2D-5

# On k-component Algorithms for Graphs and Digraphs.

Seishi Makino
IBM Research, Tokyo Research Laboratory

## 1. Introduction.

Consider a digraph (or graph) $G$ without any self-loops or parallel-edges. We denote the vertex-set and edge-set of $G$ by $V(G)$ and $E(G)$, respectively. We denote the cardinality of $V(G)$ and $E(G)$, namely, $|V(G)|$ or $|E(G)|$ by $n$ and $e$, respectively. $G$ is *k-connected*, if there exist at least $k$ vertex-disjoint paths from $v$ to $w$, for every ordered pair of $v$ and $w$ $(v,w \in V(G))$. A *k-component* of $G$ is a maximal $k$-connected subgraph of $G$ (see Figure 1).

Hopcroft and Tarjan discovered algorithms for finding 1-components, 2-components and 3-components of a graph, as well as 1-components of a digraph, within $O(n+e)$ time [Ta72], [HoTa73]. The existence of such linear algorithms for finding $k$-components of a graph for any fixed $k(>3)$ was conjectured in [AhHoUl74]; however, no such linear algorithm has been found. Recently, Kanevsky and Ramachandran discovered an algorithm that finds all separating triplets of a 3-connected graph in $O(n^2)$ time [KaRa87], which suggests the existence of an efficient algorithm for finding 4-components of a graph; however, finding all separating triplets does not directly yield a decomposition into 4-components, and the problem still remains.

Matula discovered a polynomial algorithm that finds all the $k$-components of a graph, using cluster analysis techniques [Ma77]. Note that the problems for graphs are easily reducible to those for symmetrical digraphs, although the converse is not true. Our purpose is to construct an algorithm to find all the k-components of a digraph. Our algorithm can be regarded as an extension and refinement (a practical, and probably more efficient version) of Matula's.

Proceeding to our main results, we need to introduce some notations. For $v,w \in V$, let $[v,w]$ denote a directed edge which leaves $v$ and enters $w$. For $v \in V$, let $\Gamma^+(v) = \{w|\ [v,w] \in E(G)\}$, and $\Gamma^-(v) = \{w|\ [w,v] \in E(G)\}$. We denote $\deg(v) = \min\{\Gamma^+(v), \Gamma^-(v)\}$. $G$ is *complete* if $\Gamma^+(v) = \Gamma^-(v) = n-1$, for $\forall v \in V(G)$. For $U(\subset V(G))$, $\ll U \gg_G$ denotes the *induced subgraph* of $G$ by $U$, that is, a subgraph of $G$ whose vertex-set is $U$, and whose edge-set is $\{[v,w] \in E(G) | v,w \in U\}$. $S(\subset V(G))$ is a *separator* of $G$ if $\ll V(G) \backslash S \gg_G$ is not 1-connected. For $l \geq 1$, a *l-separator*, denoted by $\xi^l(G)$, is a minimum separator, $S$, of $G$, such that $|S| \leq l$, Note that $\xi^l(G)$ is nothing but a minimum separator of $G$, if it exists, and that otherwise $\xi^l(G)$ is empty (see Figure 1). We assume all digraphs in our algorithm are represented by adjacency structures [AhHoUl74], [Ta72], and manipulated in an efficient way. Our results are summarized as follows.

**Theorem 1** *Let G be a digraph with n vertices and e edges. All the possible v-components* $(k \leq v \leq l)$ *of G are found within* $O((n-k) \times Max\{n+e, T_l\})$ *time, where* $T_l$ *is the time bound for computing* $\xi^l(G)$.

Many algorithm are available for computing $\xi^l(G)$ for digraphs [EvTa75],[Ga80]. In particular, Galil's approach [Ga80] yields an $O(l \times (n+e) \times \sqrt{n} \times Max\{l, \sqrt{n}\})$ time algorithm for computing $\xi^l(G)$. Thus we have:

**Corollary 1** *All the possible v-components* $(k \leq v \leq l)$ *of a digraph G are found within* $O((n-k) \times l \times (n+e) \times \sqrt{n} \times Max\{l, \sqrt{n}\})$ *time.*

Provided that $G$ is a symmetrical digraph, or a graph, we can use more efficient algorithms for computing $\xi^l(G)$, [EsHa84], [GrHa86]. Combining Granot-Hassin's algorithm [GrHa86] with Even-Tarjan's maximum-flow algorithm [EvTa75], we can compute $\xi^l(G)$ in $O(n \times (n+e) \times min\{l, \sqrt{n}\})$ time. Thus we have:

**Corollary 2** *All the possible v-components* $(k \leq v \leq l)$ *of a symmetrical digraph (or graph) G are found within* $O((n-k) \times n \times (n+e) \times min\{l, \sqrt{n}\})$ *time.*

Note that Corollary 1 and Corollary 2 show the same time bound, if $l \leq \sqrt{n}$ . In particular, combined with Kanevsky-Ramachandran and Hopcroft-Tarjan algorithms [KaRa87], [HoTa73], Theorem 1 yields an $O(n^3)$ bound for finding 4-components of a graph.

## 2. Proof of Theorem 1.

The proof of Theorem 1 is essentially the same as that shown in [M88].

## 3. Algorithm.

```
1. procedure FMC(k,l,G)
2  begin if k ≤ l then
3.   begin if |V(G)| ≥ k + 1 then
4.     repeat
5.       for ∀v ∈ V(G) do
6.       if deg(v) < k then
           eliminate v, together with
           Γ⁺(v), and Γ⁻(v) from G;
7.       find l-components of G;
8.       eliminate all edges linking between
         any two l-components from G;
9.     until
         |V(G)| < k + 1, or min{deg(v)|v ∈ V(G)} ≥ k;
10.    if |V(G)| = k + 1 and G is complete then
11.      output G as a k-component;
12.    if |V(G)| > k + 1  then
13.      for each l-component G_i of G do
14.        begin find ξ^l(G_i);
15.        if ξ^l(G_i) is empty then
16.          output G_i as a (l + 1)-component;
17.        else if |ξ^l(G_i)| ≥ k then
18.          begin output a copy of G_i
               as a |ξ^l(G_i)|-component;
19.            G_i ← ≪V(G_i)\ξ^l(G_i)≫_{G_i};
20.            find l-components of G_i;
21.            eliminate all edges linking between
               any two l-components from G_i;
22.            k' ← Max{k,|ξ^l(G_i)| + 1};
23.            for each l-component P_i^j of G_i do
24.              begin Let G_i^j ← ≪V(P_i^j) ∪ ξ^l(G_i)≫_{G_i};
25.                  FMC(k',l,G_i^j);
26.              end;
27.          end;
28.        end;
29.  end;
30. end;
```

## 4. Remarks.

In our algorithm, degree checks and eliminations of unavailable edges in lines 4 - 8 are not necessary to achieve the time bounds of section 1. However, in many cases, these simple tricks yield considerable reduction of the cost, while the precise estimation still remains.

## 5. References.


[AhHoUl74] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and analysis of computer algorithms, Addison-Wesley, Reading, MA, 1974.*

[EsHa84] A.H. Esfhanian and S. L. Hakimi, *On computing the connectivities of graphs., Networks 14, 1984, 355-366.*

[EvTa75] S. Even and R.E. Tarjan, *Networkflow and testing graph connectivity, SIAM. J. Comput.4, No.4, 1975, 507-518.*

[Ga80] Z. Galil, *Finding the vertex connectivity of graphs, SIAM. J. Comput. 9, No.1, 1980, 197-199.*

[GrHa86] F. Granot and R. Hassin, *Multiterminal maximum flows in node capacitated networks, Discrete Appl. Math. 13, Nos.2,3, 1986, 157-163.*

[HoTa73] J.E. Hopcroft and R.E. Tarjan, *Dividing a graph into triconnected components, SIAM. J. Comput. 2, No.3, 1973, 135-158.*

[KaRa87] A. Kanevsky and V. Ramachandran, *Improved algorithms for graph four-connectivity, FOCS, IEEE, 1987, 252-259.*

[M88] S. Makino, *An algorithm for finding all the k-components of a digraph, Intern. J. Comput. Math., 1988, to appear.*

[Ma77] D. Matula, *Graph theoretic techniques for cluster analysis algorithms, F. Van Ryzon, ed., Classification and Clustering, Academic Press, New York, 1977.*

[Ta72] R. Tarjan, *Depth-first search and linear graph algorithms, SIAM. J. Comupt., 1, No.2, 1972, 146-160.*


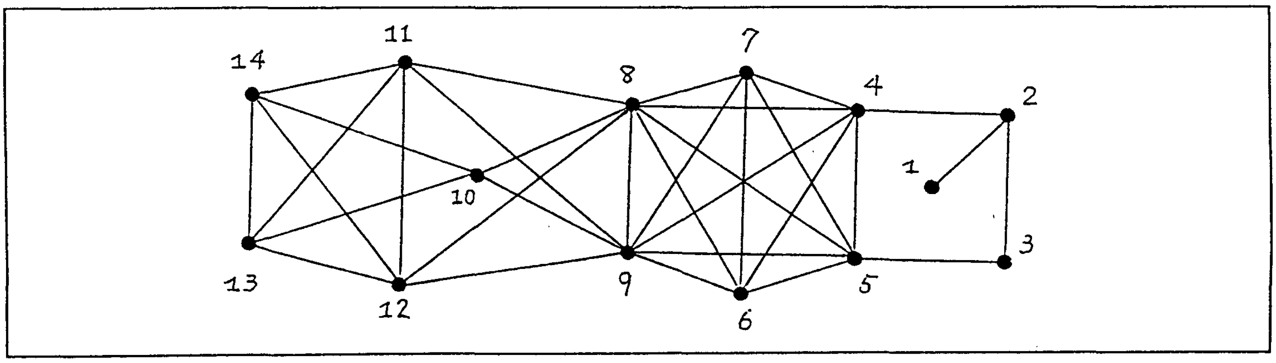


**Figure 1.** A graph G with 14 vertices. G itself is the 1-component. 2-component is $\ll 2,3,...,14 \gg_G$, 3-component is $\ll 4,5,...,14 \gg_G$, 4 and 5-component is $\ll 4,5,...,9 \gg_G$. $\xi^4(\ll 4,5,...,14 \gg_G)$ is {8,9,10}. $\xi^2(\ll 4,5,...,14 \gg_G)$ is empty.