse st^* but not s^*t, then t would be strictly s^*-reachable in $N^{\#}_{\mathcal{M}}(f)$, contradicting Lemma 5.3. If p would traverse s^*t but not st^*, we would exchance p by p' to obtain a contradiction.
($\leftarrow$) Let y be a minimum-deficient matching; g, the corresponding flow on $N^*_{\mathcal{M}}$, and $def(y) < def(x)$. Decompose

$$g - f \equiv \sum_{i=1}^{k} (f_{p_i} + f_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are valid in $N^{\#}_{\mathcal{M}}(f)$. By the choice of y, we have $f(s^*, v) < g(s^*, v)$ for at least one node $v \in V(\mathcal{M})$. Hence, there is an augmenting path among $p_1, p_2, \ldots, p_k$ which traverses the arc s^*v. ■

To find a valid augmenting path p as in the this theorem, one would simply ignore the arcs s^*t and st^* during the balanced network search. If t^* still is strictly s^*-reachable (say by a valid path p), we would augment on the paths $p, p', p^* := (s^*, t, s, t^*)$, and p'^*. Otherwise, if t is reached (say by a path q), we would augment on $q \circ (t, s, t^*)$ and $(s^*, t, s) \circ q'$. In either case, we obtain a flow which corresponds to a matching again.

6. DISTANCE LABELS

Of course, we are interested in efficient techniques to decide whether valid st-paths exist, and, if so, to obtain such a path. Kocay and Stone [20] devised a BNS procedure which determines a valid st-path in $O(n^2)$-time. We will present methods which even run in (almost) $O(m)$ time in [10] and [11].

The balanced augmentation algorithm together with such a procedure will behave polynomially, if the maximum flow value on N polynomially depends on the encoded length of the network N. This is satisfied for $N_{\mathcal{M}}$, where $\mathcal{M}$ is an instance for the f-factor problem on simple graphs.

In this section, we will show that the number of aug-

mentation steps can be made independent from the arc capacities by choosing minimum-length valid *st*-paths at every single step. The distances of nodes on valid paths are called **balanced distances.** With respect to the source s, we define distance labels by

$$d(v) := \min\{\infty, \{|p| : p \text{ is a valid } sv\text{-path in } N(f)\}\}.$$

Any valid sv-path q with $|q| = d(v)$ is called **minimum valid** or simply a $d(v)$-path. In what follows, we consider the method *BNS* to be implemented in such a way that all paths $p_1, p_2, \ldots, p_j$ determined for augmentation are minimum valid. This section contains an analysis of the number of augmentation steps of the balanced augmentation algorithm under this assumption. The key to our results is the observation that antiparallel arcs are relatively rare within the paths $p_1, p_2, \ldots, p_j$:

Lemma 6.1. *Let p be a $d(t)$-path in $N(f)$, and uv', some arc on p. Then, p traverses neither $v'u$ nor $u'v$.*

Proof. Since valid paths are simple, p cannot contain any pair $uv', v'u$ of antiparallel arcs. Now assume that p traverses some arc pair $uv', u'v$. Then, p can be written as $p = p_1 \circ uv' \circ p_2 \circ u'v \circ p_3$ or $p = p_1 \circ u'v \circ p_2 \circ uv' \circ p_3$ for certain paths p_1, p_2, p_3.

We shorten p by setting $p := p_1 \circ p_2' \circ p_3$. By this operation, p looses exactly two arcs, has not to be a simple path any longer, but can be shortened again to a simple *st*-path q. This path q not only is simple, but also valid in $N(f)$, since q only contains arcs of p and p', and q can contain a pair a, a' only if p has traversed both arcs a, a' before. Since q is strictly shorter than p, we obtain a contradiction to the choice of p. ■

Theorem 6.2. *Let p be a $d(t)$-path in $N(f)$, put $g := f + f_p + f_{p'}$, and let q be a $d(t)$-path in $N(g)$. Then, $|p| \le |q|$ holds.*

Proof. Let $h := g + f_q + f_{q'}$ and assume that $|q| \le |p|$. As explained at the end of Section 3, we can consider f and g as circulations. By Theorem 4.1, we have

$$\delta := h - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are valid cycles in $N(f)$. We have $\delta(ts) = 4$; hence, there are exactly two augmenting paths among $p_1, p_2, \ldots, p_k$. Without loss of generality, let p_1, p_2 be these paths and $|p_1| \le |p_2|$. Using $\Pi(\delta)$ as given in the proof of Theorem 4.1, we observe

$$\begin{aligned} 2(|p| + |q|) &\ge \Pi(\delta) \\ &= 2(|p_1| + |p_2| + \cdots + |p_k|) \qquad (13) \\ &\ge 2(|p_1| + |p_2|). \end{aligned}$$

As p is minimum valid, we have $|p| \le |p_1|$. Then, (13) shows $|q| \ge |p_2|$ and, hence, $|q| \ge |p|$, so $|p| = |q|$. ■

If we partition the augmentation step along any pair of paths p_j and p_j' into $balcap(p_j)$ augmentation along p_j and p_j', Theorem 6.2 shows that the length of the paths p_j derived by the balanced augmentation algorithm cannot decrease. Let $2\nu := val(f)$ be the value of the maximum balanced flow f computed by the algorithm. We collect all augmenting paths of equal length into the set

$$phase_i := \{p_j : 1 \le j \le \nu, |p_j| = 2i + 1\}$$

and call any period during which the balanced augmentation algorithm derives $d(t)$-paths of equal length a **phase.** The existence of phases, as implied by the last theorem, is crucial for efficient implementations of the method.

Corollary 6.3. *Let p, q be paths as in Theorem 6.2, assume that $|q| = |p|$, and let uv' be some arc on q. Then, p traverses neither $v'u$ nor $u'v$.*

Proof. Suppose otherwise, so that p contains $v'u$ or $u'v$. Then, we would have $\delta(uv') = 0$ in the proof of Theorem 6.2, and, hence, $2(|p| + |q|) > \Pi(\delta)$, that is, the inequality (13) would be strict. ■

The preceding corollary is important for two reasons: First, it shows that all augmentation steps of a single phase **commute,** that is, those steps can be swapped without turning intermediate flows inadmissible. Moreover, the number of augmentation steps of any phase i can now be bounded from above: Each path $p_j \in phase_i$ contains at least one arc a that prevents us from increasing the flow value by more than $balcap(p_j)$ units; such an arc is called a **blocking** arc on p_j.

If a is blocking and p_j does not traverse a', then $rescap(a) = 0$ holds after iteration j. Otherwise, if a' is also on p_j, after this augmentation step, $rescap(a) \le 1$ holds. Because of Corollary 6.3, $rescap(a)$ cannot increase in phase i again. In particular, any arc a can occur as a blocking arc at most twice within a given phase.

Since the complementary arc a' occurs as a blocking arc whenever a does, and since the reverse arcs $\bar{a}$ and $\bar{a}'$ do not occur in this phase at all (by Corollary 6.3), we conclude $|phase_i| \le m + n$. Obviously, the length of any valid *st*-path is odd and restricted by the total number of nodes of N, so the balanced augmentation algorithm runs in at most n phases. We get the following important result:

Theorem 6.4. *If all derived augmenting paths are minimum valid, the balanced augmentation algorithm consists*

of $O(n)$ phases, and each phase consists of $O(m + n)$ augmentation steps. ■

There still remains the problem how to find a $d(t)$-path efficiently. As we will see in [11], this can be done with a calculus which generalizes the state-of-the-art algorithm of Micali and Vazirani [23] for the cardinality matching problem. In fact, one of the major advantages of phase ordering is the following: The augmenting paths of each phase have not to be computed independently, so that the computational effort may be decreased considerably. Especially in the case of the cardinality matching problem, considerably stronger complexity bounds are available. The following result is due to Hopcroft and Karp [16].

Lemma 6.5. *Let N_G be the 0–1-balanced flow network associated with graph G. If all derived augmenting paths are minimum valid, the augmenting paths of a particular phase are pairwise disjoint.*

Proof. Let $p_1, p_2, \ldots, p_r$ be the augmenting paths computed during some phase of the algorithm, and f, the flow just before the first augmentation step of this phase. Since there are no antiparallel arc pairs among $p_1, p_1', p_2, p_2', \ldots, p_r, p_r'$, the augmenting paths are at least edge-disjoint.

Let u be an outer node other than t. In the case of $f(s, u) = 1$, there is a unique node v with $f(u, v') = 1$. Otherwise, we have $f(s, u) = 0$. In either case, there is only one arc a with $rescap(a) > 0$ and end node u. Hence, the node u can be traversed by at most one of the paths $p_1, p_1', p_2, p_2', \ldots, p_r, p_r'$. An analogus statement holds for the inner nodes other than s. ■

Theorem 6.6. *Let N_G be the 0–1-balanced flow network associated with graph G. If all derived augmenting paths are minimum valid, the balanced augmentation algorithm consists of $O(\sqrt{n})$ phases.*

Proof. Let 2ν be the value of a maximum balanced flow on N_G, $r := \lceil \sqrt{n} \rceil$. If we have $2(\nu - r) \leq 0$, the assertion is obvious. Hence, let $2(\nu - r) > 0$, let f be the balanced flow of value $2(\nu - r)$ which is constructed by the algorithm, and let g a maximum balanced flow on N_G. By Theorem 4.1, we can write

$$g - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are valid in $N(f)$, $r \leq k$ and $p_1, p_2, \ldots, p_r$ are augmenting.

Note that $p_1, p_1', p_2, p_2', \ldots, p_r, p_r'$ are pairwise disjoint and can traverse at most $val(f) = 2(\nu - r)$ backward arcs altogether. Thus, at least one of the paths traverses at most $(1/r)(\nu - r)$ backward arcs. In particular, we have

$$\begin{aligned} d(t) &\leq \frac{2}{r}(\nu - r) + 3 \\ &= \frac{2\nu}{r} + 1 \leq 2\sqrt{n} + 1 \leq 2\lfloor \sqrt{n} \rfloor + 3, \end{aligned} \tag{14}$$

since, trivially, $\nu \leq n$. Hence, at most $\lfloor \sqrt{n} \rfloor + 1$ phases occur before f is constructed, and at most $r = \lceil \sqrt{n} \rceil \leq \lfloor \sqrt{n} \rfloor + 1$ phases occur after the construction of f. This is the bound $O(\sqrt{n})$. ■

7. WEIGHTED PROBLEMS

To conclude the discussion of optimality criteria for balanced network flows, we must say something about weighed problems. Although the optimality criteria are simple, the respective algorithms seem rather involved.

Assume that N is a balanced flow network with lower- and upper-capacity bounds and, in addition, that a cost label $\gamma(a) = \gamma(a')$ is assigned with every arc $a \in A(N)$. Let us call a balanced circulation f **optimum** iff

$$\gamma(f) := \sum_{a \in A(N)} \gamma(a) f(a)$$

is minimum among all balanced circulations on N. If p is a valid path in $N(f)$, we also write $\gamma(p) := \gamma(f_p)$. There is an obvious linear reduction of all given matching problems to these class of flow networks.

Theorem 7.1 (Primal Approach). *Let f be a balanced circulation on the balanced flow network N. Then, f is optimum iff there is no valid cycle p in $N(f)$ with length $\gamma(p) < 0$.*

Proof. Let g be optimum and decompose

$$g - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are valid in $N(f)$. Since $\gamma(g) - \gamma(f) < 0$, and since γ is linear, we can find an index i so that $\gamma(p_i) < 0$. ■

We have shown how to find some admissible balanced circulation. But remember how difficult primal algorithms for the ordinary max-flow problem are to implement efficiently. It is an open question if the ideas used there apply also to balanced flows. To our knowledge, a primal matching algorithm has been deviced only once, for

the perfect matching problem by Cunningham and Marsh [4].

We may be interested in a balanced *st*-flow f of given value ν with $\gamma(f)$ minimum. Such a flow is called (ν)-**optimum.** Note that we do not need to distinguish between (ν)-optimum and *extreme* flows as in the graph theoretical context. If $f \equiv 0$ is (0)-optimum, a series of (ν)-optimum balanced flows can be derived using the following idea:

Theorem 7.2 (SAP Approach). *Let f be a (ν)-optimum balanced st-flow on the balanced flow network N, and p, a shortest valid augmenting path in $N(f)$. Then, $g :\equiv f + f_p + f_{p'}$ is $(\nu + 2)$-optimum balanced.*

Proof. Suppose that g is not $(\nu + 2)$-optimum balanced. If we put $lcap(t, s) := \nu$ and $ucap(t, s) := \nu$ in N, we can find a valid cycle q in $N(g)$ with length $\gamma(q) < 0$. Let $h :\equiv g + f_q + f_{q'}$ and decompose

$$h - f \equiv \sum_{i=1}^{k} (f_{p_i} + f'_{p_i}),$$

where $p_1, p_2, \ldots, p_k$ are valid in $N(f)$. Without loss of generality, let p_1 be the augmenting cycle among $p_1, p_2, \ldots, p_k$. Since $p_2, \ldots, p_k$ do not traverse the return arc in either direction, we obtain $\gamma(p_2) \geq 0, \ldots, \gamma(p_k) \geq 0$. Observe that

$$\gamma(p) + \gamma(q) = \gamma(p_1) + \gamma(p_2) + \cdots + \gamma(p_k)$$

and $\gamma(p_2) + \cdots + \gamma(p_k) \geq 0 > \gamma(q)$. Hence, we have that $\gamma(p_1) < \gamma(p)$, a contradiction. ■

An algorithm is known which determines a shortest valid augmenting path in polynomial time. It is a primal-dual procedure and was described by Goldberg and Karzanov [14]. By the reduction principle of Section 5, the SAP approach applies to the problem of finding an optimum balanced circulation and, in turn, to all discussed matching problems.

Note that a maximum balanced flow of minimum costs can be found by an SAP algorithm efficiently only if the maximum flow value depends polynomially on the network size, that is, the f-factor problem for simple graphs is solved efficiently, but not the general problem.

The traditional approach to weighted matching problems is the primal-dual approach which depends on the famous algorithm given by Edmonds [6]. Pulleyblank [25] and Marsh [22] applied the idea to capacitated matching problems, by mere reduction to the perfect matching problem. In the latter paper, scaling techniques were used, so that the resulting algorithm is weakly polynomial.

It should turn out that the SAP algorithm and the balanced augmentation algorithm are special implementations of the PD algorithm. A description of the PD algorithm would require an LP formulation of balanced circulations, which is nontrivial and therefore omitted. An PD-optimality criterion would result from the complementary slackness conditions.

8. TENACITY LABELS

This section deals with the distance between the nodes of any balanced network N with respect to valid paths, now that we have seen their relevance for matching algorithms. To this end, we will classify the arcs of N. We will use another labeling $t : V(N) \cup A(N) \to \mathbf{N}^{\infty}$ associated with the distance function d, the so-called **tenacity** which is defined by

$$t(w) := d(w) + d(w')$$

$$t(u, v') := d(u) + d(v) + 1$$

for all nodes $u, v, w \in V(N)$. The tenacity will be used to decide within the computation of d or a $d(t)$-path which arc should be investigated next. Since u and v have the same, but w and w' have different parity for each node v in $V(N)$, tenacity labels are always odd or infinite. We also note that tenacity labels are invariant under complementarity, that is, $t(w) = t(w')$ and $t(u, v') = t(v, u')$ hold.

Lemma 8.1. *Let N be a balanced network and assume that t is strictly reachable. Then, $t(w), t(a) \leq d(t)$ holds for every node v and every arc a traversed by some $d(t)$-path q.*

Proof. If we assume that $uv' = a$ and use the fact that the complement of a valid path is a valid path again, the assertion follows by observing that

$$\begin{aligned} d(t) = |p| &= |p[w]| + |p[w, t]'| \\ &\geq d(w) + d(w') = t(w) \end{aligned} \tag{15}$$

and

$$\begin{aligned} d(t) = |p| &= |p[u]| + |(u, v')| \\ &+ |p[v', t]'| \geq d(u) + d(v) + 1 = t(a). \end{aligned} \tag{16}$$

Here, $p[x]$ denotes the part of a simple path p until x is reached, and $p[y, z]$ denotes the subpath of p connecting y and z. ■

The distances of the source s to the nodes in $V(N)$ with respect to arbitrary directed paths can be obtained

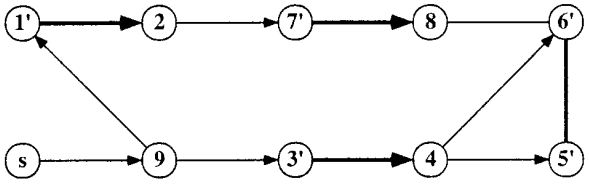

Fig. 5. Choosing an appropriate predecessor.

by a breadth-first search — hence, in $O(m)$ steps. This is usually used by augmentation algorithms for the maximum-flow problem. Here, the BFS grows an arborescence that contains a minimum-length directed path for each node reached before.

The following simple example shows that minimum valid paths cannot be determined in this manner. It is also a counterexample for the incorrect cardinality matching algorithm of Pape and Conradt [24] (see [17]).

Figure 5 consists of a simple graph G and a matching x of G (consisting of the lines in bold face). There is some additional information shown which previews the correct computation of the balanced distances with respect to $N_G(f)$, but which shall not be described yet. It is easy to see that $d(3') = 2$ and $d(3) = 9$ hold and that the unique $d(3)$-path is $(s, 9, 1', 2, 7', 8, 6', 5, 4', 3)$. Hence, node 3 has tenacity $t(3) = 11$. Suppose that we have computed $d(v)$ and some predecessor of v on a $d(v)$-path for every strictly reachable node v (In this network, every node but $4'$ has a unique predecessor.)

The predecessor function determines an arborescence. We might expect this tree to contain a system of minimum valid paths for all strictly reachable nodes. Unfortunately, the only $d(3)$-path is not entirely contained in this tree since the predecessor of $6'$ is not the node 8 but 4, and not even $d(3) = d(4') + 1$ holds!

Nevertheless, the distance labels d can be computed in approximately $O(m)$ steps. Micali and Vazirani [23] solved this task for the cardinality matching problem by an appropriate classification of the edges of the graph G which is generalized to balanced networks in what follows.

Let v and v' be two complementary nodes with $d(v) < d(v')$. Then, we call v a **minlevel** node and v' a **maxlevel** node. For example, the source s is a minlevel node, and its complementary node t, a maxlevel node. The distinction between minlevel and maxlevel nodes is reasonable, since minlevel node distance labels can be computed by a rudimentary BFS, as the following lemma shows:

Lemma 8.2. *Let $v \neq s$ be a minlevel node; p, a $d(v)$-path; and u', the predecessor of v on p. Then, $d(v) = d(u') + 1$ holds, and any $d(u')$-path can be extended to a $d(v)$-path by appending $u'v$.*

Proof. Choose some $d(u')$-path q. Because of $d(u') < d(v) < d(v')$, neither v nor v' can occur on q. In particular, $r := q \circ u'v$ is a valid sv-path. Since $p[u'$ is a valid su'-path, $|q| + 1 \leq |p|$ holds. If even $|q| + 1 < |p|$ would hold, we would have $|r| < |p|$ in contradiction to the minimality of p. Thus, r is a $d(v)$-path and $d(v) = d(u') + 1$ holds. ■

If u' and v are nodes as in Lemma 8.2, we call u' a **predecessor** of v and the arc $u'v$ a **prop.** Every arc which is neither a prop nor a co-prop is said to be a **bridge.**

Corollary 8.3. *Let $v \neq s$ be a minlevel node of N. A path p is a $d(v)$-path iff $p = q \circ u'v$ holds for a prop $u'v$ and a $d(u')$-path q.* ■

The simple proof of the following observations may be left to the reader:

Observation 8.4. *In any balanced network N, the following hold:*

(a) A co-prop never is a prop.
(b) A co-bridge a' is a bridge itself, and $t(a) = t(a')$.
(c) Every loop is a bridge. ■

In Section 12, we will prove that only bridges a with $t(a) \leq t(v)$ can occur on $d(v)$-paths. In particular, if v is a maxlevel node, every $d(v)$-path is proven to traverse exactly one bridge a of tenacity $t(a) = t(v)$. Such a bridge a is said to be a **petal** of node v. The role of petals for maxlevel nodes is analogous to that of props for minlevel nodes in the reconstruction of (minimum) valid paths.

There is a class of bridges that cannot occur on minimum valid paths at all: Suppose that not only the arc a but also its reverse arc $\bar{a}$ are bridges and that $t(a) \leq t(\bar{a})$ holds. We will show later that then only a can be traversed by a minimum valid path. In this context, the next lemma should be noted:

Lemma 8.5. *Let $a = uv'$ and $\bar{a} = v'u$ be bridges and assume that $t(a)$ is finite. Then, $t(a) \neq t(\bar{a})$ holds.*

Proof. For $u = v$, trivially, $t(a) = 2d(u) + 1 \neq 2d(u') + 1 = t(\bar{a})$ holds. We therefore assume that $u \neq v$ and $t(a) = t(\bar{a}) < \infty$. Then, either u is a minlevel and v is a maxlevel node or vice versa. Since a and a' are complementary, we can assume u to be the minlevel node.

For reasons of parity, $d(u)$ and $d(v')$ cannot be equal. If $d(u) < d(v')$ holds, no $d(u)$-path p can traverse v or v'. Thus, $p \circ a$ is a $d(v')$-path, and a, a prop. In the same manner from $d(v') < d(u)$, we conclude $\bar{a}$ to be a prop. ■

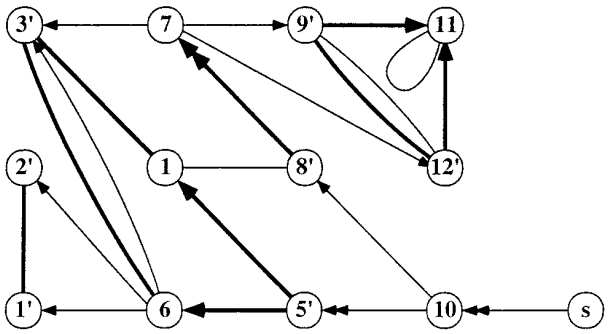

Fig. 6. The residual network associated with the example in Figure 2.

On the other hand, the reverse arc of a bridge may be a prop, and some node cannot be reached without this bridge sometimes. To see this, we consider the multigraph $G = (V, E, c)$ of Figure 2 and the subgraph network $\mathcal{M} := (V, E, c, 2, 2)$ again which models the search for a 2-factor of G. We also use the matching x shown there and the corresponding flow f on $N_{\mathcal{M}}$.

Since the networks $N_{\mathcal{M}}$ and $N_{\mathcal{M}}(f)$ associated with $\mathcal{M}$ are quite large in general and in particular for our running example, we chose a more economic representation of the residual networks $N_{\mathcal{M}}(f)$ that depicts the underlying multigraph $G(\mathcal{M})$ and is to be read in the following manner: Only the minlevel nodes are shown, so that some nodes of G with infinite tenacity are neglected. Forward edges with respect to $N_{\mathcal{M}}$ are drawn in thin, backward edges in bold-faced lines. If v is a predecessor of node w', the representation contains an arc directed from v and v' to w'. The prop gets a double arrowhead iff its residual capacity is greater than one. Co-props and bridges a with $t(a) = \infty$ are not represented, while the remaining bridges are represented in complementary pairs by an undirected edge. We warn the reader that this representation already uses information which is not yet available to us!

Figure 6 depicts the residual network $N_{\mathcal{M}}(f)$ belonging to our running example in which $(s, 10, 8', 7, 3', 6, 1', 2, 6', 5, 10', t)$ occurs as a $d(t)$-path. This path traverses the bridge $(3', 6)$, but at the same time 6 is a predecessor of $3'$. Thus, a system of $d(v)$-paths to each node $v \in V(N_{\mathcal{M}})$ can contain antiparallel arcs. In our example, we could choose another $d(t)$-path, but, in general, we cannot avoid such situations.

9. BLOSSOMS AND NUCLEI

In this section, we present two closely related subgraph structures which have been used in Tutte [26] first, called **blossoms** and **nuclei** in what follows. Both notions are almost equivalent, and subsequent authors in matching theory have introduced a series of notions which, in turn, are almost equivalent to those of Tutte. No one has explained the relationships precisely up to now. Before we can do this, we need some formalism.

We call an arc $(u', v) \in A(N)$ **strictly accessible** iff $u'v$ is traversed by some $u'v$-**accessing** path p. By that, we denote a path starting at s such that $p[u']$ is simple, and no arc pair a, a' with $rescap(a) = 1$ is traversed by p. If p is a valid sv-path, and u', the predecessor of v on p, then p is $u'v$-accessing. As a slight generalization, on a $u'v$-accessing path, v can be traversed by $p[u']$ once before. For example, props are always strictly accessible. It turns out that a bridge is strictly accessible iff it has finite tenacity.

A strictly accessible arc a is said to be **bicursal** iff a' is also accessible and **unicursal** otherwise. Iff neither a nor a' is accessible, then both arcs are called **acursal.** Let $Bic(N)$ denote the set of all bicursal arcs of N, and $N[Bic(N)]$, the subnetwork of N which has arc set $Bic(N)$ and the end nodes of bicursal arcs as nodes. The connected components of $N[Bic(N)]$ are called the **blossoms** or the **bicursal components** of N as in the original paper of Tutte. For every node v incident with any bicursal arc, we denote the blossom containing v by $B(v)$.

For example, all arcs on a valid st-path p are bicursal since p' is also valid. In particular, $B(s)$ and $B(t)$ coincide. On the other hand, both end nodes of a bicursal arc uv' have finite tenacity, since all of u, v', v, and u' are strictly reachable. This suggests another subgraph structure depending on the tenacity labels:

For every balanced network N, an edge cut $Q(N) := [S, T]$ exists which divides its node set into reachable and unreachable nodes with respect to s. This cut is defined by $S := \mathcal{B} \uplus \mathcal{C}$ and $T := \mathcal{B}' \uplus \mathcal{D}$, where we put

$$\mathcal{B} := \{v \in V(N) : d(v) < \infty, d(v') = \infty\}$$

$$\mathcal{C} := \{v \in V(N) : d(v), d(v') < \infty\}$$

$$= \{v \in V(N) : t(v) < \infty\}$$

$$\mathcal{D} := \{v \in V(N) : d(v), d(v') = \infty\}.$$

The set $\mathcal{C}$ is called the **core** of N. For the time being, we are not interested in the cut $Q(N)$, but only in the connected components of the subnetwork $N[\mathcal{C}]$ which contain precisely the arcs of N with both end nodes in the core. These connected components are called the **nuclei** or **bicursal units** of N. Furthermore, the nucleus containing a node $v \in \mathcal{C}$ is denoted by $U(v)$.

Consider Figure 7 which shows the residual network according to a 2-factor search. In this example, two blossoms exist which consist of 1, 3, 4, and their complementary nodes and s, 7, 8, 11, 12, 13, and their complements. Both blossoms are joined by the unicursal arc $(13, 3')$ and, hence, form a nucleus together. This proves that blossoms and nuclei are, in general, not the same. How-

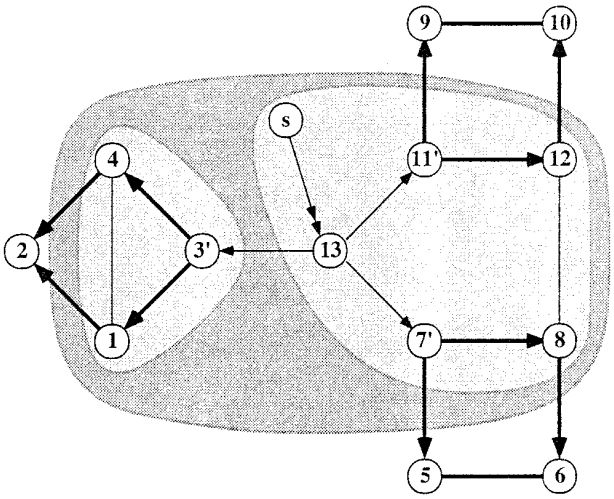

Fig. 7. Blossoms and nuclei.

ever, blossoms and nuclei have a lot of structure in common. We shall see later that every nucleus is the disjoint union of blossoms.

If t is not strictly reachable, we also call the nuclei **odd components.** This terminology is motivated by Tutte's factor theorems [26] which are not discussed here. On the other hand, if t is strictly reachable, s and t are in a common nucleus since every valid st-path is entirely contained in a blossom as noted above.

As a familiar example, consider any residual network $N(f)$. Here, the set T may be empty in special circumstances like the case $f \equiv 0$, while S will at least contain the source s. Using Theorem 4.2, maximum balanced flows can be characterized as follows:

Observation 9.1. *Let N be a balanced flow network, and f, a balanced st-flow on N. Then, the following statements are equivalent:*

(1) *The flow f is a maximum balanced flow.*
(2) *The sink t is not strictly reachable in $N(f)$.*
(3) *The edge cut $Q(N_{\mathfrak{R}}(f))$ is an st-cut.*
(4) *The source s is not contained in any blossom of $N(f)$.* ■

Next, we prove that one can associate with every blossom B of N a unique node, called the **base** of B, which connects this blossom and the remaining nodes of N in a very special manner. A corresponding result for the nuclei of N follows.

Nearly all papers on matching algorithms deal with some notion of blossoms and blossom bases. In the setup of Edmonds [7] who introduced the floristic terminology, the determination of a blossom base is straightforward. Vazirani [28] takes a different approach, defining blossoms by their base and the tenacity labels, but needs a lot of effort to show the crucial uniqueness of blossom bases.

We will use a logical arrangement similar to that of Kocay and Stone [19] and will easily derive general results which can be applied to the analysis of any matching algorithm based on augmentation. We later pay a price for this, since it will be more involved to relate the tenacity labels to the blossoms occurring in the computation of minimum valid paths.

Theorem 9.2 (Base Identity for Blossoms). *Let B is a blossom not containing the source s. Then, there is a prop $a = ub$ with $rescap(a) = 1$ which is traversed by every valid path p connecting s to a node v in B. Furthermore, $p[u] \subseteq S \backslash B$ and $p[b, v] \subseteq B$ hold.*

Proof. Let v be any node in B. Since v is incident with some bicursal arc, v is strictly reachable. Let p be a valid sv-path and $a = ub$ the last unicursal arc on p. Such an arc exists since $B \neq B(s)$ is assumed. We will prove that a is the desired arc.

Since b is incident with some bicursal arc, b' is also strictly reachable. Choose some valid sb'-path q. This path must traverse a since, otherwise, $q \circ a'$ would be a'-accessing. But then a and a' would be bicursal arcs, contradicting the choice of a. For the same reason, $rescap(a) = 1$ holds.

Let x be the first node on $q[b]$ such that x or x' is on $q[b, b']$. If x and b would differ, then either $q[x] \circ q[x, b']$ or $q[x] \circ q[b, x']'$ would be a valid sb'-path not containing a. But, then, a' would be strictly accessible, a contradiction.

Hence, $q[b)$ and $q[b, b']$ are strictly disjoint, $q[b] \circ q[b, b']$ and $q[b] \circ q[b, b']'$ are valid, and all arcs on $q[b, b']$ are bicursal. Thus, we have

$$B_0 := q[b, b'] \cup q[b, b']' \subseteq B.$$

Suppose the existence of a node $x \in B_0$ and a valid sx-path r not containing ub. Let y be the first node in B_0 visited by r. Then, either $r[y] \circ q[y, b']$ or $r[y] \circ q[b, y']'$ would be a valid sb'-path not containing a, a contradiction.

Using induction on $|B_0|$, we prove that also the remaining vertices $z \in B \backslash B_0$ can be reached only if the arc a is traversed. To see this, choose any arc zx with $z \in B \backslash B_0, x \in B_0$ and $rescap(z, x) > 0$. Such an arc exists as long as there are nodes in $B \backslash B_0$.

If some valid sz-path $\tilde{r}$ would not contain the arc a, then $\tilde{r}$ would not traverse B_0 at all, and $\tilde{r} \circ zx$ would be a valid sx-path not containing a, a contradiction. Next, consider a valid sz'-path $\tilde{q}$ and the first vertex $\tilde{y} \in \tilde{r}[b, z] \cup \tilde{r}[b, z]'$ on $\tilde{q}$. We either get the valid sz-path $\tilde{q}[\tilde{y}] \circ \tilde{r}[\tilde{y}, z]$ or the valid sb'-path $\tilde{q}[\tilde{y}] \circ \tilde{r}[b, \tilde{y}']'$, which

proves that a is also on $\tilde{q}[\tilde{y}]$. If we set $B_0 := B_0 \cup \{z, z'\}$ and iterate this induction step until $B = B_0$ holds, the uniqueness of ub is evident.

By choice of a, one has $p[b, v] \subseteq B$. Suppose that $p[u]$ contains some further node x also in B. We already know that $p[x]$ traverses the arc a itself. Since p is simple, we obtain a contradiction.

Finally, to prove the claim that a is a prop, we consider some $d(b')$-path q. Since a is on q, we have $d(b) < d(b')$, that is, b is a minlevel vertex. Furthermore, we know that a is the last arc of any $d(b)$-path. By definition, a is a prop. ■

Corollary 9.3. *Every blossom is self-complementary.*

Proof. First, consider the case $s \notin B$ and the proof of Theorem 9.2: In the beginning, $B_0 = q[b, b'] \cup q[b, b']'$ is self-complementary, and at the end, $B_0 = B$ holds. With induction on the steps $B_0 := B_0 \cup \{z, z'\}$, we obtain that likewise B is self-complementary.

Now, let $B = B(s)$. Altogether, the blossoms partition $Bic(N)$, and by the first case, all blossoms different from B are self-complementary. Obviously, $Bic(N)$, and likewise B are self-complementary. ■

Corollary 9.4. *If B is a blossom of N, then $N[B]$ entirely consists of bicursal arcs.*

Proof. Suppose that xy' is a nonaccessible arc with $y' \in B$. Since y and y' are strictly reachable, every valid sy-path q has to traverse xy'. If we put $b := y'$ and $v := y$, the proof of Theorem 9.2 shows that a and xy' are equal. In particular, x is not in B. ■

Theorem 9.5. *If B is a blossom containing s, and u another node in B, then every valid su-path is entirely contained in B.*

Proof. Suppose that p contains nodes not in B and that vw is the last unicursal arc on p. Since w and w' are strictly reachable, every valid sw'-path q has to traverse vw. As in the last proof, we conclude that all arcs on $q[w, w']'$ are strictly accessible, that is, $B_0 := q[w, w'] \cup q[w, w']' \subseteq B$. Furthermore, all nodes in B_0 can only be reached if vw is traversed, since, otherwise, $w'v'$ would be strictly accessible.

Since B is a connected component of $N[Bic(N)]$, there is some undirected sw-path r in $N[B]$. Let zx be an arc on r with $x \in B_0$, $z \notin B_0$. As in the last proof, it is evident that every valid sz-path and every valid sz'-path must traverse vw. Then, we set $B_0 := B_0 \cup \{z, z'\}$ and choose another arc zx with $x \in B_0$, $z \notin B_0$.

There is only a finite number of arcs on the path r. Using induction on $|r|$, we see that the source s can only be reached if vw is traversed which is absurd. Thus, the hypothetical arc vw cannot exist, and $p \subseteq B$ indeed holds. ■

If B and b are as in Theorem 9.2, we call $base(B) := b$ the **base** of the blossom B and denote the prop a by $prop(B)$. If t is strictly reachable, and $\tilde{B}$ is the blossom containing s and t, we set $base(\tilde{B}) := s$.

We also consider each node v which is strictly reachable, but not on some bicursal arc to be in a blossom, namely, the **bud** $B(v) = B(v') := \{v, v'\}$, and $base(B(v)) := v$ to be the base of this bud. This notion is motivated by the balanced network search algorithms to be discussed later. Where necessary, we call the blossoms other than buds **proper.** Note that buds are likewise self-complementary.

By the base identity, all proper blossom bases are minlevel nodes. Obviously, this is also true for buds. Indeed, buds often behave like proper blossoms in algorithms. However, buds are not connected and can be reached by more than one prop in general.

Corollary 9.6. *Any valid sv-path p traverses any blossom at most once. Every proper blossom $B \neq B(s)$ traversed by p is reached by $prop(B)$.*

Proof. Let x be the last node on p that is also in B. The assertion follows by applying Theorems 9.2 and 9.5 to $p[x]$. ■

Next, we consider minimum valid paths: Let v be in a blossom and $b = base(B(v))$. By the base identity, there are valid bv-paths which are entirely contained in $B(v)$. Let $d(b, v)$ denote the minimum length of such a bv-path. The corresponding bv-path is called a $d(b, v)$-path.

Corollary 9.7. *Let v be in a proper blossom with base $b = base(B(v))$. A path p is a $d(v)$-path iff $p = q \circ r$ holds for a $d(b)$-path q and a $d(b, v)$-path r.* ■

Corollary 9.8. *Let p be a $d(v)$-path; B, a blossom visited by p, $b := base(B)$ the first, and x, the last node of B on p. Then, $p[b, x]$ is a $d(b, x)$-path.* ■

We stress that valid bv-paths p of length $|p| < d(b, v)$ can exist which contain nodes not in $B(v)$. An example for this is given in Figure 8 which depicts the residual graph according to some 1-factor search. Here, exactly one blossom exists, which has base 3, and the $d(3, 5)$-path is $(3, 1', 2, 10', 9, 8', 7, 6', 5)$. The shorter path $p := (3, 1', 2, 3', 4, 6', 5)$ is valid, but visits node 4, which is not in $B(3)$. Note that the concatenation of the only $d(3)$-path $(s, 11, 4', 3)$ and p is not a valid path, since both arcs $(4', 3)$ and $(3', 4)$ are traversed.

As the preceding corollaries show, the determination of minimum valid paths in a balanced network N naturally

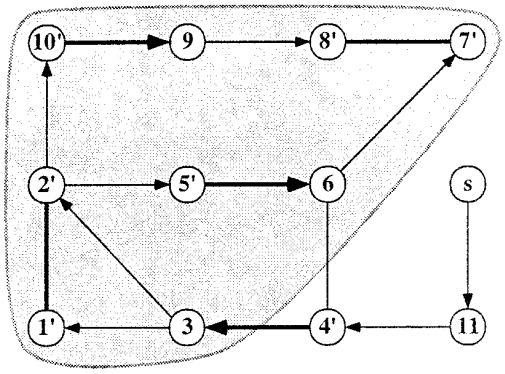

Fig. 8. Traversing a blossom.

splits into two parts, namely, traversing the various blossoms and traveling from blossom to blossom. We first pay attention to the latter task and the construction of the **layered auxiliary network** $Aux(N)$: The nodes of $Aux(N)$ are the blossom bases of N. Two bases u and v are connected by an arc $uv \in A(Aux(N))$ iff there is a predecessor w of v with $u = base(B(w))$. The capacity assigned to the arc uv is

$$auxcap(u, v) := rescap(w, v),$$

and we identify uv and the prop wv sometimes. In particular, $Aux(N)$ has parallel arcs for each prop with end node v and start node in $B(w)$. By application of Corollaries 8.3 and 9.7 to uv, we obtain the inequality

$$d(v) = d(u) + d(u, w) + 1 > d(u). \qquad (17)$$

Thus, the network $Aux(N)$ is acyclic, and for this reason, we use the term *layered*. The **layers** consist of those bases with equal distance label d. Furthermore, $Aux(N)$ is connected, and s is its unique root, as is seen from the following lemma:

Lemma 9.9. *Let b be a blossom base and $p = (x_i)_{i=0}^k$ be some $d(b)$-path with respect to N. Then, $aux(p) := (x_i : x_i \text{ is a blossom base})_{i=0}^k$ determines a directed sb-path in $Aux(N)$. On the other hand, to any directed sb-path $\tilde{p}$ in $Aux(N)$, a $d(b)$-path with respect to N with $\tilde{p} = aux(p)$ exists.*

Proof. Both claims hold trivially if b and s are equal. For any other blossom base, use Corollaries 8.3, 9.7, and induction on $d(b)$. ■

We now establish similar results for the nuclei of N which also describe their relationship to the blossoms. In the context of the cardinality matching problem, both notions coincide which is proven at the end of this section. First, we give a characterization of the bicursal bridges of N.

Theorem 9.10. *Let a be a bridge. The following statements are equivalent: (a) $t(a)$ is finite. (b) a is contained in a proper blossom. (c) a is bicursal. (d) a is contained in a nucleus.*

Proof. Write $a =: uv'$. The implications (c) → (b), (b) → (d), and (d) → (a) are trivial. Thus, we only have to prove the implication (a) → (c).

Hence, $t(a) = d(u) + d(v) + 1$, and, especially, $d(u)$ and $d(v)$ are finite. If a is a loop, that is, $u = v$, then $a = uu' = (uu')' = a'$ is bicursal indeed. Otherwise, we can assume that $d(u) \leq d(v) < \infty$ since a' is also a bridge, and properties (a) and (c) are self-complementary. Let p be a $d(u)$-path. Then, $p \circ a$ is a-accessing. If $p \circ a$ is not valid, then v' is already on p.

Suppose that a is unicursal. Then, a would be traversed by every $d(v)$-path q, and v' would be a minlevel node. Since a is not a prop, there is a $d(v')$-path r that does not contain a. Let x be the first node on r such that x or x' is on $q[v', v]$. Then, either $r[x] \circ q[x, v]$ or $r[x] \circ q[v', x']'$ would be a valid sv-path, a contradiction. ■

Corollary 9.11. *The network $N[Bic(N)]$ has node set $\mathcal{C}$. Hence, every nucleus splits into proper blossoms. Every bud consists of a node in $\mathcal{B}$ and its complementary node in $\mathcal{B}'$.*

Proof. By definition, $N[Bic(N)]$ is a subgraph of $N[\mathcal{C}]$. Let w be a strictly reachable maxlevel node; p, a $d(w)$-path; and x', the predecessor of w on p. Since $x'w$ is not a prop, it is either a co-prop or a bridge. In the former case, $x'w$ is bicursal by definition; in the latter, it is bicursal by Theorem 9.10. Thus, every node in $\mathcal{C}$ is incident with some bicursal arc. ■

Corollary 9.12. *Every nucleus is self-complementary.* ■

Theorem 9.13 (Base Identity for Nuclei). *Let U be a nucleus. If U does not contain the source s, then there is a prop ub with $rescap(u, b) = 1$ which is traversed by every valid path p connecting s to a node $v \in U$, and $p[u] \subseteq S \setminus U$ and $p[b, v] \subseteq U$ hold. If $U = U(s)$, then every valid path which connects s to a node $v \in U$ is entirely contained in U.*

Proof. Suppose that two proper blossoms $B_1, B_2 \subseteq U$ are adjacent by some arc pair a, a'. By Theorem 9.10, we can assume that a is a prop. By the base identity for blossoms, a either is $prop(B_1)$ or is $prop(B_2)$. In the former case, $(base(B_2), base(B_1))$ is an arc of $Aux(N)$; in the latter, $(base(B_1), base(B_2))$ is an arc of $Aux(N)$.

By the base identity, every blossom base has a unique

predecessor in $Aux(N)$. Hence, the blossom bases of the nucleus U span an arborescence T in $Aux(N)$ since U is connected. Let b be the root of T. Then, every sv-path meets b by Lemma 9.9. The remainder of the proof is a straightforward application of Theorems 9.2 and 9.5. ■

Corollary 9.14. *Two blossoms are connected by at most one arc pair, namely, a prop and a co-prop.* ■

Corollary 9.15. *Let G be a simple graph; f, some balanced 0–1-flow on N_G; and B, a proper blossom of $N := N_G(f)$. Then, $b := base(B)$ is an outer node or $b = s$ holds. Furthermore, B is a nucleus of N.*

Proof. Assume that $b \neq s$, and let p be a valid sb'-path. By the base identity, p visits the base b. Furthermore, p is strictly simple. Thus, the respective successors u and v of b on $p[b, b']$ and $p[b, b']'$ are different. If b would be an inner node, bu and bv would be backward edges w.r.t. N_G. Thus, $f(ub) = cap(ub) = 1$ and $f(vb) = cap(vb) = 1$ would hold, implying that $f(bt) \geq 2$, which is incompatible with $cap(bt) = 1$. This proves the first assertion.

Let $U(b)$ be the nucleus containing B. If $N[U(b)]$ entirely consists of bicursal arcs, then $U(b)$ is a blossom, that is, $U(b) = B$. Suppose, otherwise, so that $u'w$ is a unicursal arc contained in $U(b)$. Then, $B(u)$ and $B(w)$ are proper blossoms. By Theorem 9.5, $B(w)$ and $B(s)$ are different. By the proof of the last theorem, we have $u'w = prop(B(w))$. By the first assertion, w and u are outer nodes.

Let p be a $d(u)$-path, and x', the predecessor of u on p. If x would be in $V(G)'$, then $f(ux') = f(uw') = 1$, $u \neq w$, and $f(su) \leq cap(su) = 1$ would hold. Otherwise, we would have $x' = s$ and $f(su) = 0$. In either case, $f(su)$ and $\Sigma_{v \in V(G)} f(uv')$ would differ, contradicting the flow conservation condition. ■

10. SHRINKING BLOSSOMS

In what follows, let α be any **balanced network search** algorithm, that is, an algorithm which computes the distance labels d, a system of $d(v)$-paths, or simply any valid st-path with respect to the balanced network N. We wish to develop tools for the analysis of a wide range of known BNS algorithms.

A valid sv-path with respect to N is typically computed as follows: First, α computes the blossoms of N during the so-called **shrinking phase** and chooses some directed path $\tilde{p}$ in $Aux(N)$ connecting s and $base(v)$. After that, α reconstructs a valid sv-path p with respect to N and $aux(p) = \tilde{p}$. One says that the path $\tilde{p}$ is **expanded** to the path p. Since there may be several paths p with $aux(p) = \tilde{p}$, the **expansion phase** need not be deterministic.

For the most important application, namely, computing a $d(t)$-path, the usage of blossoms and the layered network is rather limited, since either t is not strictly reachable or every $d(t)$-path is entirely contained in the blossom $B(s)$. For this reason, all popular algorithms reuse for expanding the information which has been collected during the shrinking phase. This makes the expansion process deterministic. In this section, we discuss the recursive construction of blossoms during the shrinking phase of α.

Surely, any algorithm α has to investigate some arcs $a \in A(N)$, where *investigation* denotes that moment when α decides which nodes in $V(N)$ can be reached using the arc a. (Note that a could have been explored by α earlier and put aside, as, e.g., in the algorithm of Micali and Vazirani [23]). The order in which the arcs of N are investigated is called the **search strategy** of α. Our approach is based upon this strategy only so that we are able to analyze an algorithm even if it does not utilize blossoms and layered auxiliary networks explicitly (as, e.g., the algorithm of Kameda and Munro [18]).

We assume that the algorithm α always investigates complementary arcs in pairs, say, in order $a_1, a_1', a_2, a_2', \ldots, a_k, a_k'$, and that α halts with the investigation of a_k and a_k'. In the beginning, the set of arcs investigated by α is $A_0 := \emptyset$. After the investigation of a_i and a_i', it is the set $A_i := \{a_j, a_j' : 1 \leq j \leq i\}$.

For any self-complementary arc set $A \subseteq A(N)$, we denote by $N[A]$ the subnetwork of N with node set $V(N)$ and arc set A, and by $d_A(v)$, the minimum length of valid sv-paths with respect to $N[A]$. This network $N[A]$ is balanced again, so that all results of Sections 8 and 9 apply to $N[A]$; in particular, this holds if A is taken to be one of the sets A_i occurring under α. Of course, we have to take the labels $t(v)$ and also the terms blossom $B(v)$, $base(B)$, and $aux(p)$ relatively to A. For the sake of clarity, we speak of (A)-blossoms, strictly (A)-reachable nodes and (A)-valid paths, and also use appropriate indices.

We call those node sets which are (A_i)-blossoms for some $1 \leq i \leq k$ the **α-blossoms** of N. The next lemma describes the recursive construction of α-blossoms, an important tool within the design of matching algorithms. It shows that the α-blossoms form a **nested** or **laminar set family,** called the **α-shrinking family.**

Lemma 10.1. *Let $1 \leq j \leq i \leq k$, and let B_j be an (A_j)-blossom, and B_i, an (A_i)-blossom. Then, either B_i and B_j are disjoint or B_j is contained in B_i. In either case, B_j is contained in some (A_i)-blossom.*

Proof. Every (A_j)-valid path is entirely contained in $N[A_i]$. Thus, $d_{A_i}(v) \leq d_{A_j}(v)$ holds for all nodes $v \in V(N)$. If B_j is a bud, the assertion follows by the self-complementarity of A_i. Otherwise, every (A_j)-bicursal

arc xy' contained in B_j is also (A_i)-bicursal. Hence, at least the (A_i)-blossom $B_{A_i}(x, y)$ has nodes with $B_j = B_{A_j}(x, y)$ in common.

Suppose that B_j and B_i have a node u in common. Choose another node $v \in B_j$. Then, there is an undirected path in $N[Bic(N[A_j])]$ connecting u and v. Because of $Bic(N[A_j]) \subseteq Bic(N[A_i])$, u and v are in a common (A_i)-blossom. Indeed, this blossom is B_i. ■

We will now prove a somewhat stronger result, namely, that every (A_{j+1})-blossom either is the disjoint union of (A_j)-blossoms or a bud. For this, it is necessary to make some assumptions about the performance of α. We assume that every set $A = A_i$ satisfies the following conditions:

$(\alpha 1)$ A is self-complementary.

$(\alpha 2)$ A does not contain (A)-acursal arcs.

$(\alpha 3)$ Every (A)-unicursal arc is an (A)-prop.

While condition $(\alpha 1)$ is crucial for any application of our preceding results, $(\alpha 2)$ ensures that the investigated subgraph $N[A]$ grows like a search tree. Condition $(\alpha 3)$ forces every (A)-valid path p to induce a directed path $aux_A(p)$ in $Aux(N[A])$. Informally spoken: If $(\alpha 3)$ holds, one can ''detour'' only within an (A)-blossom. To show that every (A_{j+1})-blossom splits into (A_j)-blossoms, we only need conditions $(\alpha 1)$ and $(\alpha 2)$:

Theorem 10.2 (Bud Generation). *Assume that the arc set $A \subseteq A(N)$ satisfies conditions $(\alpha 1)$ and $(\alpha 2)$. Let $a = uv'$ be an arc in $A(N)\backslash A$ with $d_A(u) < \infty$ and $d_A(v) = \infty$. Then, the following assertions hold for the arc set $\tilde{A} := A \cup \{a, a'\}$:*

(a) $\mathcal{C}_{\tilde{A}} = \mathcal{C}_A$.

(b) $\mathcal{B}_{\tilde{A}} = \mathcal{B}_A \cup \{v'\}$.

(c) Every proper $(\tilde{A})$-blossom is an (A)-blossom.

Proof. Write $\hat{A} := A \cup \{a\}$. Because of $d_A(v) = \infty$, every (A)-valid su-path which does not already visit v' can be extended by a to an $(\hat{A})$-valid sv'-path. Thus, $d_{\hat{A}}(v') < \infty$ holds.

Let p be an $(\hat{A})$-valid path starting at s and ending at some node w which traverses a. Suppose that $w \neq v'$, and let y be the successor of v' on p. Since $(\alpha 2)$ is required for $v'y, y'v \in A$, the node v' would be even strictly (A)-reachable. Let q be an (A)-valid sv'-path, and x, the first node on q for which x or x' is on $p[v', w]$. If x would be on $p[v', w]$, then $q[x] \circ p[x, w]$ would be an (A)-valid sw-path. Otherwise, $q[x] \circ p[v', x']'$ would be an (A)-valid sv-path, contradicting the choice of v.

This proves that v' is the only node that becomes strictly reachable when a is investigated. In particular, $d_{\tilde{A}}(v)$ is still infinite, and no $(\tilde{A})$-valid s-path contains the arc vu'. This implies that $\mathcal{C}_{\tilde{A}} = \mathcal{C}_A$ and $\mathcal{B}_{\tilde{A}} = \mathcal{B}_A \cup \{v'\}$. Furthermore, a is the only $(\tilde{A})$-accessible arc which was not strictly (A)-accessible yet. Hence, all $(\tilde{A})$-bicursal arcs are (A)-bicursal. ■

Theorem 10.3 (First Shrinking Property). *Assume that the arc set $A \subseteq A(N)$ satisfies conditions $(\alpha 1)$ and $(\alpha 2)$. Let $a = uv'$ be an arc in $A(N)\backslash A$ with $d_A(u)$, $d_A(v) < \infty$. Then, the following assertions hold for the arc set $\tilde{A} := A \cup \{a, a'\}$:*

(a) $\mathcal{D}_{\tilde{A}} = \mathcal{D}_A$.

(b) Both u and v are in a common proper $(\tilde{A})$-blossom, denoted by $B_{\tilde{A}}(u, v)$.

(c) Except for $B_{\tilde{A}}(u, v)$, every $(\tilde{A})$-blossom is an (A)-blossom.

Proof.

(a) Suppose that $w \in \mathcal{D}_A$ for some strictly $(\tilde{A})$-reachable node w. Let x' be the predecessor of w on some $(\tilde{A})$-valid path. Since w' was not strictly (A)-reachable, the arcs $x'w$ and $w'x$ were already in A. By condition $(\alpha 2)$, $x'w$ would have been strictly (A)-accessible, contradicting $d_A(w) = \infty$. Thus, $\mathcal{D}_{\tilde{A}} = \mathcal{D}_A$ holds.

(b) Every (A)-valid sv-path can be extended to a a'-accessing path since a is not in A, and v is assumed to be strictly (A)-reachable. For the same reason, a is $(\tilde{A})$-accessible. Since u and v are adjacent, and a and a' are $(\tilde{A})$-bicursal, both nodes are in a common proper $(\tilde{A})$-blossom.

(c) Suppose the existence of a proper $(\tilde{A})$-blossom other than $B_{\tilde{A}}(u, v)$ which is not an (A)-blossom. Such a blossom could only arise if an $(\tilde{A})$-bicursal arc pair xy', yx' with $x, y \notin B_{\tilde{A}}(u, v)$ exists which is not (A)-bicursal.

By condition $(\alpha 2)$, we can assume xy' to be strictly (A)-accessible. Let p be a xy'-accessing path in $N[A]$, and q, a yx' accessing path in $N[\tilde{A}]$. Each of p and q visits $b := base_{\tilde{A}}(x, y)$, and $q[b, y]$ would be entirely contained in $B_{\tilde{A}}(x, y)$ by the base identity. Thus, $q[b, y]$, and, in turn, $p[b] \circ q[b, y]$ would be even (A)-valid. Obviously, xy' cannot be on $p[b]$ so that $r \circ yz'$ would be yx'-accessing, a contradiction. ■

Corollary 10.4. *Let A, a, and $\tilde{A}$ be as in Theorem 10.3, and let $B_A(u) = B_A(v)$. Then, by the investigation of a and a', no node becomes strictly reachable, and no arcs but a and a' become strictly accessible.*

Proof. Let p be an $(\tilde{A})$-valid path which traverses a or a'. Then, $B_A(u, v)$ is reached by $base_A(u, v)$ first, and

the part between the first and the last node of $B_A(u, v)$ on p can be exchanged by some (A)-valid path. ■

Theorem 10.2 deals with those iterations of α during which new buds can arise. Regarding the remaining iterations j of α, Theorem 10.3 shows that any new (A_{j+1})-blossom must be the disjoint union of (A_j)-blossoms. In both situations, at most one new α-blossom can arise; alternatively, the blossom structure may also stay unchanged.

The arc a investigated in the situation of Theorem 10.2 is called an α-prop of v', whereas the arc a investigated in the situation of Theorem 10.3 is called an α-**bridge.** Note that α-props and props are not the same. These notions coincide only for special BNS algorithms which compute the correct distance labels.

Next, we want to know which (A)-blossoms form the $(\tilde{A})$-blossom $B_{\tilde{A}}(u, v)$ occurring in Theorem 10.3. At this point, condition $(\alpha 3)$ and the layered network $Aux(N[A])$ come into play.

Lemma 10.5. *Assume that $A \subseteq A(N)$ is self-complementary. Let $a = uv'$ be an arc in $A(N)\backslash A$ with $d_A(u)$, $d_A(v) < \infty$, $\tilde{A} := A \cup \{a, a'\}$, and w, a strictly $(\tilde{A})$-reachable node with $d_A(w) = \infty$. Then, an $(\tilde{A})$-valid sw-path exists which traverses either a or a', but not both.*

Proof. Suppose otherwise that every $(\tilde{A})$-valid *sw*-path p traverses both a and a'. Without loss of generality, we can assume that a' is traversed before a. Let r be a $d_A(u)$-path, and y_1x_1', the last arc on r such that x_1y_1' is on $p[v', w]$. (If such an arc would not exist, $r \circ p[u, w]$ would be an $(\tilde{A})$-valid path not containing a' or could be restricted to such an (A)-valid *sw*-path.) Hence, $r[x_1', u]' \circ p[x_1, w]$ can be restricted to an (A)-valid $u'w$-path q.

By our assumption, $p[u'] \circ q$ is not valid and cannot be restricted to an $(\tilde{A})$-valid *sw*-path. Thus, arcs on $p[v]$ exist that are also on $r[x_1', u]$. Let y_2x_2' be the first such arc on $p[v]$. Then, $p[y_2] \circ r[y_2, u] \circ p[u, w]$ is valid and does not traverse a', contradicting the hypothesis.■

Theorem 10.6 (Second Shrinking Property). *Assume that $A \subseteq A(N)$ satisfies conditions $(\alpha 1)-(\alpha 3)$. Let $a = uv' \in A(N)\backslash A$ be an arc with $d_A(u)$, $d_A(v) < \infty$, and $\tilde{A} := A \cup \{a, a'\}$. Furthermore, let w be a strictly $(\tilde{A})$-reachable node with $d_A(w) = \infty$; p, a minimum length $(\tilde{A})$-valid sw-path visiting a but not a'; and q, a $d_A(w')$-path. Then, $p[u]$ is a $d_A(u)$-path and $q \circ p[v', w]'$ is a $d_A(v)$-path.*

Proof. Since $p[u]$ is (A)-valid, $aux_A(p[u])$ is a directed path in $Aux(N[A])$. We will show that also $\tilde{r} := aux_A(p[v', w])$ is a directed path in $Aux(N[A])$.

First, let $B = \{x, x'\}$ be an (A)-bud visited by $p[v', w)$; x, the base of B, and r, a $d_A(x)$-path. Suppose that x was on $p[v', w]$. Let y be the first node on r such that y or y' is on $p[x, w]$. If y would be on $p[x, w]$, then $r[y] \circ p[y, w]$ would be an (A)-valid *sw*-path, contradicting the choice of w. Otherwise, $r[y] \circ p[x, y']'$ would have been an (A)-valid *sx'*-path, contradicting the choice of x. Thus, x' is on $p[v', w]$. The arc by which x' is left on p is the complementary arc of an (A)-prop by condition $(\alpha 3)$.

Now, let B be a proper (A)-blossom traversed by $p[v', w]$; x, the last vertex of B on p; and r, a $d_A(x)$-path. Let y be the first node on r such that y or y' is on $p[x, w]$. If y would be on $p[x, w]$, then $r[y] \circ p[y, w]$ would be an (A)-valid *sw*-path, a contradiction. Otherwise, $r[y] \circ p[x, y']'$ is an (A)-valid *sx'*-path that reaches B via $prop_A(B)$ by the base identity.

Suppose that $p[x, w]$ does not traverse $prop_A(B)'$. Then, $p[x, y']$ would be entirely contained in B. Choosing $b := base_A(B)$ and a $d_A(b, y')$-path $\tilde{r}$, we would have $r[b] \circ \tilde{r} \circ p[x, w]$ as an (A)-valid *sw*-path, a contradiction. Thus, $prop_A(B)'$ is an arc on $p[x, w]$. By the choice of x, we have $x = base_A(B)'$.

Now, let z be the first node of B on p. We already know that z is on $p[v', x]$, since, otherwise, p would not only traverse $prop_A(B)$ but also $prop_A(B)'$. Hence, $p[u]$ and $p[v', w]$ have no proper blossoms in common.

To see that $p[z, x]$ is entirely contained in B, choose some $d_A(x')$-path r. Let y be the first node on r such that y or y' is on $p[z, x]$ and assume that $y \neq x'$. Then, either $r[y] \circ p[y, x]$ or $r[y] \circ p[z, y']'$ would be (A)-valid with end node in B. But, then, p would either traverse $prop_A(B)$ and also $prop_A(B)'$ or would traverse $prop_A(B)'$ twice. Thus, $y = x'$ holds, $r \circ p[z, x]'$ is (A)-valid, and $p[z, x]$ is contained in B.

But, then, $p[z, x]'$ is an $d(x', z')$-path by the minimality of p. It is now evident that $\tilde{r}$ is a directed path in $Aux(N[A])$ and that $aux_A(q) \circ \tilde{r}$ can be expanded into the $d_A(v)$-path $q \circ p[v', w]'$ again. Furthermore, $p[u]$ is a $d_A(u)$-path and $aux_A(p[u])$ a directed path in $Aux(N[A])$ as an immediate consequence of Lemma 9.9. ■

If a and w are as in Theorem 10.6, we call a the ***α*-petal** of w. It is obvious that the knowledge of the α-petal is crucial for the reconstruction of some *sw*-path and that this arc is, therefore, stored by typical BNS algorithms. We now describe how to determine blossom bases and the blossoms that have to be merged together during an arc investigation step.

Lemma 10.7. *Assume that $A \subseteq A(N)$ satisfies conditions $(\alpha 1)-(\alpha 3)$. Let p be a valid cycle with respect to $N[A]$. Then, p is contained in a proper (A)-blossom.*

Proof. For any node x on p, at least one of x and x' is strictly (A)-reachable, by condition $(\alpha 2)$. Since p' is

also a valid cycle, we may assume that p contains strictly (A)-reachable nodes. Pick such a node y for which $d_A(y)$ is minimal, and let q be a $d_A(y)$-path.

Now, let z' be the predecessor of y on p. Then, $q[y] \circ p[y, z']$ is (A)-valid. If $z'y$ is (A)-bicursal, then y and z are in a common (A)-blossom and we have $p \subseteq B_A(y, z)$ by the base identity. Otherwise, either $z'y$ or $y'z$ is an (A)-prop by conditions $(\alpha 2)$ and $(\alpha 3)$. But, then, z or z' would have been traversed by $q[y]$, contradicting the choice of y. ■

Theorem 10.8. *Let A, uv', and $\tilde{A}$ be as in the shrinking properties; B, an (A)-blossom; and $b, := base_{\tilde{A}}(u, v)$. Then, B is contained in $B_{\tilde{A}}(u, v)$ iff $base_A(B)$ is on some directed $(b, base_A(u))$-path or $(b, base_A(v))$-path in $Aux(N[A])$.*

Proof. First assume that B is a blossom contained in $B_{\tilde{A}}(u, v)$. If $base_A(B)$ and b are equal, the statement is evident. Thus, assume that $base_A(B) \neq b$ in what follows:

First, let B be an (A)-bud. By Theorem 10.3, the first shrinking property, $B_{\tilde{A}}(u, v)$, is a proper blossom. Therefore, $w := base_A(B)'$ is strictly $(\tilde{A})$-reachable, but not (A)-reachable. W.l.o.g. let p and q be sw- and sw'-paths as in Theorem 10.6. (If such paths do not exist, exchange a and a'.) Then, $r := q \circ p[v', w]'$ is a $d_A(v)$-path by the second shrinking property, and $aux(r[b, v])$ is the requested path in $Aux(N[A])$.

Now, let B be a proper (A)-blossom and $wz := prop_A(B)$. Since $z \neq b$ is required, wz is contained in $B_{\tilde{A}}(u, v)$ by Corollary 9.4 and the base identity. Thus, wz is $(\tilde{A})$-bicursal, but (A)-unicursal.

Suppose that p is a $z'w'$-accessing path in $N[\tilde{A}]$. Without loss of generality, we can assume that uv' is on p, but vu' is not on $p[v', w']$, since, otherwise, u and v can be exchanged. If w' would be already on $p[v', z']$, then $z'w'$ would be (A)-bicursal by the preceding lemma, contradicting the choice of $wz = prop_A(B)$.

Let q be a $d_A(w)$-path, and x', the first node on $p[v', w']$ such that $q \circ p[x', w']'$ is valid. If x and v are equal, then $r := q \circ p[v', w']'$ is (A)-valid, and $aux(r)$ is directed in $Aux(N[A])$. By the base identity, b is reached on $aux_A(r)$ before wz is traversed.

Let y be the predecessor of x' on $p[v', w']$ otherwise. If yx' would be also on q, then $q[x', w] \circ p[x', w']'$ would be contained in a proper (A)-blossom by the base identity. Otherwise, x would be already on q, and $q[x, w] \circ p[x', w']'$, an (A)-valid cycle which is contained in a proper (A)-blossom by the preceding lemma. In either case, wz would be (A)-bicursal, a contradiction.

The converse direction follows by the base identity and Lemma 9.9. ■

Since $base_{\tilde{A}}(u, v)$ must be determined by a more or less sophisticated search procedure, this theorem cannot be used for the construction of $B_{\tilde{A}}(u, v)$ immediately. In fact, all known BNS algorithms know the members of $B_{\tilde{A}}(u, v)$ in the moment when its base is recognized. But how can the base determination be managed?

Let a be an arc of the layered auxiliary network $Aux(N)$ with $auxcap(a) = 1$, and let v be some blossom base of N. If a is on every directed sv-path in $Aux(N)$, we call a **a bottleneck** of v. Note that the bottlenecks of v are ordered by the distance labels of their end nodes. Actually, BNS algorithms do not compute $b := base_{\tilde{A}}(u, v)$, but, rather, the arc $bot_A(u, v)$ of the layered auxiliary network $Aux_A(N)$ which denotes the (A)-bottleneck of $base_A(u)$ and $base_A(v)$ with maximum distance label.

Corollary 10.9. *Let x denote the unique (A)-predecessor of $b := base_{\tilde{A}}(u, v)$. Then, $bot_A(u, v)$ and $(base_A(x), b)$ are the same.*

Proof. It is obvious that $(base_A(x), b)$ is an (A)-bottleneck of $base_A(u)$ and $base_A(v)$. Suppose yz to be an (A)-bottleneck of $base_A(u)$ and $base_A(v)$ with $d_A(b) < d_A(z)$. Then, $B_A(y)$ and $B_A(z)$ must be parts of $B_{\tilde{A}}(u, v)$ by Theorem 10.8.

Since yz is a bottleneck, there is only one (A)-predecessor w of z and also $rescap(w, z) = 1$ holds. The arc wz would be (A)-unicursal and $(\tilde{A})$-bicursal; then, hence, any $z'w'$-accessing path in $N[\tilde{A}]$ would traverse uv' or vu'. But u and v can be reached in $N[A]$ only by traversing wz, a contradiction. ■

11. TREE-GROWING BNS ALGORITHMS

In contrast to conditions $(\alpha 1)$ and $(\alpha 2)$ of the preceding section, it is not obvious how to design a BNS algorithm which satisfies condition $(\alpha 3)$. There are different concepts for this. In the present section, we replace $(\alpha 3)$ by the more restrictive condition

$(\alpha 4)$ At most one (A)-unicursal arc with given end node v exists.

This simplifies the computation of the arc $bot_A(u, v)$ considerably. If an algorithm α satisfies conditions $(\alpha 1)$–$(\alpha 4)$, the occurring layered auxiliary networks are trees. However, such an algorithm will find *minimum* valid paths only in special circumstances.

Unlike $(\alpha 1)$ and $(\alpha 2)$, a search strategy based on Theorems 10.2 and 10.3 does not maintain condition $(\alpha 4)$: In the situation of Theorem 10.2, the node v' may be already strictly (A)-reachable, in which case we call a an ***α*-anomaly** of v'.

One could expect that the investigation of anomalies is redundant, but note that the node v may become strictly

reachable later. Then, we say that the anomaly a is **resolved** since a becomes available for a blossom shrinking operation according to Theorem 10.3. We will study an example for this phenomenon in our forthcoming paper [10] where the algorithm of Kameda and Munro is discussed. This algorithm does not investigate resolved anomalies at all and, indeed, is incorrect.

A search strategy which does not investigate α-anomalies before they are resolved is called **tree growing.** For this case, we can prove that the iterated arc set $\tilde{A}$ satisfies conditions $(\alpha 1)-(\alpha 4)$ whenever A does and, therefore, that the network $Aux(N[A_{i+1}])$ is a tree again. The algorithms of Edmonds, Kameda/Munro, and Kocay/Stone are tree growing. Especially for Edmonds' algorithm [7], the layered auxiliary networks are called **planted trees.**

Lemma 11.1. *Let $A \subseteq A(N)$ be self-complementary; $w \in V(N)$, a node; and $d(w) < d_A(w) = \infty$. Then, an arc $a \in A(N)\backslash A$ exists which either satisfies the hypothesis of Theorem* 10.3 *or satisfies the hypothesis of Theorem* 10.2 *and is not an anomaly.*

Proof. We choose that node $w \in V(N)$ with $d(w) < d_A(w) = \infty$ which has minimum $d(w)$ and a $d(w)$-path p. Let $a = uv'$ be the last arc on p which is not in A. If v' and w are equal, then v' is not in $\mathcal{B}_A$, and $d_A(u)$ is finite by choice of w.

Otherwise, $d_A(u)$ and $d_A(v')$ are finite by choice of w. Let q be a $d_A(v')$-path, and y, the first node on q such that y or y' is on $p[v', w]$. If y would be on $p[v', w]$, then $p[y]\circ p[y, w]$ would be an (A)-valid sw-path, a contradiction. Thus, $q[y]\circ p[v', y']'$ is an (A)-valid sv-path, and v' is in $\mathcal{O}_A$. ■

Theorem 11.2 (Termination). *Let α be some tree-growing BNS algorithm and $a_1, a_1', a_2, a_2', \ldots, a_k, a_k'$ the order of arc investigation with respect to some balanced network N.*

Then, A_i satisfies the conditions $(\alpha 1)-(\alpha 4)$ for $i = 1, 2, \ldots, k$. If there is no further arc $a_{k+1} = uv' \notin A_k$ as considered in Lemma 11.1*, then every strictly reachable node is already strictly (A_k)-reachable, and the blossoms of N are exactly the (A_k)-blossoms.*

Proof. The set A_0 trivially satisfies $(\alpha 1)-(\alpha 4)$. We assume A_i to satisfy these conditions and use induction on i. It is obvious that $(\alpha 1)$ and $(\alpha 2)$ hold. Thus, we only have to prove condition $(\alpha 4)$ since $(\alpha 3)$ is less restrictive by the base identity.

If v_{i+1}' is in $\mathcal{D}_{A_i}$, that is, if a new bud is created, then v_{i+1}' is not incident with any arc in A. If $d_i(v_{i+1})$ is finite, that is, in the case of a shrinking operation, then each of uv' and vu' are strictly accessible. Thus, condition $(\alpha 4)$ holds.

Assume that the hypothesis of the second assertion holds. Then, all strictly reachable nodes are strictly (A_k)-reachable by the preceding lemma. In particular, $\mathcal{O}$ and $\mathcal{O}_A$ are equal. By the hypothesis, no arc connecting two proper (A_k)-blossoms exists. ■

Theorem 11.3. *Let $a \in A(N)$ and $rescap(a) = 1$. Then, a is bicursal iff a is an α-bridge for some tree growing BNS algorithm α.*

Proof. $(\rightarrow)$ Let p and q be a- and a'-accessing paths, respectively, and $a = uv'$. Let x be the last node on $q[v]$ such that x or x' is traversed by $p[u]$. If x' is on $p[u]$, then each of x and x' is strictly $(\tilde{A})$-reachable where we put $\tilde{A} := p[u] \cup p[u]' \cup q[v] \cup q[v]'$. Let $y := base_{\tilde{A}}(B_{\tilde{A}}(x))$ in this case, and $y := x$ otherwise.

In either case, $p[y] \circ q[y, v]$ is valid; hence, $A := p[u] \cup p[u]' \cup q[y, v] \cup q[y, v]'$ satisfies conditions $(\alpha 1)$ and $(\alpha 2)$. To see that $(\alpha 4)$ is also fullfilled, suppose that two different (A)-unicursal arcs with a common end node exist. One of these arcs would be on $p[y, u]$, and the other, on $q[y, v]$, contradicting the choice of x.

An appropriate BNS algorithm α will first investigate the arcs of $p[u]$ in the order given by $p[u]$ and then investigate the arcs of $q[y, v]$ in the given order. For the above reasons, conditions $(\alpha 1)-(\alpha 4)$ are also satisfied by the intermediate arc sets. At last, uv' and vu' are investigated. Since u and v are strictly (A)-reachable, a indeed is an α-bridge.

$(\leftarrow)$ If $a = uv'$ is an α-bridge as in the first shrinking property, then u and v are strictly (A)-reachable for some arc set $A \subseteq A(N)$ with $uv', vu' \notin A$ occurring during the course of α. But, then, any $d_A(u)$- and $d_A(v)$-paths can be extended to uv'- and vu'-accessing paths, respectively. ■

We can give a simple path expansion rule that works for all tree-growing BNS algorithms. To this end, we must assign at the point of investigating $a_{j+1} = uv'$

(p1) $prop[v'] := a_{j+1}$, if neither v nor v' are strictly (A_j)-reachable,
(p2) $petal[w] := a_{j+1}$, if Theorem 10.6 applies.

Now, for every strictly reachable node x, some valid sx-path can be obtained. At this point, we can give a correctness proof for the expansion rule which is defined by $path(x, x) := (x)$,

$$path(x, y) := path(x, z') \circ prop[y],$$

if $prop[y] = z'y$ is assigned, and, if $petal[y] = uv'$ is assigned,

$$path(x, y) := path(x, u) \circ petal[y] \circ [path(y', v)^-]'.$$

Theorem 11.4 (Path Expansion). *Let α be a tree-growing BNS algorithm; N, a balanced network; and $a_1, a_1', a_2, a_2', \ldots, a_k, a_k'$, the order in which α investigates the arcs of N.*

Let w be a strictly reachable node, $j(w) := min\{i : 0 \le i \le k, d_{A_i}(w) < \infty\}$ be the index of the investigation step during which w becomes strictly reachable, and $j \ge j(w)$. Let $(x_i)_{i:=0}^n$ be the directed path in $Aux(N[A_j])$ with start node $x := x_0$ and end node $x_n = base_{A_j}(B_{A_j}(w))$. Then, $path(x, w)$ is an (A_j)-valid xw-path which splits into parts

$$path(x_0, x_1) \circ path(x_1, x_2) \circ \cdots \circ path(x_{n-1}, x_n) \circ path(x_n, w).$$

Proof. In the case of $w = x$, especially $w = s$, the assertion is evident. Furthermore, $w = s$ is the only case with $j(w) = 0$. Thus, assume that $w \neq x$, and, as induction hypothesis, that the assertion holds for every $\tilde{w} \in V(N)$ with $j(\tilde{w}) < j(w)$.

Let $\tilde{A} := A_{j(w)}$ and $A := A_{j(w)-1}$. If $prop[w]$ is defined (i.e., if we are in the situation of Theorem 10.2), let $\tilde{w} := prop[w]^-$. Since w has become strictly reachable during the investigation of $(\tilde{w}, w)$, $\tilde{w}$ is strictly (A)-reachable, but w is not. By the induction hypothesis, $path(x, \tilde{w})$ determines an (A)-valid $x\tilde{w}$-path which splits into parts

$$path(x_0, x_1) \circ path(x_1, x_2) \circ \cdots \circ path(x_{m-1}, x_m) \circ path(x_m, \tilde{w}).$$

Obviously, $path(x, \tilde{w}) \circ prop[w]$ is an $(\tilde{A})$-valid xw-path. But $path(x_m, \tilde{w}) \circ prop[w]$ and $path(x_m, w)$ are the same. The assertion now follows:

If $u := petal[w]^-$ and $v := (petal[w]')^-$ are defined (i.e., if we are in the situation of the shrinking properties), then u and v are strictly (A)-reachable. By the induction hypothesis, $p := path(x, u)$ and $q := path(w', v)$ are (A)-valid paths.

Of course, p and q may have an arc a in common. However, since there are no paths alternative to $aux(p)$ and $aux(q)$ in $Aux(N[A])$, and by the second shrinking property, there is no proper (A)-blossom which is traversed by p and also by q. Hence, a is unicursal, that is, an (A)-prop, and every $d_A(u)$-path and every $d_A(v)$-path traverses a. By the second shrinking property, we have $rescap(a) > 1$.

Thus, $p \circ petal[w] \circ q[w', v]'$ is an $(\tilde{A})$-valid sw-path. The assertion follows by the identity

$$path(x, u) \circ (u, v') \circ [path(w', v)]' = path(x, w).$$

12. THE COMPOSITION OF MINIMUM VALID PATHS

There is another concept for a balanced network search which is more sophisticated than tree-growing procedures and which yields the distance labels and a $d(v)$-path for all nodes $v \in V(N)$ implicitly. Here, a search strategy is chosen such that in addition to conditions $(\alpha 1)-(\alpha 3)$ every strictly (A)-reachable node v already has $d_A(v) = d(v)$ and such that any pair of complementary vertices is first reached at the minlevel node.

Subsequent investigation steps do not influence the distance labels of strictly (A)-reachable nodes since these labels are already correct. Even more, an (A)-prop is also an $(\tilde{A})$-prop, and one can decide ad hoc whether or not the current investigated arc a is an $(\tilde{A})$-prop.

Unfortunately, under these conditions, the algorithm α cannot be implemented in such a way that the occurring layered auxiliary networks are trees. Thus, the computation of bottlenecks requires a sophisticated search procedure running on $Aux(N[A])$, which was named **double depth first search** (DDFS) by Micali and Vazirani [23]. Informally spoken, the DDFS tries to derive two disjoint su- and sv-paths. This procedure will be described in our forthcoming paper [11].

We will now discuss the arc sets A corresponding to a search strategy which ensures that the conditions $(\alpha 1)-(\alpha 3)$ of the last section hold and that all strictly (A)-reachable nodes have correct distance $d_A(v) = d(v)$. Here, the tenacity labels will play an important role. The statements that follow do not only describe a specific algorithm, but also give a perception of the general structure of minimum valid paths.

Let N be a balanced network and $i \in \mathbf{N}_0$. Then, we call the bridges uv' with $t(u, v') \le 2i - 1$ the (i)-**bridges** and the props $u'v$ with $d(v) \le i$ the (i)-**props.** The (i)-bridges, the (i)-props, and the co-(i)-props together are the (i)-**arcs.** We are interested in the network N_i which is the restriction of N to the set of (i)-arcs.

For the moment, let A denote the set of (i)-arcs for some $i \in \mathbf{N}_0$. We then speak of (i)-valid paths instead of (A)-valid paths, (i)-blossoms, (i)-bases, and (i)-reachability and write $d_i(v) := d_A(v), B_i(v) := B_A(v)$ as well as

$$\mathcal{O}_i := \{v \in V(N) : d_i(v), d_i(v') < \infty\}.$$

Assume that some algorithm α has investigated the set A of (i)-arcs at some point and the set $\tilde{A}$ of $(i+1)$-arcs at some later point. To apply the results of Section 10, we have to prove that conditions $(\alpha 1)-(\alpha 3)$ hold for A,

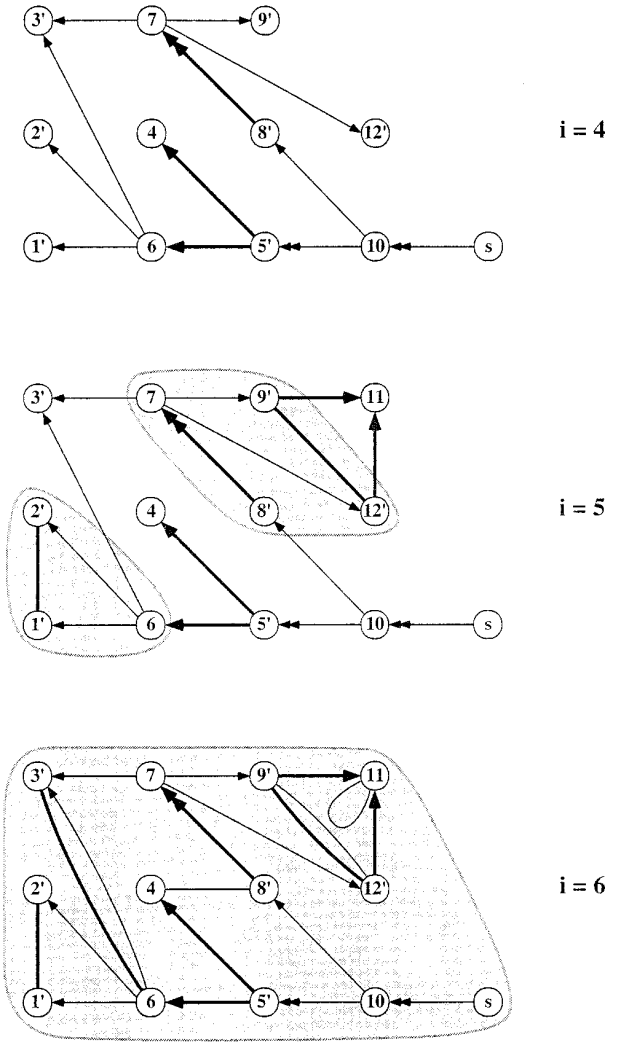

Fig. 9. Subnetworks occurring during the BNS.

$\tilde{A}$ and also for intermediate self-complementary arc sets $\hat{A}$ with $A \subseteq \hat{A} \subseteq \tilde{A}$.

Figure 9 shows some of the networks N_i belonging to our running example of Figures 2 and 6, using the notation of Section 8. For $i < 5$, these networks are less interesting, since no (i)-bridges and no proper (i)-blossoms exist. On the other hand, there are two proper (5)-blossoms, the first one consisting of $1, 2, 6$, and their complementary nodes and the second consisting of 7, 8, 9, 10 and their complements. Both (5)-blossoms which coincide with the (5)-nuclei are parts of the only (6)-blossom which additionally contains the nodes 3, 4, 5, 11, 12, s, and their complements.

In particular, a valid st-path exists in $N^6(f)$, and f can be augmented. With the exception of $(3', 4)$ and $(4', 3)$, any residual arc is a (6)-arc. By the way, the investigation of $(3', 4)$ is uninteresting since all nodes are already strictly reachable.

We now state the major result of this section which shows the relationship to the setup of Vazirani's blossoms [28]:

Theorem 12.1. *Let $d(v) < d(v')$. Then, the following assertions hold:*

$$d_i(v) = \left\{ \begin{matrix} d(v), & \text{if } d(v) \le i \\ \infty, & \textit{otherwise} \end{matrix} \right\}$$

$$d_i(v') = \left\{ \begin{matrix} d(v'), & \textit{if } t(v) \le 2i - 1 \\ \infty, & \textit{otherwise} \end{matrix} \right\}$$

Proof. For the time being, we only consider the case $i = 0$: There are no (0)-arcs. Thus, the minlevel node s is the only strictly (0)-reachable node. On the other hand, no nodes v with $t(v) < 3$ and no nodes $v \neq s$ with $d(v) = 0$ exist. ■

It will take a considerable effort to prove Theorem 12.1 in general. Meanwhile, we assume that Theorem 12.1 is proven for some fixed $j \in N_0$ and all integers $i \le j$. All results that we state in the following depend on Theorem 12.1 and, hence, are valid only for $i \le j$ until we can do the induction step. First, we have

Corollary 12.2.

$$\mathcal{B}_i = \{v \in V(N) : t(v) > 2i - 1, d(v) \le i < d(v')\}$$

$$\mathcal{C}_i = \{v \in V(N) : t(v) \le 2i - 1\}$$

$$\mathcal{D}_i = \{v \in V(N) : t(v) > 2i - 1, \; i < d(v), i < d(v')\}.$$

Corollary 12.3. *Let v be any node with $d(v) \le i$. Then, $d_i(v) = d(v)$ holds.*

Proof. If v is a minlevel node, the assertion is an immediate consequence of Theorem 12.1. Otherwise, we have $d(v') < d(v)$ and, hence, $t(v) \le 2i - 1$; again, the assertion follows from Theorem 12.1. ■

There is still some ambiguity concerning the props of the network N_i. Again, let A denote the set of (i)-arcs of the balanced network N. We now can prove that the (A)-props, the props in $N[A]$, are the (i)-props (which our notation already has suggested):

Lemma 12.4. *An (i)-arc $u'v$ is an (i)-prop iff u' is a predecessor of v on some $d_i(v)$-path and $d_i(v) < d_i(v')$.*

Proof. $(\rightarrow)$ Let $u'v$ be an (i)-prop. Then, $u'v$ is a prop and $d(v) \le i$. By definition, v is a minlevel node and $d(u') = d(v) - 1 \le i - 1$. Corollary 12.3 shows that

$$d_i(u') = d(u') < d(v) = d_i(v) < d(v') \leq d_i(v'). \tag{18}$$

Therefore, no $d_i(u')$-path traverses the co-prop $v'u$, and, thus, any such path can be extended to a $d_i(v)$-path by appending the arc $u'v$. In other words, u' is the predecessor of v on some $d_i(v)$-path.

($\leftarrow$) If $d_i(v)$ is finite, then $d_i(v) = d(v)$ holds by Theorem 12.1. Obviously, v is a minlevel node and $u'v$ is a prop. ■

Theorem 12.5. *Let uv', vu' be a pair of bridges and assume that $t(u, v') = 2i - 1$. Then, $t(u)$, $t(v) \leq t(u, v')$ hold. In particular, u and v are in a common (i)-blossom.*

Proof. As in the proof of Lemma 9.10, we may assume that $d(u) \leq d(v)$ so that v cannot occur on any $d(u)$-path p. If p contains the node v', then $d(v') < d(u)$ holds. Otherwise, $p \circ uv'$ is a valid sv'-path and $d(v') \leq d(u) + 1$. In either case, $t(v) \leq t(u, v')$ and, therefore, $v \in \mathcal{O}_i$ hold. If u is in $B_i(v)$, then $t(u) \leq 2i - 1$ holds, and nothing remains to be shown.

Therefore, we assume u not to be in $B_i(v)$ and choose some $d_i(v)$-path q. Then, $q \circ vu'$ would be (i)-valid since $B_i(v)$ is visited by q only once and reached by an (i)-prop. But then each of uv' and vu' would be i-accessible, contradicting the definition of $B_i(v)$. ■

Corollary 12.6. *The set A of (i)-arcs satisfies conditions $(\alpha 1)$–$(\alpha 3)$.*

Proof. Obviously, the set A is self-complementary. An (i)-bridge is (i)-bicursal by the proof of the preceding theorem. An (i)-prop uv' is accessed via any $d_i(v')$-path by Lemma 12.4. This proves $(\alpha 2)$. If wv' is another (i)-prop with end node v', and each of uv' and wv' is unicursal, then Lemma 12.4 proves that both arcs are (A)-props. This is $(\alpha 3)$. ■

Theorem 12.7 (Structure Theorem for Minimum Admissible Paths). *Let $v \in V(N)$ either be a minlevel node with distance $d(v) = i + 1$ or a maxlevel node with tenacity $t(v) = 2i + 1$. Any $d(v)$-path p has the following properties:*

(a) p is a $d_{i+1}(v)$-path.

(b) p traverses exactly one arc a which is not an (i)-arc.

(c) If v is a maxlevel node, then a is a bridge and $t(a) = t(v)$.

(d) p visits any (i)-nucleus at most once.

(e) Any (i)-nucleus $U \neq U(s)$ traversed by p before a is reached by $prop_i(U)$.

(f) Any (i)-nucleus U traversed by p after a is left by $prop_i(U)'$.

(g) If both xy' and $y'x$ are bridges and xy' is on p, then $t(x, y') < t(y', x)$ holds.

Proof. First, consider the case $i = 0$: Then, $d(v) = 1 < d(v')$ holds, $p = sv$ is the only $d(v)$-path, and (a)–(g) are trivially satisfied. Using induction, we can assume that the statements are proven for any $i < j$. Hence, we only consider the case $i = j$. Let p be a fixed $d(v)$-path p, and let u' be the predecessor of v on p.

First, we assume v to be a minlevel node and $d(v) = j + 1$, that is, $t(v) > 2j + 1$. Because of Lemma 8.2, $p[u']$ is a $d(u')$-path, $d(u') = j$ holds, and $u'v$ is a prop. Note that u' is either a minlevel node with $d(u') \leq j$ or a maxlevel node with $t(u') \leq 2j - 1$. Hence, the path $p[u']$ is (j)-valid by induction hypothesis (a), p is $(j + 1)$-valid, and (a), (b), and (g) are evident for the pair v, p. Since v is in no (j)-nucleus, also (d), (e), and (f) hold. Thus, the induction step is complete.

From now on we assume that v is a maxlevel node with $t(v) = 2j + 1$ and that the statements are proven for all maxlevel nodes w with $t(w) = 2j + 1$ and $d(w) < d(v)$. Since no $d(v')$-path can traverse $u'v$, we have $d(u) \leq d(v') + 1$ and, therefore, $t(u) \leq t(v)$. Clearly, $u'v$ cannot be a prop.

We first assume $u'v$ to be a bridge. Because of $d(u') < d(v)$, we have $t(u'v) \leq t(v)$. If even $t(u'v) < t(v)$ would hold, that is, $t(u'v) \leq 2j - 1$, Theorem 12.5 would yield the contradiction $t(v) \leq t(u'v) < t(v)$. Thus, we obtain $t(u'v) = t(v)$, and, therefore, $d(u') + 1 = d(v)$. In particular, $p[u']$ is a $d(u')$-path.

If $t(u)$ is strictly smaller than $t(v)$, then $p[u']$ is a $d_j(u')$-path by the induction hypothesis. Otherwise, we must have $t(u) = t(v)$ and $d(u) > d(u')$ since we have required that $v'u$ is not a prop. But then $d(u') \leq j$ holds, and $p[u']$ is a $d_j(u')$-path again. The induction step is now obvious, except for assertion (g). To see this, assume that vu' is also a bridge. Then, we have $t(u'v) \neq t(v, u')$ by Lemma 8.5. Now, Theorem 12.5 implies that $t(u'v) < t(v, u')$, since, otherwise, $t(v) \leq t(v, u') < t(u', v) = t(v)$ would hold.

Finally, we consider the case that $v'u$ is a prop. Then, $d(v') + 1 = d(u) < d(u')$ holds and $v'u$ is even a (j)-prop. If, in addition, $d(u') + 1 = d(v)$ holds, then $t(u)$ and $t(v)$ are equal and $p[u']$ is a $d(u')$-path. By the induction hypothesis, $p[u']$ is even a $d_{j+1}(u')$-path. But, then, the induction step is evident again.

Otherwise, $t(u) < t(v)$ and $d(u') + 1 < d(v)$ hold. By the induction hypothesis, any $d(u)$-path q is a $d_j(u)$-path. This path reaches the nucleus $U := U_j(u)$ by $v'u$, since, otherwise, $q \circ u'v$ would be (j)-valid. In particular,

$rescap(v', u) = 1$ holds; hence, p cannot traverse the arc $v'u = prop_j(U)$.

Let $x'y$ be that arc through which p reaches U the last time. Since $t(y) \leq 2j - 1$ holds, all $d(y)$-paths and all $d(y')$-paths are (j)-valid by induction hypothesis. By the base identity 9.13, any $d(u)$-path can be extended to a valid sy'-path by appending $p[y, u']'$ and any $d(y')$-path to a valid sx-path by appending $y'x$. Thus, the following inequalities hold:

$$d(x) - 1 \leq d(y') \leq d(u) + |p[y, u']| = d(v') + |p[y, v]| \tag{19}$$

$$d(x') + 1 \leq |p[x']| + 1 = d(v) - |p[y, v]|. \tag{20}$$

Adding these inequalities yields $t(x) \leq t(x', y) \leq t(v)$. Note that $t(x', y) = t(v)$ implies that $d(x') = |p[x']|$. Note also that $x'y$ is not a prop since it would be a (j)-prop reaching U otherwise, contradicting the base identity. Three cases arise:

- In the case $t(x) < t(v)$, x is in a (j)-nucleus different from U. If $y'x$ would be a prop, it would be a (j)-prop connecting two (j)-nuclei, a contradiction. Hence, $x'y$ is a bridge of tenacity $t(x', y) = t(v) = 2j + 1$ and $p[x']$ is a $d_j(x')$-path by the induction hypothesis. Moreover, if yx' is also a bridge, then $t(x', y) < t(y, x')$ holds.
- If $t(x) = t(v)$ and $d(x') < d(x)$ hold, that is, $d(x') \leq j$, then $p[x']$ is an $d_j(x')$-path by the induction hypothesis. But $x'y$ is not a (j)-arc, since, otherwise, $p[y]$ would be (j)-valid, contradicting the base identity. Thus, $x'y$ is a bridge of tenacity $t(x', y) = t(v)$. Again, if yx' is also a bridge, then $t(x', y) < t(y, x')$ holds [as in the case $t(u) = t(v)$ above].
- If $t(x) = t(v)$ and $d(x') > d(x)$ hold, then $p[x']$ is a $d_{j+1}(x')$-path by the induction hypothesis. Furthermore, $y'x$ is a (j)-prop since (19) gives the equality $d(x) = d(y') + 1$.

In all cases, $p[x']$ cannot traverse the nucleus U, since, otherwise, $prop_j(U)$ or $prop_j(U)'$ would be on $p[x']$ by the induction hypothesis. But, then, we can exchange $p[y, u']$ with the complementary path of a $d_j(u, y')$-path which, again, is entirely contained in U, and obtain a valid sv-path again. Because of the minimality of p, we have $|p[y, u']'| = d_j(u, y')$. By the induction hypothesis, $p[y, u']$ is (j)-valid. ■

We can now do the induction step for the proof of Theorem 12.1. Here, we prove the conditions $(\alpha 1) - (\alpha 3)$ only for a specific chain extending the set of (j)-arcs to the set of $(j + 1)$-arcs, but note that the selection of this chain is immaterial.

Lemma 12.8. *Let uv', vu' be a pair of bridges and $t(u, v') = t(v, u') = 2i + 1$. Then, u and v are strictly (i)-reachable, and $t(u, v') = t(v, u') = d_i(u) + d_i(v) + 1$ holds.*

Proof. Again, we can assume that $d(u) \leq d(v)$ and, therefore, $d(u) \leq i$. As in the proof of Theorem 12.5, we obtain $t(v) \leq t(u, v') = 2i + 1$. Suppose that $d_i(v)$ is infinite. By Corollary 12.3, we would have $d(v) \geq i + 1$ and, hence, $d(v') \leq i < d(v)$, that is, v would be a maxlevel node. Then, Theorem 12.1 shows $t(v) = 2i + 1 = t(u, v')$, and, therefore,

$$d(v') = 2i + 1 - d(v) = (d(u) + d(v) + 1) - d(v) = d(u) + 1 \tag{21}$$

hold, contradicting the assumption that uv' was a bridge. Hence, we have $d_i(u) = d(u)$ and $d_i(v) = d(v)$ by Corollary 12.3. ■

Proof (of Theorem 12.1). Let A be the set of (j)-arcs; $\{a_1, a_1', a_2, a_2', \ldots, a_k, a_k'\}$, the set of bridges a_r with tenacity $t(a_r) = 2j + 1$; and $\{a_{k+1}, a_{k+2}, \ldots, a_{k+1}\}$, the set of props $a_r = u_r'v_r$ *with* $d(v_r) = j + 1$. Also, put $A_r := A \cup \{a_1, a_1', a_2, a_2', \ldots, a_r, a_r'\}$ for $r = 1, 2, \ldots, k + l$. Then, A_{k+l} is the set of $(j + 1)$-arcs.

By Corollary 12.6, A satisfies the conditions $(\alpha 1) - (\alpha 3)$. By Lemma 12.8, both end nodes of a_r are strictly (A_r)-reachable for $r = 1, 2, \ldots, k$; hence, a_r and a_r' are (A_r)-bicursal. By the first shrinking property and induction on r, A_r satisfies conditions $(\alpha 1)$ and $(\alpha 2)$ for $r = 1, 2, \ldots, k$, and $\mathcal{D}_{A_r} = \mathcal{D}_j$, $\mathcal{B}_{A_r} \subseteq \mathcal{B}_j$ hold. Furthermore, $d(w) = d_{A_r}(w) = d_j(w)$ holds for any strictly (j)-reachable node w. Thus, every (j)-prop is a (A_r)-prop, and condition $(\alpha 3)$ holds for A_r, $r = 1, 2, \ldots, k$.

Choose some $w \in \mathcal{C}_{A_k} \backslash \mathcal{C}_j$ with $d_{A_r}(w) = \infty$, $d_{A_{r+1}}(w) < \infty$. Let p be a $d_{A_{r+1}}(w)$-path which traverses $a_{r+1} = uv'$, and q, a $d_{A_r}(w')$-path. An application of the second shrinking property 10.6 and Lemma 12.8 yield

$$t(w) \leq |p| + |q| = |p[u]| + 1 + |p[v', w]| + |q| = d_{A_r}(u) + d_{A_r}(v) + 1 = t_{A_r}(u, v') = t(u, v') = 2j + 1 \tag{22}$$

and even $t(w) = 2j + 1$ and $d(w) > d(w')$, since w is not strictly (j)-reachable. On the other hand, $d(w) = d_{A_k}(w)$ holds for any maxlevel node w with $t(w) = 2j + 1$ by Theorem 12.7. Thus, exactly the maxlevel nodes w with $t(w) = 2j + 1$ become strictly reachable during the investigation of the bridges of tenacity $2j + 1$.

If $a_r = u_r'v_r$ *is a prop, and* $k + 1 \leq r \leq k + l$, then $d(u_r') = j$ and $d(v_r) = j + 1$ hold. It is obvious that a_r

is (A_r)-accessible and an (A_r)-prop. An application of Theorem 10.2 and induction on r show that A_r satisfies $(\alpha 1)-(\alpha 3)$ for $r = k + 1, k + 2, \ldots, k + l$ and that only the node v_r becomes strictly reachable during the investigation of a_r and a_r'. Finally, $d(w) = d_{j+1}(w)$ holds for any minlevel node w with distance label $d(w) = j + 1$. ■

We already know the most important consequences of Theorem 12.1, namely, the results which prepared the induction step and which now hold for all values of i. We also conclude that the general blossoms and nuclei studied in Section 9 coincide with the (i)-blossoms and the (i)-nuclei, respectively.

Corollary 12.9. *Every blossom of the balanced network N is an (i)-blossom for some appropriate $i \in N$.*

Proof. We choose $i := \max\{d(v) : v \in V(N)\} + 1$. Then, $d(v) < i - 1$ holds for every minlevel node and $t(v) < 2i - 1$ holds for every node $v \in \mathcal{C}$. Because of Theorem 12.1, we obtain $\mathcal{C} = \mathcal{C}_i$ and $\mathcal{B} = \mathcal{B}_i$. That means that the buds of N are exactly the (i)-buds.

The only arcs of N that are not (i)-arcs are the bridges of infinite tenacity. By Theorem 9.10, these arcs cannot be contained in a nucleus. Let uv' be a bicursal arc; p, a uv'-accessing path, and $b := base(U(u, v))$. Then, $p[b, v']$ is (i)-valid by the base identity. If q is a $d_i(b)$-path, then $q \circ p[b, v']$ is (i)-valid and uv'-accessing, that is, every bicursal arc is (i)-bicursal, and the proper blossoms are exactly the proper (i)-blossoms. ■

SUMMARY

In this paper, we have presented a series of tools which appear in the setup of nearly all algorithms for the cardinality matching problem. Using the results of Section 10, one will find it easy to investigate the relationship between specific matching algorithms that depend on augmentation. By our discussion, the correctness of the known cardinality matching algorithm is easy to verify.

Our results generalize the techniques for the cardinality matching problem and the maximum-flow problem and describe algorithms for the f-factor and the degree constrained subgraph problem. As in the earlier work of Kocay and Stone, we have encoded these matching problems into a special class of flow networks and considered augmentation algorithms and the process of balanced network search (BNS).

By our discussion, the BNS splits into three parts: The first task is the management of the search strategy. The second task is determining all blossoms that have to be merged together if a bridge is investigated. The third task is merging these blossoms actually together which is a special case of the well-known disjoint set union problem (DSU).

It is somewhat surprising that the DSU is the critical part in determining the asymptotic complexity. Nevertheless, the BNS can be realized in (almost) linear time algorithms. If a depth first search strategy is chosen (as in the algorithm of Kameda and Munro [18]), one can avoid to merge blossoms explicitly, but, then, augmenting paths are missed for some pathological instances.

In forthcoming papers, we will present three such implementations of the BNS: In [10], we will present a DFS-like procedure extending the Kameda/Munro algorithm together with an example where an augmenting path is missed. We will also describe a BFS-like algorithm which is an improvement of the Kocay/Stone [20] algorithm and is similar to the Edmonds [7] search strategy. From the latter method, an $O(nm)$-time algorithm for the k-factor problem and an $O(m^2)$-time algorithm for the f-factor problem result.

Another paper [11] will concern the double depth first search procedure and the extension of the algorithm of Micali/Vazirani [23] to capacitated matching problems. This BNS method determines minimum valid paths and results in strongly polynomial time algorithms for the maximum balanced flow problem.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, Network Flows, Prentice Hall, Englewood Cliffs, NJ (1993).

[2] R. P. Anstee, An algorithmic proof of Tutte's f-factor theorem, J Alg 6 (1985), 112–131.

[3] C. Berge, Two theorems in graph theory, Proceed Nat Acad Sci 43 (1957), 842–844.

[4] W. H. Cunningham and A. B. Marsh, A primal algorithm for optimum matching, Math Program Study 8 (1978), 50–72.

[5] U. Derigs, Programming in networks and graphs, Springer, Heidelberg, 1988.

[6] J. Edmonds, Maximum matching and a polyhedron 0,1 vertices, J Res Natl Bur Stand Sect B 69 (1965), 125–130.

[7] J. Edmonds, Paths, trees and flowers, Can J Math 17 (1965), 449–467.

[8] S. Even and R. E. Tarjan, Network flow and testing graph connectivity, SIAM J Comput 4 (1975), 507–512.

[9] L. R. Ford and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, NJ, 1962.

[10] C. Fremuth-Paeger and D. Jungnickel, Balanced network flows (II): Simple augmentation algorithms, Networks, to appear.

[11] C. Fremuth-Paeger and D. Jungnickel, Balanced network flows (III): Strongly polynomial augmentation algorithms, Networks, to appear.

[12] C. Fremuth-Paeger and D. Jungnickel, Balanced network flows (IV): Duality and structure theory, Networks, submitted.

[13] C. Fremuth-Paeger and D. Jungnickel, Balanced network flows (V): Cycle canceling algorithms, Networks, submitted.

[14] A. Goldberg and A. V. Karzanov, Path problems in skew-symmetric graphs, Combinatorica 16 (1996), 353–382.

[15] P. Hell and D. G. Kirkpatrick, Algorithms for degree constrained graph factors of minimum deficiency, J Alg 14 (1993), 115–138.

[16] J. E. Hopcroft and R. M. Karp, An $n^{5/2}$ algorithm for maximum matching in bipartite graphs, SIAM J Comput 2 (1973), 225–231.

[17] D. Jungnickel, *Graphen, Netzwerke und Algorithmen,* 3rd ed. BI-Wissenschafts-verlag, Mannheim, 1994. English edition to appear in 1998 published by Springer.

[18] T. Kameda and I. Munro, An $O(|V|*|E|)$ algorithm for maximum matching of graphs, Computing 12 (1974), 91–98.

[19] W. Kocay and D. Stone, Balanced network flows, Bull ICA 7 (1993), 17–32.

[20] W. Kocay and D. Stone, An algorithm for balanced flows, JCMCC 19 (1995), 3–31.

[21] L. Lovasz and M. D. Plummer, Matching Theory, North-Holland, Amsterdam, 1986.

[22] A. B. Marsh, Matching algorithms, PhD Thesis, John Hopkins University, Baltimore, 1979.

[23] S. Micali and V. V. Vazirani, An $O(\sqrt{V}E)$ algorithm for finding maximum matching in general graphs, Proceedings of the 21st Annual IEEE Symposium in Foundation of Computer Science, 1980, pp. 17–27.

[24] U. Pape and D. Conradt, Maximales matching in graphen, Ausgewählte Operations Research Software in FORTRAN, H. Späth (Editor), Oldenbourg, Munich, 1980, 103–114.

[25] W. Pulleyblank, Faces of matching polyhedra, PhD Thesis, University of Waterloo, 1973.

[26] W. T. Tutte, The factors of graphs, Can J Math 4 (1952), 314–328.

[27] W. T. Tutte, Antisymmetrical digraphs. Can J Math 19 (1967), 1101–1117.

[28] V. V. Vazirani, A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{V}E)$ general graph maximum matching algorithm, Combinatorica 14 (1994), 71–109.