An Efficient Algorithm for Finding Maximum Cycle Packings in Reducible Flow Graphs

Xujin Chen and Wenan Zang[*][†]
Department of Mathematics
University of Hong Kong
Hong Kong, China

Abstract

Reducible flow graphs occur naturally in connection with flow-charts of computer programs and are used extensively for code optimization and global data flow analysis. In this paper we present an $O(n^2 m \log(n^2/m))$ algorithm for finding a maximum cycle packing in any weighted reducible flow graph with n vertices and m arcs; our algorithm heavily relies on Ramachandran's earlier work concerning reducible flow graphs.

Keywords. feedback set, cycle packing, network flow, algorithm, complexity.

[*]Supported in part by the Research Grants Council of Hong Kong (Project No. HKU 7054/03P).
[†]Corresponding author. E-mail: wzang@maths.hku.hk.

1 Introduction

Let G be a digraph with a nonnegative integral weight $w(e)$ (resp. $w(v)$) on each arc e (resp. vertex v). A collection $\mathcal{C}$ of distinct cycles $C_1, C_2, \ldots, C_k$ of G together with nonnegative integers $m(C_1), m(C_2), \ldots, m(C_k)$ is called a *cycle packing* with respect to w if for each arc e (resp. vertex v) of G, the sum $\sum_{e \in C_i} m(C_i) \leq w(e)$ (resp. $\sum_{v \in C_i} m(C_i) \leq w(v)$); a set X of arcs (resp. vertices) in G is called a *feedback arc (resp. vertex) set* if X intersects every cycle in G. The *cycle packing problem* is to find a cycle packing $\mathcal{C}$ with maximum weight $\sum_{C_i \in \mathcal{C}} m(C_i)$, and the *feedback arc (resp. vertex) set problem* consists in finding a feedback arc (resp. vertex) set X with minimum weight $\sum_{x \in X} w(x)$. These two problems form a primal-dual pair in integer programming and thus are closely tied to each other. While the latter is a well-known NP-hard problem [12] and has been studied extensively, the present paper concerns itself with the former, which also arises in a variety of applications. Recently, Caprara, Panconesi, and Rizzi [7] gave a thorough and detailed analysis of the hardness and approximablity of the cycle packing problem on unweighted undirected graphs. We point out that a slight modification of their approaches can lead to essentially the same statements for the problem on unweighted digraphs; that is, it admits no fully polynomial time approximation scheme unless $P = NP$, and can be approximated within a factor of $\frac{1}{2 \log n}$, where n is the number of vertices in the input digraph and the base of log is 2. In this paper we focus our study of the cycle packing problem on reducible flow graphs.

Reducible flow graphs (or simply reducible graphs) occur naturally in connection with flow-charts of computer programs and are used extensively for code optimization and global data flow analysis, so they have attracted tremendous research efforts in the past three decades. Hopcroft and Ullman [17] obtained the first efficient algorithm for recognizing reducible flow graphs, which was improved by Tarjan [24]. The reader is referred to Hecht and Ullman [15] for various good characterizations of all reducible flow graphs. In [22], Shamir gave a linear time algorithm for finding minimum feedback vertex sets in reducible graphs, and an $O(n^2 m \log(n^2/m))$ algorithm was discovered by Ramachandran [19] for finding a minimum weighted feedback arc set in an arc-weighted reducible graph and for finding a minimum weighted feedback vertex set in a vertex-weighted reducible graph with n vertices and m arcs. In [20], Ramachandran managed to prove that the cardinality of a minimum feedback arc set in a reducible flow graph is equal to the cardinality of a maximum collection of arc disjoint cycles, thereby establishing a conjecture of Frank and Gyarfas [11]; her proof also contains an $O(\min\{mn^{5/3}, m^2\})$ algorithm for finding the corresponding set of arc disjoint cycles. We remark that, first, although Ramachandran's proof [20] is concerning the unweighted case, it yields the corresponding minimax theorem in the weighted case; that is, in any arc-weighted reducible flow graph, the maximum weight of a cycle packing is equal to the minimum weight of a feedback arc set. Moreover, this minimax relation implies the corresponding theorem for the vertex-weighted case (the unweighted version is due to Frank and Gyafas [11]). Second, Ramachandran's proof [20], algorithmic in nature, can be further extended to find maximum cycle packings in weighted reducible flow graphs. One subroutine of this approach, called $O(n)$ times, is a maximum flow algorithm, in which the so-called newly added arcs must be saturated by the flows at each step, so the augmenting path method for the maximum flow has to be applied; this leads to a gap between the time complexity of this algorithm and that of Ramachandran's algorithm [19] for the feedback set problem on weighted reducible flow graphs as, to the best of our knowledge, none of the most efficient maximum flow algorithms currently known for general networks is based on the augmenting path method directly. The purpose of this paper is to bridge this complexity gap and present an $O(n^2 m \log(n^2/m))$ algorithm for finding a maximum cycle packing in any weighted reducible flow graph. Our algorithm heavily relies on Ramachandran's earlier work [19, 20] and thoroughly exploits the laminar structure of reducible flow graphs. Moreover, it can use any fastest (integral) maximum flow algorithm as its subroutine.

The remainder of this paper is organized as follows. In section 2, we give some preliminary results on reducible flow graphs and network flows. In section 3, we describe Ramachandran's algorithms and results, present our algorithm, and establish its correctness. In section 4, we conclude this paper with some remarks and open problems.

2 Preliminaries

Let us now introduce some notions and terminologies. Let $G = (V, A)$ be a digraph. We denote by (u, v) an arc in A from its *tail* u to its *head* v. A *walk* in G is a finite sequence $W = v_0 e_1 v_1 \dots e_k v_k$, whose terms are alternately vertices and arcs, such that $e_i = (v_{i-1}, v_i)$ for $1 \le i \le k$. If $v_0, v_1, \dots, v_k$ are distinct, then W is called a *path from* v_0 *to* v_k or a $v_0 - v_k$ *path*; if $v_0, v_1, \dots, v_{k-1}$ are distinct, $v_k = v_0$, and $k \ge 2$, then W is called a *cycle*. Graph G is called *acyclic* if it contains no cycles. For convenience, we let $P[u, v]$ denote the section of a path P from u to v.

A *rooted directed graph* or *a flow graph* is a digraph G with a distinguished vertex r, called its *root*, such that there is a path in G from r to every vertex in G. Let $G = (V, A, r)$ be a flow graph, and let $u, v \in V$. We say that u *dominates* v (or u is a *dominator* of v) if every $r - v$ path in G passes through u. Note that u dominates itself. Let R be a subgraph of G. A vertex v in R is called an *entry vertex* of R if $v = r$ or if there is an arc (u, v) of G with u outside

R. A *DAG* of G is a maximal acyclic subgraph of G rooted at r. A *depth first search DAG* of G is a DAG containing a directed spanning tree T with root r grown by the depth first search (DFS) algorithm [23, 25]; it can be seen from the definition that this DAG is obtained from T by adding a maximal subset of arcs in G so that no cycle is created. Graph G is called *reducible* if the depth first search DAG of G is unique. Unless G is acyclic, DFS also discovers *back arcs*, which are arcs of G not included in the depth first search DAG.

The following characterizations of reducible flow graphs will be used repeatedly in this paper.

Theorem 2.1. *Let $G = (V, A, r)$ be a flow graph, and let $D = (V, A_D, r)$ be an arbitrary depth first search DAG of G with back arc set $B = A \backslash A_D$. Then the following statements are equivalent.*

(i) G is reducible;

(ii) G contains no F as a subgraph (see FIG. 1*);*

(iii) D is the unique DAG of G;

(iv) The arc set A of G can be partitioned into two sets A_1 and A_2 such that (V, A_1, r) is a DAG of G and u dominates v in G for each (v, u) in A_2;

(v) Every cycle C of G contains exactly one back arc $e \in B$. Moreover, the head of e is an entry vertex of C which dominates any other vertex on C.

Proof. The equivalence of (i)-(iv) can be found in [15, 16]. The implication (i)$\Rightarrow$(v) is contained in [22], and the converse (v)$\Rightarrow$(i) follows from (ii) and the fact that no vertex of the cycle in F dominates all other vertices on this cycle. $\square$

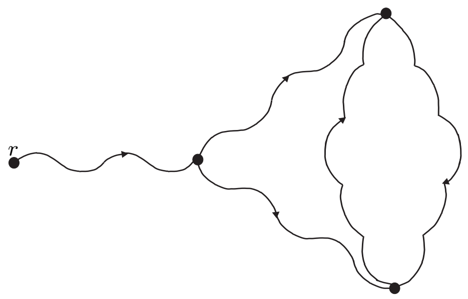


FIG. 1. *The Forbidden Subgraph F.*

Let $G = (V, A, r)$ be a reducible flow graph and let u, v be two vertices in G. Observe that

$$\text{if } u \text{ dominates } v \text{ then there is a DAG path from } u \text{ to } v. \tag{2.1}$$

(Indeed, let P be a path from r to v in the depth first search DAG. Then P passes through u since u dominates v. Hence $P[u,v]$ is as desired.) As shown by Aho and Ullman [3], the dominator relation on G can be represented in the form of a tree rooted at r. Let V_h denote the set of heads of all back arcs in G. The *head dominator tree* T_h of G, introduced by Ramachandran [19, 20], represents the domination relation restricted to $V_h \cup \{r\}$: the descendants of a vertex u in T_h are precisely the vertices in V_h dominated by u in G. For each $u \in V_h \cup \{r\}$, let $(u_1, v_1), (u_2, v_2), \ldots, (u_k, v_k)$ be all the back arcs in G whose heads are dominated by u. The set $V_u = \{u' \in V\colon\ u'$ lies on a DAG path from u to some u_i, for $1 \leq i \leq k\}$ is called the *dominated back arc vertex set of* u, and the subgraph of G induced by V_u is denoted by $G_s(V_u)$ in Ramachandran [19, 20] (see FIG. 2).

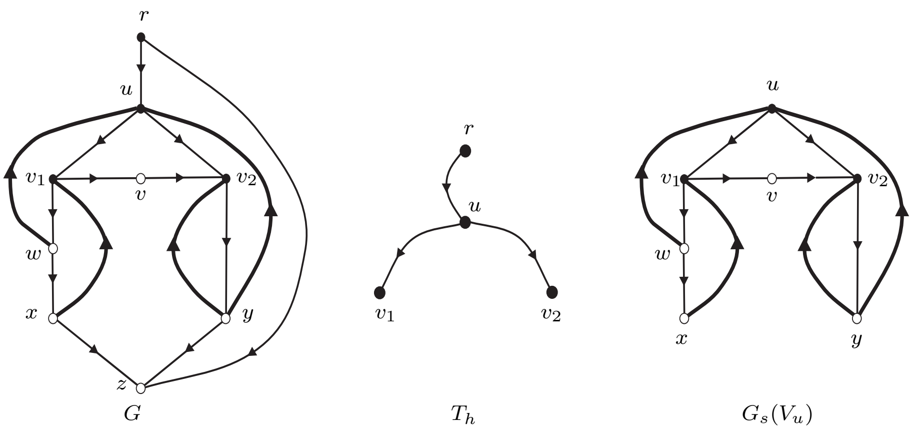


FIG. 2. *The Head Dominator Tree T_h and $G_s(V_u)$.*

Let us now exhibit some properties enjoyed by the dominated back arc vertex sets and the subgraphs induced by them.

Lemma 2.2. *Let $G = (V, A, r)$ be a reducible flow graph and let u, v be two vertices in $V_h \cup \{r\}$. Then the following statements hold.*

(i) u dominates all vertices in V_u;

(ii) $G_s(V_u)$ is a reducible flow graph rooted at u;

(iii) u is the unique entry vertex of $G_s(V_u)$ in G;

(iv) If $G_s(V_u)$ and $G_s(V_v)$ have a common vertex, then either u dominates v or v dominates u.

Proof. (i) This observation is due to Ramachandran [19, 20].

(ii) By the definition of V_u, $G_s(V_u)$ is a flow graph rooted at u. Suppose, on the contrary, that $G_s(V_u)$ is not a reducible flow graph rooted at u. Then Theorem 2.1(ii) guarantees the existence of an F (see Fig. 1) rooted at u in $G_s(V_u)$; denote this F by H. Let P be a DAG path from r to u. By (i), u is the only common vertex of P and $G_s(V_u)$. Thus $P \cup H$ is an F rooted at r in G, contradicting Theorem 2.1(ii).

(iii) Suppose to the contrary that v is an entry vertex of $G_s(V_u)$ with $u \neq v$. Then $v \neq r$ since, by (i), u dominates v. So there exists an arc (w, v) of G with $w \notin V_u$. By (i) and the definition of V_u, (w, v) is not a back arc. Now let P be a DAG path from r to w. Observe that v is not on P (for otherwise $P[v, w] \cup \{(w, v)\}$ would be a cycle contained in the DAG, a contradiction), and that P passes through u (for otherwise $P \cup \{(w, v)\}$ would be an $r - v$ path that avoids u, but u dominates v by (i), a contradiction). By the definition of V_u, there exists a back arc (u_i, v_i) such that u dominates v_i and v is on a DAG path Q from u to u_i. Since the DAG of G is acyclic, the DAG paths $P[u, w] \cup \{(w, v)\}$ and $Q[v, u_i]$ have no vertex in common except v. Thus $P[u, w] \cup \{(w, v)\} \cup Q[v, u_i]$ is a DAG path from u to u_i that passes through w. Hence $w \in V_u$, this contradiction implies the uniqueness of the entry vertex of $G_s(V_u)$.

(iv) Suppose to the contrary that there is no domination relation between u and v. Then by (i) we have $u \notin G_s(V_v)$ and $v \notin G_s(V_u)$. Let $w \in G_s(V_u) \cap G_s(V_v)$. By (ii), there is a $u - w$ path P contained in $G_s(V_u)$. From P we deduce that $G_s(V_v)$ has an entry vertex different from v, contradicting (iii). □

Lemma 2.3. *Let $G = (V, A, r)$ be a reducible flow graph. Then there is an $O(n \log n + m)$ algorithm for finding a DFS order π of the head dominator tree T_h such that, for any two vertices u and v on T_h with $\pi(u) < \pi(v)$, vertex u is not dominated by v, and every path from v to u in G contains a back arc, where $n = |V|$ and $m = |A|$.*

Proof. Let D be the DAG of G. Since D is acyclic, there is a linear time algorithm for finding an ordering σ of all the vertices of D such that if (a, b) is an arc of D then $\sigma(a) < \sigma(b)$ (see topological sorting in [1]). Now let π be the DFS order of T_h starting from r such that, for any two children a, b of the same vertex on T_h, $\pi(a) < \pi(b)$ iff $\sigma(a) < \sigma(b)$ (that is, among the unscanned arcs leaving each vertex u on T_h, we always search (u, v) with the smallest order $\sigma(v)$ first). Then π is as desired. To justify it, let u and v be two vertices on T_h with $\pi(u) < \pi(v)$. If u is dominated by v, then there is a path from v to u in T_h. Since π is a DFS order of T_h starting from r, we have $\pi(v) < \pi(u)$, a contradiction. Thus u is not dominated by v. It remains to show that there is no path from v to u in D.

Suppose to the contrary that D contains a $v - u$ path P. If u dominates v, then P together with a path from u to v in D (recall (2.1)) would lead to a cycle in D, a contradiction. Let us consider the case when v is not dominated by u. Now we have $r \notin \{u, v\}$. Let w be the nearest common ancestor of u and v on T_h, and let x, y be the two children of w on T_h such that $u \in G_s(V_x)$ and $v \in G_s(V_y)$. By Lemma 2.2(iv), V_x and V_y are vertex disjoint. It can also be seen from Lemma 2.2(iii) that P passes through x. In view of (2.1), there is a path Q from y to v in D. So $P \cup Q$ contains a path from y to x in D. Hence $\sigma(y) < \sigma(x)$. It follows that the DFS specified in our algorithm would yield $\pi(v) < \pi(u)$, a contradiction.

Since sorting k numbers can be done in time $O(k \log k)$, the time complexity of our algorithm is $O(n \log n + m)$. □

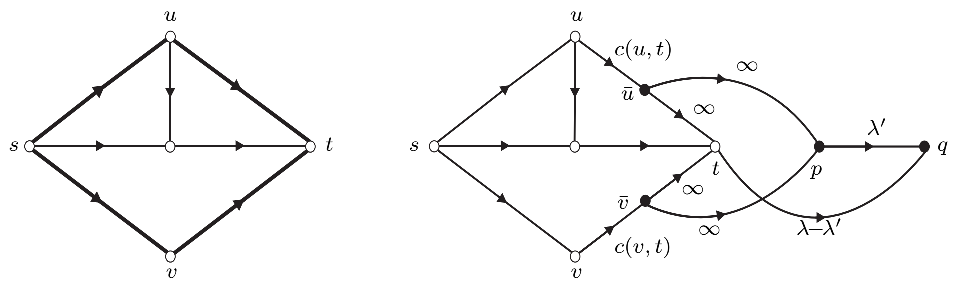
To solve the (weighted) feedback vertex set problem and the (unweighted) maximum cycle packing problem on reducible flow graphs, Ramachandran [19, 20] invented several sophisticated network flow techniques. In this paper we shall apply her novel ideas and develop her methods. As usual, a *flow network* $N = (V, A, s, t, c)$ is a digraph $G = (V, A)$ with two distinguished vertices, a *source* s and a *sink* t, and a nonnegative integral *capacity* $c(u, v)$ on each arc (u, v). See Ahuja, Magnanti and Orlin [1] and Ford and Fulkerson [10] for in-depth accounts of network flow theory. It is clear that any subgraph (V', A') of G with $\{s, t\} \subseteq V'$ corresponds to a *subnetwork* $N' = (V', A', s, t, c)$ of N. For convenience, we let $|x|$ denote the value of a flow x.

Lemma 2.4. *Let $N' = (V', A', s, t, c)$ be a subnetwork of a flow network $N = (V, A, s, t, c)$, and let μ and μ' be the maximum $s - t$ flow values of N and N', respectively. Suppose λ and λ' are two nonnegative integers satisfying $\lambda' \le \mu'$ and $\lambda' \le \lambda \le \mu$. Then there is an $O(nm \log(n^2/m))$ algorithm for finding an $s - t$ flow x of N such that $|x| = \lambda \ge \sum_{(u,t) \in A'} x(u, t) \ge \lambda'$, where $n = |V|$ and $m = |A|$.*

Proof. Let us construct a network $\bar{N} = (\bar{V}, \bar{A}, s, q, c)$ from N as follows: first add two new vertices p and q and two new arcs (p, q) and (t, q) with capacities $c(p, q) = \lambda'$ and $c(t, q) = \lambda - \lambda'$, then subdivide each arc (u, t) in A' into two arcs $(u, \bar{u})$, $(\bar{u}, t)$ and add one arc $(\bar{u}, p)$ such that $c(u, \bar{u}) = c(u, t)$ and $c(\bar{u}, t) = \infty = c(\bar{u}, p)$. The construction of $\bar{N}$ is illustrated in Fig. 3, where the arcs in A' are bold lined. Notice that in $\bar{N}$, vertex s remains to be the source while q becomes the sink. We claim that the maximum $s - q$ flow value of $\bar{N}$ is λ.

To justify the claim, by the max-flow min-cut theorem we may turn to verify that the capacity of a minimum $s - q$ cut of $\bar{N}$ is λ. Since $[\bar{V} \backslash \{q\}, \{q\}]$ is clearly an $s - q$ cut of $\bar{N}$ with capacity λ, it suffices to show that any $s - q$ cut $[U, \bar{U}]$ in $\bar{N}$ is of capacity at least λ, where $s \in U$ and $q \in \bar{U}$. This is true if $\{(p, q), (t, q)\} \subseteq [U, \bar{U}]$. So we may assume that $\{p, t\} \cap \bar{U} \ne \emptyset$ and that $\bar{U}$ contains all $\bar{u}$ for all $(u, t) \in A'$ (for otherwise $[U, \bar{U}]$ is of capacity ∞). If $t \in \bar{U}$, then $[U \backslash \{p\}, V \backslash (U \backslash \{p\})]$, an $s - t$ cut of N, has capacity at least μ by the max-flow min-cut theorem, therefore the capacity of $[U, \bar{U}]$ is at least μ ($\ge \lambda$) by the definition of $\bar{N}$. It remains to consider the case when $t \in U$. Now we have $p \in \bar{U}$ by the previous assumption. Observe that $[U, \bar{U}]$

contains the arc (t,q) with capacity $c(t,q) = \lambda - \lambda'$, and that the capacity of $[U,\bar{U}]\backslash\{(t,q)\}$ is at least that of $[(U \cap V')\backslash\{t\}, (V'\backslash U) \cup \{t\}]$, which is an $s-t$ cut in N' with capacity at least μ' $(\geq \lambda')$. Thus the capacity of $[U,\bar{U}]$ is at least $\lambda - \lambda' + \mu' \geq \lambda$, as claimed.



$N=(V,A,s,t,c)$ and $N'=(V',A',s,t,c)$ $\qquad$ $\bar{N}=(\bar{V},\bar{A},s,q,c)$

FIG. 3. *The Construction of $\bar{N}=(\bar{V},\bar{A},s,q,c)$ from N and N'.*

Now let $\bar{x}$ be a maximum $s-q$ flow in $\bar{N}$. Clearly, $\bar{x}(p,q) = \lambda'$ and $\bar{x}(t,q) = \lambda - \lambda'$. For each $e \in A$, set $x(e) = \bar{x}(u,\bar{u})$ if $e = (u,t) \in A'$ for some u, and set $x(e) = \bar{x}(e)$ otherwise. It follows from flow conservation and the construction of $\bar{N}$ that x is an $s-t$ flow of N and

$$\begin{aligned}
\lambda &= |\bar{x}| = \bar{x}(p,q) + \bar{x}(t,q) \\
&= \sum_{u:(u,t)\in A'} \bar{x}(u,\bar{u}) + \sum_{(w,t)\in A\backslash A'} \bar{x}(w,t) \\
&= \sum_{(w,t)\in A} x(w,t) = |x| \geq \sum_{(u,t)\in A'} x(u,t) = \sum_{u:(u,t)\in A'} \bar{x}(u,\bar{u}) \geq \bar{x}(p,q) = \lambda'.
\end{aligned}$$

Thus x is as desired. Since a maximum flow in $\bar{N}$ can be found [13] in time $O(|\bar{V}||\bar{A}|\log(|\bar{V}|^2/|\bar{A}|))$ $= O(nm\log(n^2/m))$, we are done. □

To establish the correctness of our algorithm, we need to decompose a flow in a network into flows on paths. So the following theorem (see [1]) will be applied.

Theorem 2.5 (Flow Decomposition Theorem). *Let x be an integral $s-t$ flow with value k in an acyclic network $N=(V,A,s,t,c)$. Then there is an $O(nm)$ algorithm for finding $s-t$ paths $P_1, P_2, \ldots, P_\ell$ and corresponding nonnegative integers $y_1, y_2, \ldots, y_\ell$, such that*

- $\ell \leq m$,
- $\sum_{i=1}^{\ell} y_i = k$, *and*
- $\sum_{i:\,(u,v)\in P_i} y_i = x(u,v)$ *for any* $(u,v) \in A$,

where $n = |V|$ and $m = |A|$. □

For simplicity, we denote the above *path flow decomposition* of x by $\{y_1P_1, y_2P_2, \ldots, y_\ell P_\ell\}$.

3 Algorithm

Let $G = (V, A, r)$ be a reducible flow graph with a nonnegative integral weight $w(e)$ on each arc $e \in A$, and let V_h, T_h, and $G_s(V_u)$ be the same as given in the preceding section. In [19, 20], Ramachandran solved the (weighted) feedback set problem and the (unweighted) cycle packing problem on reducible flow graphs by using clever network flow techniques, and the following networks played important roles in her algorithms.

For each $u \in V_h \cup \{r\}$, the *maximum flow network of G with respect to head u* is a flow network $G_m(u)$ obtained from $G_s(V_u)$ by first splitting each head v in $G_s(V_u) \cap V_h$ into two vertices v and v' and then adding a new vertex t. Each DAG arc entering (resp. leaving) the original head v in G corresponds to one that enters (resp. leaves) the newly formed head v, and each back arc entering the original v corresponds to one that enters v' in $G_m(u)$. Moreover, there is an arc (v', t) with infinity capacity. The capacity of any other arc in $G_m(u)$ is equal to its corresponding weight in G. In $G_m(u)$, vertices u and t are interpreted as the source and sink, respectively. We propose to call v' the *image* of v.

For each $u \in V_h \cup \{r\}$, the *mincost maximum flow network of G with respect to head u* is a flow network $G_{mm}(u)$ obtained inductively as follows: If u is a vertex with no child in T_h, set $G_{mm}(u) = G_m(u)$; else, let $v_1, v_2, \ldots, v_k$ be all the children of u on T_h, and let the maximum flow value of $G_{mm}(v_i)$ be c_i for $1 \leq i \leq k$. Then the vertex set of $G_{mm}(u)$ is the same as that of $G_m(u)$, and the arc set of $G_{mm}(u)$ consists of all arcs in $G_m(u)$, $G_{mm}(v_i)$ for $i = 1, 2, \ldots, k$, and $F_u = \{(u, v_i) : i = 1, 2, \ldots, k\}$. The capacity of each arc (u, v_i) in F_u is set to be c_i. Set F_u is called the *mincost-arc set for head u*.

For $G = (V, A, r)$ in Fig. 2, the construction of $G_m(u)$ and $G_{mm}(u)$ is illustrated in Fig. 3. Now we are ready to present Ramachandran's algorithms.

Ramachandran's Algorithm for finding a minimum (weighted) feedback arc set [19]

Input: An arc-weighted reducible flow graph $G = (V, A, r)$.
Output: A weight of a minimum feedback vertex set of G.

```
begin
1. Preprocess $G$: Label the heads of back arcs in $G$ in postorder (see [25]). Derive the head dominator tree $T_h$ for $G$. Introduce a pointer from each vertex $i$ in $T_h$ (except $r$) to its parent $h_i$. Let the number of vertices in $T_h$ be $h$.
2. For $i = 1, 2, \ldots, h$ process vertex $i$:
   a. Find the capacity of minimum cut, $c_i$, in $G_m(i)$.
   b. If $i \neq h$ then introduce an arc of weight $c_i$ from $h_i$ to $i$ in $G$. (Note that $G$ changes during
```

the execution of the algorithm so that $G_m(i)$ is the same as $G_{mm}(i)$ if G were unchanged.)
3. Output c_h as the weight of a minimum feedback set of G.
end

The arc introduced in step 2b from h_i to i is called a *newly added arc.*

Ramachandran's Algorithm for finding a maximum (unweighted) cycle packing [20]
Input: A reducible flow graph $G = (V, A, r)$.
Output: A maximum collection of arc disjoint cycles in G.
Description: Set the weight of each arc equal to 1. In step 2a of the above algorithm, let the children of head i in T_h be $i_1, i_2, \ldots, i_k$. Let $\mathbf{F}_{i_1}, \mathbf{F}_{i_2}, \ldots, \mathbf{F}_{i_k}$ be the maximum flow on $G_m(i_1), G_m(i_2), \ldots, G_m(i_k)$, respectively, found by the algorithm. During the ith execution of step 2, extend these flows to a feasible flow $\mathbf{F}$ for $G_m(i)$ by saturating the newly added arcs from i to i_j, for $j = 1, 2, \ldots, k$. Then find a maximum flow $\mathbf{F}_i$ by successively finding an augmenting path from i to t with no newly added arcs, and pushing the flow along it until no such an augmenting path can be found. (At this point a maximum flow and a minimum cut for $G_m(i)$ are obtained.) The maximum flow for $G_m(h)$ corresponds to a maximum collection of arc disjoint cycles in G with cardinality c_h.

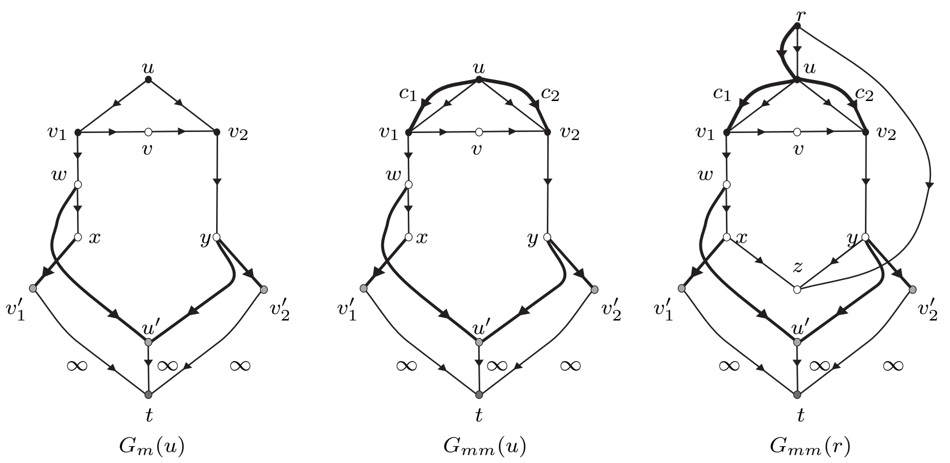


FIG. 4. *The Construction of* $G_m(u)$ *and* $G_{mm}(u)$.

The results established by Ramachandran are summarized in the following theorems.

Theorem 3.1 [19]. *In any arc-weighted reducible flow graph* $G = (V, A, r)$*, a minimum (weighted) feedback arc set can be found in time* $O(n^2 m \log(n^2/m))$*, where* $n = |V|$ *and* $m = |A|$.

Theorem 3.2 [20]. *In any reducible flow graph* $G = (V, A, r)$*, the cardinality of a minimum feedback arc set in a reducible flow graph is equal to the cardinality of a maximum collection of* *arc disjoint cycles. Moreover, the corresponding set of arc disjoint cycles can be found in time* $O(\min\{mn^{5/3}, m^2\})$*, where* $n = |V|$ *and* $m = |A|$*.*

As remarked in Section 1, Ramachandran's proof [20] not only yields the corresponding minimax theorem in the weighted case but also can be further extended to find maximum (weighted) cycle packings in weighted reducible graphs. The purpose of this paper is to present a faster algorithm for the maximum (weighted) cycle packing problem than this direct extension. Our algorithm is built heavily on Ramachandran's work [19, 20], and can use any maximum flow algorithm as its subroutine. Recall that in Ramachandran's algorithm [20] the maximum flow must saturate all newly added arcs; this requirement will be relaxed in our algorithm, and it is this relaxation that leads to improved efficiency.

Theorem 3.3. *In any arc-weighted reducible flow graph* $G = (V, A, r)$*, a maximum (weighted) cycle packing can be found in time* $O(n^2 m \log(n^2/m))$*, where* $n = |V|$ *and* $m = |A|$*.*

We break the proof of Theorem 3.3 into a series of lemmas.

Lemma 3.4. *Let* π *be the DFS search order as described in Lemma 2.3 and let* a *and* b *be two vertices in* V_h *with* $\pi(a) < \pi(b)$*. Then*

(i) a' *(the image of* a*) is not contained in* $G_{mm}(b)$*, and* b' *is contained in* $G_{mm}(a)$ *iff* a *dominates* b*;*

(ii) There is no path from b *to* a *in* $G_{mm}(r)$*;*

(iii) Either $G_{mm}(a)\backslash\{t\}$ *and* $G_{mm}(b)\backslash\{t\}$ *are vertex disjoint or* $G_{mm}(b)$ *is contained in* $G_{mm}(a)$*;*

(iv) $G_{mm}(a)$ *is an acyclic digraph rooted at* a*.*

Proof. Since $\pi(a) < \pi(b)$, by Lemma 2.3, a is not dominated by b, and hence by Lemma 2.2(i), $a \notin V_b$. So a' (the image of a) is not contained in $G_{mm}(b)$. From the construction of $G_{mm}(a)$ we see that b' is contained in $G_{mm}(a)$ iff $b \in V_a$ iff a dominates b by Lemma 2.2(i) and the definition of V_a. So (i) follows.

Suppose to the contrary that $G_{mm}(r)$ contains a path P from b to a. In view of the construction of $G_{mm}(r)$, none of t and v' (the image of v) for all v in G is contained in P. Thus P only contains DAG arcs and newly added arcs. Let us now replace each newly added arc (u, v) on P with a DAG path from u to v (see (2.1)). Then the resulting path is a DAG path from b to a, contradicting Lemma 2.3. Thus we have (ii).

Suppose $G_{mm}(a)\backslash\{t\}$ and $G_{mm}(b)\backslash\{t\}$ have a vertex in common. From the construction we deduce that $G_s(V_a)$ and $G_s(V_b)$ have a vertex in common. By Lemma 2.2(iv), a dominates

b (recall that a is not dominated by b) and thus by definition $V_b \subseteq V_a$. It follows from the construction that $G_{mm}(b)$ is contained in $G_{mm}(a)$. Hence (iii) holds.

By Lemma 2.2(ii), $G_s(V_a)$ is a reducible graph rooted at a. Let D be the DAG of $G_s(V_a)$ and let D' be the digraph obtained from D by adding all newly added arcs in $G_{mm}(a)$. It follows from Lemma 2.2(i) and (2.1) that D' is acyclic. Suppose $G_{mm}(a)$ contains a cycle C. Then the construction of $G_{mm}(a)$ implies that none of t and v' (the image of v) for all v in V_a is contained in C. Thus C is entirely contained in D', contradicting the fact that D' is acyclic. By (2.1), D is a digraph rooted at a, so is $G_{mm}(a)$ in view of the construction. Thus (iv) is established. □

Recall the construction of $G_m(u)$ and $G_{mm}(u)$ and the definition of the mincost-arc set F_u for head u, it is easy to see that $G_{mm}(u)$ is obtained from $G_m(u)$ by adding all arcs in F_v for all vertices v in $G_s(V_u) \cap (V_h \cup \{r\})$. The arcs in $G_{mm}(u)$ but not in $G_m(u)$ are precisely the newly added arcs as defined above. Observe that the set of all the newly added arcs in $G_{mm}(u)$ forms the subtree of T_h rooted at u.

An $r-t$ flow x in $G_{mm}(r)$ is called *good* if $x(u,v) \leq \sum_{a' \in G_{mm}(v)} x(a',t)$ holds for any newly added arc (u,v) of $G_{mm}(r)$. For example, for $G_{mm}(r)$ in Fig. 4, an $r-t$ flow x is good if $x(r,u) \leq x(u',t) + x(v'_1,t) + x(v'_2,t)$, $x(u,v_1) \leq x(v'_1,t)$, and $x(u,v_2) \leq x(v'_2,t)$. Our objective is to prove that a good integral maximum $r-t$ flow of $G_{mm}(r)$ corresponds to a maximum (weighted) cycle packing of G.

Lemma 3.5. *There is an $O(n^2 m \log(n^2/m))$ algorithm for finding a good integral maximum $r-t$ flow x in $G_{mm}(r)$, where $n = |V|$ and $m = |A|$.*

Proof. Let x be an arbitrary integral maximum $r-t$ flow in $G_{mm}(r)$. We may assume that x is not good and so

(1) some newly added arc (p,q) of $G_{mm}(r)$ satisfies

$$x(p,q) > \sum_{a' \in G_{mm}(q)} x(a',t).$$

Let π be the DFS order of T_h exhibited in Lemma 2.3, and let (u,v) be a newly added arc among all those (p,q) described in (1) such that

(2) $\pi(v)$ is minimized, that is, $\pi(v) \leq \pi(q)$ for any above-mentioned arc (p,q).

By Lemma 3.4(iv), $G_{mm}(r)$ is acyclic, so, by Theorem 2.5, x admits a path flow decomposition $\{y_1P_1, y_2P_2, \ldots, y_\ell P_\ell\}$. Without loss of generality, we assume that $P_1, P_2, \ldots, P_\alpha$ are all the paths in this decomposition that pass through v. Using Theorem 2.5, we get

(3) $\sum_{j=1}^{\alpha} y_j \geq x(u,v)$ for the above newly added arc (u,v).

Set $Q_j = P_j[v,t]$ for $1 \le j \le \alpha$. By Lemma 2.2(iii) and the construction of $G_{mm}(v)$, v is the only entry vertex of $G_{mm}(v)\backslash\{t\}$. It follows from Theorem 2.5 and once again the construction of $G_{mm}(v)$ that

(4) for each arc (a,b) of $G_{mm}(v)$, we have $x(a,b) = \sum_{j=1:\,(a,b)\in Q_j}^{\alpha} y_j$.

We extract from $G_{mm}(r)$ a flow network $G'_{mm}(v)$ with source v, sink t, and capacity function c' as follows: $G'_{mm}(v)$ is the union of $G_{mm}(v)$ and all Q_j for $j = 1, 2, \ldots, \alpha$, and for each arc (a,b) of $G'_{mm}(v)$ its capacity

(5) $c'(a,b)$ is set to be $c(a,b)$ if (a,b) is an arc in $G_{mm}(v)$ and to be $\sum_{j=1:\,(a,b)\in Q_j}^{\alpha} y_j$ if (a,b) is outside $G_{mm}(v)$.

FIG. 5 illustrates the construction of $G'_{mm}(v_1)$ for the network depicted in FIG. 4. Suppose $P_1 = ruv_1wxv'_1t$, $P_2 = ruv_1wu't$, $P_3 = ruv_1vv_2yv'_2t$, and $P_4 = ruv_2yu't$ are all the paths in the flow decomposition. Then P_1, P_2, P_3 are all the paths through v_1, so $Q_1 = v_1wxv'_1t$, $Q_2 = v_1wu't$, and $Q_3 = v_1vv_2yv'_2t$. Thus $G'_{mm}(v_1)$ is as shown below.

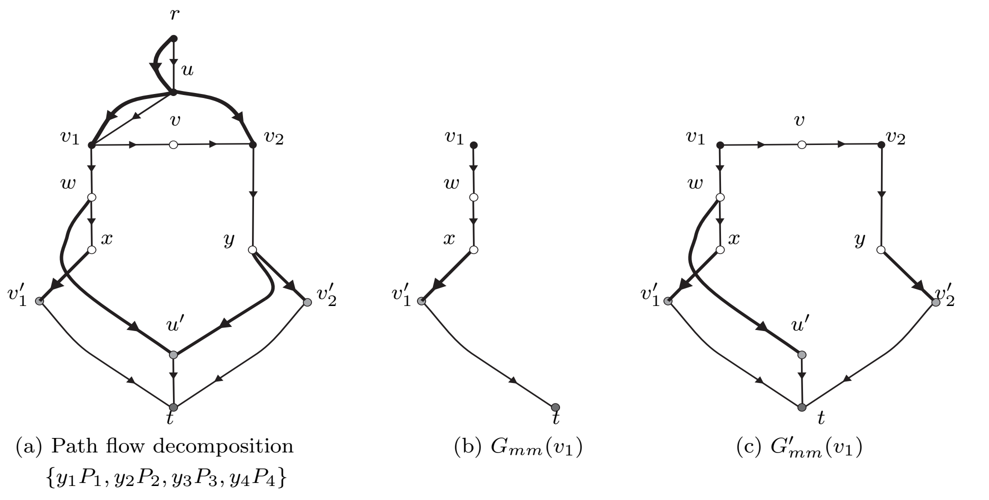


FIG. 5. *The Construction of* $G'_{mm}(v_1)$.

Since $\{y_1Q_1, y_2Q_2, \ldots, y_\alpha Q_\alpha\}$ is a path flow decomposition of a $v-t$ flow in $G'_{mm}(v)$ of value $\sum_{i=1}^{\alpha} y_i$, and $x(u,v)$ is not more than the maximum flow value of network $G_{mm}(v)$ (recall the construction of $G_{mm}(r)$), (3), (5), and Lemma 2.4 (with $G_{mm}(v)$, $G'_{mm}(v)$, $\sum_{j=1}^{\alpha} y_j$ and $x(u,v)$ in place of N', N, λ and λ' over there, respectively) guarantee the existence of an integral $v-t$ flow z in $G'_{mm}(v)$ such that

(6) $\sum_{j=1}^{\alpha} y_j = |z| \ge \sum_{a'\in G_{mm}(v)} z(a',t) \ge x(u,v)$.

Define an integral vector x' on the arcs of $G_{mm}(r)$ by

(7) $x'(a,b) = x(a,b)$ if (a,b) is outside $G'_{mm}(v)$ and $x'(a,b) = z(a,b) + x(a,b) - \sum_{j=1:(a,b)\in Q_j}^{\alpha} y_j$ otherwise.

(8) x' is also an integral maximum $r-t$ flow in $G_{mm}(r)$.

Using (5), (7) and the $v-t$ flow z, it is easy to see that x' satisfies the capacity constraint and conserves at each vertex, so x' is an $r-t$ flow. Observe that

$$\begin{aligned}
\sum_{a' \in G_{mm}(r)} x'(a',t) &= \sum_{a' \notin G'_{mm}(v)} x'(a',t) + \sum_{a' \in G'_{mm}(v)} x'(a',t) \\
&= \sum_{a' \notin G'_{mm}(v)} x(a',t) + \sum_{a' \in G'_{mm}(v)} [z(a',t) + x(a',t) - \sum_{j=1:(a',t)\in Q_j}^{\alpha} y_j] \quad (\text{by } (7)) \\
&= \sum_{a' \in G_{mm}(r)} x(a',t) + \sum_{a' \in G'_{mm}(v)} z(a',t) - \sum_{j=1}^{\alpha} y_j \\
&= \sum_{a' \in G_{mm}(r)} x(a',t).
\end{aligned}$$

Since x is a maximum $r-t$ flow in $G_{mm}(r)$, so is x', and hence we have (8).

(9) $x'(a',t) \le x(a',t)$ for any vertex $a' \in G'_{mm}(v) \backslash G_{mm}(v)$.

Indeed, since (a',t) is contained in $G'_{mm}(v)$ for $a' \in G'_{mm}(v)\backslash G_{mm}(v)$, from (7) we deduce that $x'(a',t) = z(a',t)+x(a',t)-\sum_{j=1:\,(a',t)\in Q_j}^{\alpha} y_j \le c'(a',t)+x(a',t)-\sum_{j=1:\,(a',t)\in Q_j}^{\alpha} y_j = x(a',t)$ by (5), as desired.

We propose to prove that

(10) for any newly added arc (p,q) of $G_{mm}(r)$ with $\pi(q) \le \pi(v)$,

$$x'(p,q) \le \sum_{a' \in G_{mm}(q)} x'(a',t). \tag{3.1}$$

To this end, observe that, by Lemma 3.4(iv), $G_{mm}(v)$ is a digraph rooted at v, so is $G'_{mm}(v)$ by the construction. Hence, by Lemma 3.4(ii), we have

(11) $G'_{mm}(v)$ contains no vertex $a \in V_h \cup \{r\}$ with $\pi(a) < \pi(v)$.

Let us now verify that

(12) $x'(u,v) \le \sum_{a' \in G_{mm}(v)} x'(a',t)$.

By (11), u is outside $G'_{mm}(v)$. Thus by (7) we have $x'(u,v) = x(u,v)$. For each arc (a,b) in $G'_{mm}(v)$, by Theorem 2.5 we get $x(a,b) \ge \sum_{j=1:(a,b)\in Q_j}^{\alpha} y_j$, so $x'(a,b) \ge z(a,b)$ (recall (7)). It follows from (6) that $x'(u,v) = x(u,v) \le \sum_{a' \in G_{mm}(v)} z(a',t) \le \sum_{a' \in G_{mm}(v)} x'(a',t)$. Hence (12) is proved.

Let us turn to consider an arbitrary newly added arc (p,q) of $G_{mm}(r)$ with $(p,q) \neq (u,v)$ and $\pi(q) \le \pi(v)$. Since the newly added arcs form the arc set of the head dominator tree T_h, (u,v) is the unique newly added arc entering v. By (11), we get

(13) $\pi(q) < \pi(v)$ and hence (p, q) is outside $G'_{mm}(v)$ by (11).

Thus, by Lemma 3.4(iii), either $G_{mm}(q)\backslash\{t\}$ and $G_{mm}(v)\backslash\{t\}$ are vertex disjoint or $G_{mm}(v)$ is contained in $G_{mm}(q)$; we shall deal with these two cases separately.

(14) If $G_{mm}(q)\backslash\{t\}$ and $G_{mm}(v)\backslash\{t\}$ are vertex disjoint, then (3.1) holds.

To justify (14), we first claim that $G_{mm}(q)\backslash\{t\}$ and $G'_{mm}(v)\backslash\{t\}$ are vertex disjoint. Suppose the contrary: b is a common vertex of these two digraphs. From Lemma 3.4(iv) and the construction of $G'_{mm}(v)$, we see that $G'_{mm}(v)$ is a digraph rooted at v. Hence there is a path P in $G'_{mm}(v)\backslash\{t\}$ from v to b. It follows from (13) and Lemma 3.4(ii) that P contains an entry vertex of $G_{mm}(q)\backslash\{t\}$ other than q. However, by Lemma 2.2(ii), (iii) and the construction of $G_{mm}(q)$, q is the unique entry vertex of $G_{mm}(q)\backslash\{t\}$, this contradiction establishes the claim. Using the claim, (13), and (7), we get $x'(a', t) = x(a, t)$ for each $a' \in G_{mm}(q)$, and $x'(p, q) = x(p, q)$. In view of (2), $x(p, q) \leq \sum_{a'\in G_{mm}(q)} x(a', t)$, so we have (3.1) and hence (14).

(15) If $G_{mm}(v)$ is contained in $G_{mm}(q)$, then (3.1) holds.

By direct computation, we have

$$\begin{aligned}
&\sum_{a'\in G_{mm}(q)} x'(a',t) - \sum_{a'\in G_{mm}(q)} x(a',t) = \sum_{a'\in G_{mm}(q)\cap G'_{mm}(v)} [x'(a',t) - x(a',t)] \\
=\ & \sum_{a'\in G_{mm}(v)} [x'(a',t) - x(a',t)] + \sum_{a'\in G_{mm}(q)\cap (G'_{mm}(v)\backslash G_{mm}(v))} [x'(a',t) - x(a',t)] \quad \text{(by hypothesis)} \\
\geq\ & \sum_{a'\in G_{mm}(v)} [x'(a',t) - x(a',t)] + \sum_{a'\in G'_{mm}(v)\backslash G_{mm}(v)} [x'(a',t) - x(a',t)] \quad \text{(by (9))} \\
=\ & \sum_{a'\in G'_{mm}(v)} x'(a',t) - \sum_{a'\in G'_{mm}(v)} x(a',t) \\
=\ & \sum_{a'\in G'_{mm}(v)} \left[z(a',t) + x(a',t) - \sum_{j=1:\, a'\in Q_j}^{\alpha} y_j \right] - \sum_{a'\in G'_{mm}(v)} x(a',t) \quad \text{(by (7))} \\
=\ & |z| - \sum_{j=1}^{\alpha} y_j = 0 \quad \text{(by (6))}.
\end{aligned}$$

Thus $\sum_{a'\in G_{mm}(q)} x'(a', t) \geq \sum_{a'\in G_{mm}(q)} x(a', t)$. Notice that by (13) and (7), $x'(p, q) = x(p, q)$, and by (2), $x(p, q) \leq \sum_{a'\in G_{mm}(q)} x(a', t)$, so we obtain (3.1) and hence (15).

Combining (12), (14) and (15), we get (10). Now let us replace x by x' and repeat the process. From (2), (8) and (10) we conclude that a good integral maximum $r-t$ flow in $G_{mm}(r)$ can be obtained after at most n iterations. Since the initial maximum flow x and x' in (7) can both be found in $O(nm \log(n^2/m))$ time [13], the whole algorithm runs in time $O(n^2 m \log(n^2/m))$.

□

Lemma 3.6. *Given a good integral $r - t$ flow x in $G_{mm}(r)$, there is an $O(m^2)$ algorithm for finding a cycle packing of G with weight $|x|$ and with at most m distinct cycles, where $m = |A|$.*

Proof. The statement clearly holds if x is a zero flow. So we assume x is nonzero. Let $p(u)$ denote the parent of u on T_h for each $u \in V_h \backslash \{r\}$, let π be the DFS order of T_h specified in Lemma 2.3, and let v', the image of v, be the vertex of $G_{mm}(r)$ (recall the construction) such that

(i) $x(v', t) > 0$, and

(ii) subject to (i), $\pi(v)$ is maximized.

Condition (i) guarantees the existence of an $r-t$ path P through v' in $G_{mm}(r)$ such that $x(e) > 0$ for any arc e on P. Now let $Q = u_0 u_1 \ldots u_k$ denote the path from r to v on T_h, where $u_0 = r$, $u_k = v$, and $u_i = p(u_{i+1})$ for $i = 0, 1, \ldots, k-1$. By Lemma 2.2(ii), (iii), and the construction of $G_{mm}(r)$, u_i is the only entry vertex of $G_{mm}(u_i)\backslash\{t\}$ for $0 \leq i \leq k$. By Lemma 3.4(iii), $G_{mm}(u_j)$ is contained in $G_{mm}(u_i)$ whenever $0 \leq i < j \leq k$. Hence vertices $u_0, u_1, \ldots, u_k$ appear sequentially on P.

Using Lemma 3.4(i), we have

(1) v' is contained in no $G_{mm}(u)$ with $\pi(u) > \pi(v)$, and v' is contained in $G_{mm}(u)$ with $\pi(u) < \pi(v)$ iff u dominates v.

We claim that

(2) $P[v, v']$ contains no arc $(p(u), u)$ of T_h with $\pi(p(u)) \geq \pi(v)$.

Suppose to the contrary that such u exists. Since $x(p(u), u) > 0$ and x is a good flow, $x(a', t) > 0$ for some vertex a' in $G_{mm}(u)$. Thus $\pi(a) \geq \pi(u) > \pi(v)$, contradicting the selection (ii) of v'. So we get (2).

Now let R be the path obtained from P by replacing $P[u_i, u_{i+1}]$ with the newly added arc (u_i, u_{i+1}) whenever $x(u_i, u_{i+1}) > 0$ for $i = 0, 1, \ldots, k-1$. Then $x(e) > 0$ for each arc e on R. Let δ denote the minimum $x(e)$ for all e on R. We define a vector x' on the arc set of $G_{mm}(r)$ as follows: $x'(e) = x(e) - \delta$ if e is an arc on R and $x(e)$ otherwise. From (1) and (2), we can conclude that x' remains to be a good flow of $G_{mm}(r)$ and $|x'| = |x| - \delta$. Let e' denote the arc of P entering v', and let C be the cycle of G uniquely contained in the union of $P[v, v']$ and the back arc corresponding to e'. (For the example depicted in FIG. 5, $P_3[v_2, v_2'] \cup \{(y, v_2)\} = v_2 y v_2' t \cup \{(y, v_2)\}$ contains a unique cycle $v_2 y v_2$ of G in FIG. 2.) Now let $\mathcal{C}$ contain C such that the multiplicity of C, $m(C)$, is δ. Replace x by x' and repeat the process until x becomes a zero flow. Clearly, $\mathcal{C}$ is a cycle packing with weight equal to the value of the initial $r - t$ flow x and contains at most m distinct cycles.

Since P can be found by breadth first search in $O(m)$ time and the number of iterations is bounded above by the number of distinct cycles in $\mathcal{C}$ (which is at most m), the algorithm runs in $O(m^2)$ time. □

Lemma 3.7. *For any cycle packing $\mathcal{C}$ of G with k distinct cycles, where $k \leq m$, there is an integral $r - t$ flow x of $G_{mm}(r)$ such that $|x|$ equals the weight of $\mathcal{C}$.*

Proof. Let $C_1, C_2, \ldots, C_k$ be all distinct cycles in $\mathcal{C}$ such that the multiplicity of each C_i, $m(C_i)$, is y_i. Recall Theorem 2.1(v), each C_i contains precisely one back arc (v_i, u_i). Let P_i be the unique path from r to u_i on T_h (we view P_i as a path in $G_{mm}(r)$ consisting of the corresponding newly added arcs), and let Q_i denote the concatenation of P_i, $C_i \backslash \{(v_i, u_i)\}$, (v_i, u'_i), and (u'_i, t) for $i = 1, 2, \ldots, k$. (For example, the cycle $v_2 y v_2$ of G in Fig. 2 corresponds to path $ruv_2yv'_2t$ in $G_{mm}(r)$.) We aim to show that $\{y_1Q_1, y_2Q_2, \ldots, y_kQ_k\}$ is a path flow decomposition of a flow x in $G_{mm}(r)$ (which is acyclic by Lemma 3.4(iv)). For this purpose, it suffices to check the capacity constraint.

Suppose to the contrary that the capacity constraint is violated on some arc (u, v). Then (u, v) must be a newly added arc. We select such an arc so that $\pi(v)$ is maximized. Thus

(i) $\sum_{i:\,(u,v)\in Q_i} y_i > c(u, v)$ and

(ii) $\sum_{i:\,(a,b)\in Q_i} y_i \leq c(a, b)$ for all arcs (a, b) of $G_{mm}(v)$.

For each $i = 1, 2, \ldots, k$, since $C_i \backslash \{(v_i, u_i)\}$ is a DAG path by Theorem 2.1(v) and (u, v) is a newly added arc, Q_i passes through (u, v) iff P_i passes through (u, v) iff v dominates u_i. Thus if Q_i passes through (u, v), then all the vertices on C_i are in V_v by the definition of V_v. So it follows from the construction of $G_{mm}(v)$ that $Q_i[v, t]$ is entirely contained in $G_{mm}(v)$. Without loss of generality, we assume $Q_1, Q_2, \ldots, Q_\alpha$ are all the paths in $Q_1, Q_2, \ldots, Q_k$ that pass through (u, v). In view of (ii), $\{y_1Q_1[v, t], y_2Q_2[v, t], \ldots, y_\alpha Q_\alpha[v, t]\}$ is a path flow decomposition of a $v - t$ flow in $G_{mm}(v)$. Observe that $\sum_{i=1}^{\alpha} y_i$ is bounded above by the maximum flow value of $G_{mm}(v)$ which is $c(u, v)$ by the construction of $G_{mm}(r)$. Hence, $\sum_{i:(u,v)\in Q_i} y_i = \sum_{i=1}^{\alpha} y_i \leq c(u, v)$, contradicting (i). This completes the proof. □

We are ready to present our algorithm for finding a maximum cycle packing in any arc-weighted reducible flow graph.

Algorithm for finding a maximum (weighted) cycle packing

Input: An arc-weighted reducible flow graph $G = (V, A, r)$.

Output: A maximum cycle packing $\mathcal{C}$ of G.

Step 0. Construct Ramachandran's flow network $G_{mm}(r)$ (see the beginning of this section).
Step 1. Find a good integral maximum $r-t$ flow x in $G_{mm}(r)$ as described in Lemma 3.5.
Step 2. Convert x to a cycle packing $\mathcal{C}$ of G as described in Lemma 3.6 and return $\mathcal{C}$.

The correctness of this algorithm follows instantly from Lemma 3.5-3.7. Since the dominator tree can be constructed in linear time [14], so can be the underlying digraph of $G_{mm}(r)$. By calling the maximum flow algorithm [13] at most n times, we can get the capacities of all newly added arcs in $G_{mm}(r)$. In view of the complexity stated in each lemma, we conclude that the algorithm runs in $O(n^2 m \log(n^2/m))$ time. This completes the proof of Theorem 3.3.

4 Concluding Remarks

In this paper we have obtained a polynomial time algorithm for finding a maximum weighted cycle packing in any arc-weighted reducible flow graph. We remark that our algorithm can also be employed to solve the maximum cycle packing problem in the vertex weighted case as the latter can be easily transformed into the former. Moreover, there is a linear time algorithm for finding maximum vertex-disjoint cycles in any reducible flow graph (the dual of Shamir's problem [22]).

Minimax relations play important roles in combinatorics and optimization. In addition to their great theoretical interest, they often yield polynomial-time solutions of the corresponding problems. As shown by Ramachandran [20] and by Lucchesi and Younger [18], reducible flow graphs and planar digraphs satisfy the minimax arc theorem on packing and covering cycles. Major open problems in this direction are to characterize all digraphs with this minimax arc (resp. vertex) relation, for any nonnegative integral weight function defined on the arc (resp. vertex) set. See Cai *et al.* [4, 5, 6] for a complete characterization of all tournaments and bipartite tournaments and Ding *et al.* [8, 9] for the description of all undirected graphs.

Ramachandran [21] came up with parallel algorithms for recognizing reducible flow graphs, for finding dominators, and for finding a minimum feedback vertex set in an unweighted reducible flow graph. Certainly, parallel algorithms for the general cycle packing and feedback set problems on reducible flow graphs also deserve good research efforts.

Acknowledgment. The authors are deeply indebted to two anonymous referees for their invaluable comments and suggestions which have greatly improved the presentation of this paper.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows – Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[2] R. K. Ahuja, J. B. Orlin, and R. E. Tarjan, Improved time bounds for the maximum flow problem, *SIAM J. Comput.* **18** (1989), 939-954.

[3] A. V. Aho and J. D. Ullman, *Principle of Compiler Design*, Addison-Wesley Reading, MA, 1977.

[4] M. C. Cai, X. T. Deng, and W. Zang, A TDI system and its application to approximation algorithms, in: *Proc. 39^{th} IEEE Symposium on Foundations of Computer Science*, Palo Alto, CA, 1998, pp. 227-233.

[5] M. C. Cai, X. T. Deng, and W. Zang, An approximation algorithm for feedback vertex sets in tournaments, *SIAM J. Comput.* **30** (2001), 1993-2007.

[6] M. C. Cai, X. T. Deng, and W. Zang, A min-max theorem on feedback vertex sets, *Math. Oper. Res.* **27** (2002), 361-371.

[7] A. Caprara, A. Panconesi, and R. Rizzi, Packing cycles in undirected graphs, *J. Algorithms* **48** (2003), 239-256.

[8] G. Ding and W. Zang, Packing cycles in graphs, *J. Combin. Theory Ser. B* **86** (2002), 381-407.

[9] G. Ding, Z. Xu, and W. Zang, Packing cycles in graphs, II, *J. Combin. Theory Ser. B* **87** (2003), 244-253.

[10] L. R. Ford, Jr. and D. R. Fulkerson, *Flows in Networks*, Princeton Univ. Press, Princeton, NJ, 1962.

[11] A. Frank and A. Gyarfas, Directed graphs and computer programs, in: *Problemes Combinatoires et Theorie des Graphes*, Colloque Internationaux C.N.R.S. **260** (1976), pp. 157-158.

[12] M. R. Garey and D. S. Johnson, *Computers and Intractability*, W. H. Freeman and Company, New York, 1979.

[13] A. Goldberg and R. E. Tarjan, A new approach to the maximum flow problem, in: *Proc. 18th Annual ACM Symposium on Theory of Computing*, 1985, pp. 136-146.

[14] D. Harel, A linear algorithm for finding dominators in flow graphs and related problems, in: *Proc. 17th Annual ACM Symposium on Theory of Computing*, 1984, pp. 185-194.

[15] M. S. Hecht and J. D. Ullman, Flow graph reducibility, *SIAM J. Comput.* **1** (1972), 188-202.

[16] M. S. Hecht and J. D. Ullman, Characterizations of reducible graphs, *J. ACM* **21** (1974), 367-375.

[17] J. E. Hopcroft and J. D. Ullman, Am $n \log n$ algorithm for detecting reducible graphs, in: *Proc. 6th Annual Princeton Conference on Information Sciences and Systems*, Princeton, NJ, 1972, pp. 119-122.

[18] C.L. Lucchesi and D.H. Younger, A minimax theorem for directed graphs, *J. London Math. Soc.* **17**(1978), 369-374.

[19] V. Ramachandran, Finding a minimum feedback arc set in reducible flow graphs, *J. Algorithms* **9** (1988), 299-313.

[20] V. Ramachandran, A minimax arc theorem for reducible flow graphs, *SIAM J. Discrete Math.* **3** (1990), 554-560.

[21] V. Ramachandran, Parallel algorithms for reducible flow graphs, *J. Algorithms* **23** (1997), 1-31.

[22] A. Shamir, A linear time algorithm for finding minimum cutsets in reducible graphs, *SIAM J. Comput.* **8** (1979), 645-655.

[23] R. E. Tarjan, Depth-first search and linear graph algorithm, *SIAM J. Comput.* **1** (1972), 146-160.

[24] R. E. Tarjan, Testing flow reducibility, *J. Comput. System Sci.* **9** (1974), 355-365.

[25] R. E. Tarjan, *Data Structure and Network Algorithms*, SIAM, Philadelphia, PA, 1983.