A Measure of Parallelization for the Lexicographically First Maximal Subgraph Problems

Ryuhei Uehara

Center for Information Science, Tokyo Woman's Christian University,
uehara@twcu.ac.jp

Abstract. A maximum directed tree size (MDTS) is defined by the maximum number of the vertices of a directed tree on the directed acyclic graph of a given undirected graph. The MDTS of a graph *measures* the parallelization for the lexicographically first maximal subgraph (LFMS) problems. That is, the complexity of the problems on a family $\mathcal{G}$ of graphs gradually increases as the value measured on each graph in the family grows; (1) if the MDTS of each graph in $\mathcal{G}$ is $O(\log^k n)$, the lexicographically first maximal independent set problem on $\mathcal{G}$ is in NC^{k+1}, and the LFMS problem for π is in NC^{k+s}, where π is a property on graphs such that π is nontrivial, hereditary, and NC^{s-1} testable; (2) both problems above are P-complete if the MDTS of each graph in $\mathcal{G}$ is cn^ϵ. It is worth remarking that the problem to compute the MDTS is in NC^2. This is important in the sense that a "measure" means only if measuring the complexity of a problem is easier than solving the problem.
Key words: Analysis of algorithms, P-completeness, NC algorithms, the lexicographically first maximal independent set problem, the lexicographically first maximal subgraph problems.

1 Introduction

The maximal independent set (MIS) problem is a typical maximality problem, that is to find a maximal vertex-induced subgraph that satisfies a specified graph property. Since Karp and Wigderson showed that the MIS problem is in the class NC [12], much work has been devoted to the study of parallel complexity of maximality problems [14, 1, 7, 6, 4, 17, 18]. On the other hand, the lexicographically first maximal independent set (LFMIS) problem is a typical P-complete problem [3], and P-completeness of the lexicographically first maximal subgraph (LFMS) problems for some properties π was shown [15, 17] (see also [8] for a comprehensive reference). As noticed by Iwama and Iwamoto in [10], one of approaches to making clear the boundary between NC and P-completeness is to find (sufficient) conditions for problems to be in NC or to be P-complete. In [10], they produced a new problem on graphs, called α-connectivity, whose complexity gradually increases as the value of α grows. Our approach is slightly different from theirs. We provide a "measure" of parallelization for the LFMIS and LFMS problems. That is, the complexity of the problems on a family of graphs gradually increases as

the value measured on each graph in the family grows. The measure is the *maximum directed tree size* (MDTS) of G, which is defined by the maximum number of the vertices of a directed tree on the directed acyclic graph of G. The problem to compute the MDTS is in NC^2. This result is important since a "measure" means to us only if measuring the complexity of a problem is easier than solving the problem. We define two graph families $\mathcal{G}_{\log^k n}$ and $\mathcal{G}_{\rm poly}$ as follows:

$\mathcal{G}_{\log^k n}$ contains every graph whose MDTS is $O(\log^k n)$ for some positive integer k, and

$\mathcal{G}_{\rm poly}$ contains every graph whose MDTS is at most cn^ϵ for any fixed positive constant c and ϵ.

Then the results in this paper are the following: The LFMIS problem on $\mathcal{G}_{\log^k n}$ is in NC^{k+1}, and the problem on $\mathcal{G}_{\rm poly}$ is P-complete. Let π be a property on graphs such that π is nontrivial, hereditary on vertex-induced subgraphs, satisfied by all independent edges, and NC^{s-1} testable. Then, the LFMS problem for π on $\mathcal{G}_{\log^k n}$ is in NC^{k+s}, and the problem on $\mathcal{G}_{\rm poly}$ is P-complete.

2 Preliminaries

We will deal only with graphs and digraphs without loops or multiple edges. Throughout this paper, unless stated otherwise, G always denotes the input (undirected) graph, $V = \{0, 1, \cdots, n-1\}$ and E denote the set of vertices and edges in G, respectively. Without loss of generality, we assume that G is connected. The *neighborhood* of a vertex v in G, denoted $N_G(v)$, is the set of vertices in G adjacent to v. The *degree* of a vertex v in G is $|N_G(v)|$, and denoted by $\mathrm{d}_G(v)$. Vertices of degree 0 are called *isolated vertices*. For $U \subseteq V$, $N_G(U)$ is $\cup_{u\in U} N_G(u)$, and $G[U]$ is the graph (U, F), where $F = \{\{u, v\} \mid u, v \in U \text{ and } \{u, v\} \in E\}$. For a graph G, $|G|$ denotes the number of vertices in G.

Let $X = \{x_1, x_2, \cdots, x_k\}$ and $Y = \{y_1, y_2, \cdots, y_h\}$ be any subsets of V. (Throughout this paper, we assume that sets are always sorted, that is, $x_i < x_j$ and $y_i < y_j$ for each $1 \le i < j \le k, h$.) Let $<$ be the total ordering on the sets X and Y defined as follows: $X < Y$ if and only if $x_i = y_i$ for every i with $1 \le i \le k$, or there is an index $i' \ge 1$ such that $x_j = y_j$ for every $1 \le j < i'$, and $x_{i'} < y_{i'}$.

A subset U of V is called an *independent set* if $G[U]$ only contains isolated vertices. A *maximal independent set* (MIS) in G is an independent set that is not properly contained in another independent set. The MIS problem is to find, given a graph G, an MIS in G. The *lexicographically first maximal independent set* (LFMIS) in G is the MIS I in G such that $I < J$ for every MIS J in G. The LFMIS problem is to find, given a graph G, the LFMIS in G.

For $G = (V, E)$, $\vec{G} = (V, \vec{E})$ is defined by the directed acyclic graph obtained from G by replacing each edge $\{u, v\}$ by the arc $(\min\{u, v\}, \max\{u, v\})$. (It is easy to show that the resulting graph $\vec{G}$ is acyclic for any graph G.) The *neighborhood* of a vertex v in $\vec{G}$, denoted $N_{\vec{G}}(v)$, is the set of vertices u in $\vec{G}$ such that $(v, u) \in \vec{E}$. The *outdegree* of a vertex v in $\vec{G}$, denoted by $\mathrm{d}^+{}_{\vec{G}}(v)$, is $|N_{\vec{G}}(v)|$,

and the *indegree* of a vertex v in $\vec{G}$, denoted by $\mathrm{d}^-_{\vec{G}}(v)$, is $|N_G(v)| - |N_{\vec{G}}(v)|$. We mention that any directed acyclic graph $\vec{G}$ has at least one vertex v with $\mathrm{d}^-_{\vec{G}}(v) = 0$, and at least one vertex u with $\mathrm{d}^+_{\vec{G}}(u) = 0$ (see [9, Theorem 16.2]).

Lemma 1. *Let v be a vertex with $\mathrm{d}^-_{\vec{G}}(v) = 0$. Then, v is in the LFMIS of G.*

Proof. Let I be the LFMIS of G. To derive a contradiction, assume that $v \notin I$. Let $U = N_G(v) \cap I$. Then, by the maximality of I and the assumption, $U \neq \emptyset$. Let $J = I - U + \{v\}$. Clearly, J is an independent set, and it is not difficult to see that there is a maximal independent set K such that $J \subseteq K$, consequently, $I - U + \{v\} \subseteq K$. For each u in U, since $\mathrm{d}^-_{\vec{G}}(v) = 0$, $u > v$ holds. This implies that $K < I$, which contradicts that I is the LFMIS of G. □

Now, we define the measure of the parallelization for the lexicographically first maximal problems. A *directed tree* T in G is the edge-induced subgraph of G such that T is a rooted tree, and every path from the root to each leaf is a directed path on $\vec{T}$. The *maximum directed tree size* (MDTS) of G, denoted by $\mathrm{MDTS}(G)$, is defined by the maximum number of vertices of a directed tree in G.

Lemma 2. *Let T be a directed tree in G with $|T| = \mathrm{MDTS}(G)$. Then $\mathrm{d}^-_{\vec{G}}(r) = 0$, where r is the root of T.*

Proof. To derive contradictions, assume that $\mathrm{d}^-_{\vec{G}}(r) > 0$. Then there is a vertex s with (s, r) in $\vec{G}$. First we assume that s is in T. Then, the directed path from r to s with the arc (s, r) makes a cycle, which contradicts the acyclicness of $\vec{G}$. Thus, s is not in T. Then, adding s to T, we get the directed tree properly containing T. This contradicts that $|T| = \mathrm{MDTS}(G)$. □

We use P to denote the class of all polynomial time computable problems, and NC^k to denote the class of all problems computable by a uniform polynomial size circuit family of depth $O(\log^k n)$. The class NC is defined by $\cup NC^k$ (for further details, refer to [3, 11]). Although the P-completeness is defined via NC^1-reducibility in [3], we use the log space reducibility simply as in [15]: A function F_0 is said to be P-*complete* if F_0 is in P and for each F in P there are log space computable functions f and g such that $F(x) = g(F_0(f(x)))$ for all inputs. It is well known that the MIS problem is in NC [12, 1, 14, 7, 6, 13], and the LFMIS problem is one of the fundamental P-complete problems [15, 8].

Recall that the EREW PRAM is the parallel model where the processors operate synchronously and share a common memory, but no two of them are allowed simultaneous access to a memory cell (whether the access is for reading or for writing in that cell). The CRCW PRAM differs from the EREW PRAM in that both simultaneous reading and simultaneous writing to the same cell are allowed; in case of simultaneous writing, the processor with lowest index succeeds. It is well known [11] that a problem is in NC^{k+1} if it is solved in $O(\log^k n)$ time using a polynomial number of processors on a CRCW or EREW PRAM. As the input representation of G, to express $\vec{G}$, we assume that each

vertex v has two lists of the edges incident to it, one is the list of the arcs to v, and the other is the list of the arcs from v.

Lemma 3. **[2]** *The problem to compute* MDTS(G) *is in* NC^2.

Proof. For each vertex v, let r_v be the number of vertices reachable from v on $\vec{G}$. It is not difficult to see that MDTS(G) is equal to the maximum number of r_v for each v in G. That is, the following algorithm computes MDTS(G):

1: In parallel, compute r_v for each vertex v in G.
2: Compute $\max\{r_v \mid v \text{ in } G\}$.

The classic naive parallel algorithm can compute the first step as a kind of graph reachability problem. It computes the transitive closure of the adjacency Boolean matrix of G, and r_v is given by the number of 1s of the vth row (the details are discussed in [5, 16]). Thus, the first step is computed by using $O(\log n)$ time and n^3 processors on a CRCW PRAM. Clearly, the resources are sufficient to compute the second step. Thus, the problem is in NC^2. □

3 Lexicographically first maximal independent set problem

3.1 An algorithm for the LFMIS problem

The parallel algorithm for the LFMIS problem is very simple:

1: Set $G' := G$, and $I := \emptyset$.
2: While G' has at least one vertex, do the following step: In parallel, for each vertex v with $\mathrm{d}^-_{\vec{G'}}(v) = 0$, add v into I, and delete v and every vertex in $N_{\vec{G'}}(v)$ from G'.
3: Output I.

Lemma 4. *Let v and v' be any vertices with $\mathrm{d}^-_{\vec{G'}}(v) = \mathrm{d}^-_{\vec{G'}}(v') = 0$. Then $v \notin N_{\vec{G'}}(v')$.*

Proof. Assume that $v \in N_{\vec{G}}(v')$. Then (v', v) is the arc in G', which contradicts $\mathrm{d}^-_{\vec{G'}}(v) = 0$. □

Lemma 4 shows that every vertex v with $\mathrm{d}^-_{\vec{G'}}(v) = 0$ never fail to be added into I in step 2. That is, such a vertex is not deleted from G' as the neighbor of another vertex.

Theorem 5. *The algorithm outputs the LFMIS of G.*

Proof. We show the theorem by induction on the number of iteration. Let I_G be the LFMIS of G, and I_i be the contents of the variable I after the ith iteration. Let D_i be the set of vertices that are deleted from G' as the neighbors of the vertices in I_i in step 2. Then, it is sufficient to show that I_i is a subset of I_G,

$I_{i+1} - I_i \neq \emptyset$, and every vertex in D_i is not in I_G for each $i \geq 1$. Lemma 1 and 4 imply that I_1 is a subset of I_G, and D_1 is a set of the vertices not in I_G. We consider the ith iteration with $i > 1$. Let I' be the set of vertices v in G' with $\mathrm{d}^-_{\vec{G'}}(v) = 0$. Since $\vec{G'}$ is acyclic, $I' \neq \emptyset$, or $I_{i+1} - I_i \neq \emptyset$. Next we show that every vertex v in I' is not only in the LFMIS of G', but also in I_G. To derive a contradiction, we assume that v is a vertex in $I' - I_G$. Let $U = N_G(v) \cap I_G$. Then, by the maximality of I_G and the assumption, $U \neq \emptyset$. Let u is a vertex in U. If u is in I_{i-1}, v must be deleted from G' as a neighbor of u since $v \notin I_{i-1}$, or $u < v$. On the other hand, u is not in D_{i-1} since u is in I_G. Thus, u must be in G', or U is the set of vertices in G'. Let $J = I_G - U + \{v\}$. Clearly, J is an independent set of G, and there is a maximal independent set K such that $J \subseteq K$, consequently, $I_G - U + \{v\} \subseteq K$. For each u in U, since $\mathrm{d}^-_{\vec{G'}}(v) = 0$ and u is in G', $u > v$ holds. This implies that $K < I_G$, which contradicts that I_G is the LFMIS of G. Thus, I_i is the subset of I_G, and clearly, every vertex in D_i is not in I_G. This completes the proof. □

For the time complexity of the algorithm, we have the following theorem:

Theorem 6. *Let $\mathcal{G}$ be the family of graphs whose MDTS are bounded above by $t(n)$. Then the LFMIS problem on $\mathcal{G}$ can be solved in $O(t(n))$ time using $O(n+m)$ processors on an EREW PRAM.*

Proof. We first show the number of the iterations of the while-loop. Fix an iteration. Let T be a directed tree in G' with $|T| = \mathrm{MDTS}(G')$, and r be the root of T. Then, by Lemma 2, $\mathrm{d}^-_{\vec{G'}}(r) = 0$. Thus, r will be deleted from G' in step 2, consequently, $\mathrm{MDTS}(G')$ will decrease at least one through step 2. Since $\mathrm{MDTS}(G') = 0$ if and only if G' contains no vertices, the number of the iterations of the while-loop is bounded above by $t(n)$. The implementation of the algorithm on an EREW PRAM is easy, and omitted here. □

3.2 *P*-completeness of the LFMIS problem

It is well known that the LFMIS problem is P-complete [15, 8]. In other words, the LFMIS problem on $\mathcal{G}$ is P-complete, where $\mathcal{G}$ is the family of all graphs whose MDTS are bounded above by n. Using this observation, we have the following theorem.

Theorem 7. *The LFMIS problem on $\mathcal{G}_{\mathrm{poly}}$ is* P*-complete under the log space reductions.*

Proof. We reduce the general LFMIS problem to the restricted one. For a given graph $G(V, E)$ with n vertices, we construct a new graph $G'(V', E')$ that consists of $f(n)$ copies of G and an additional vertex, where $f(n) = \left(\frac{n+1}{c}\right)^{\frac{1}{\varepsilon}}$. The role of the additional vertex will be explained later on. Intuitively, the vertex $in+j$ in the ith copy of G corresponds to the vertex j in G for each i with $0 \leq i \leq f(n) - 1$ and j with $0 \leq j \leq n - 1$. More precisely, $V' = \{0, 1, \cdots, nf(n)\}$, and E' consists of $\{\{in + j_0, in + j_1\} \mid \{j_0, j_1\} \in E\}$ for each i with $0 \leq i \leq f(n) - 1$,

and j_0, j_1 with $0 \leq j_0, j_1 \leq n-1$. The additional vertex $nf(n)$ joints each copy of G in the following way: There exists at least one vertex v in G such that $\mathrm{d}^+_{\vec{G}}(v) = 0$ since $\vec{G}$ is acyclic. For the vertex v, E' also consists of $\{in+v, nf(n)\}$ for each i with $0 \leq i \leq f(n)-1$. Clearly, this is the log space reduction, and G' is connected. Now, $\mathrm{MDTS}(G')$ is at most $n+1$ and the number of vertices of G' is $n' = nf(n)+1$. Thus, for the resulting graph G' with n' vertices, $\mathrm{MDTS}(G') \leq n+1 = c(f(n))^\epsilon \leq c(nf(n)+1)^\epsilon = cn'^\epsilon$. This completes the proof. □

Summarizing up Lemma 3, Theorem 6 and 7, we obtain the following corollary:

Corollary 8. *1. The problem to compute MDTS on any graph family is in* NC^2.
2. The LFMIS problem on $\mathcal{G}_{\log^k n}$ *is in* NC^{k+1}, *and the problem on* $\mathcal{G}_{\mathrm{poly}}$ *is* P*-complete.*

4 Lexicographically first maximal subgraph problem for a local property

Miyano showed that the lexicographically first maximal subgraph (LFMS) problem for a hereditary property is P-complete [15], and Shoudai and Miyano showed that maximal subgraph problem for a local property is in NC [18]. Using their techniques, we can generalize the results in the previous section.

Let π be a property on graphs. The problem we consider in this section is the lexicographically first maximal subgraph (LFMS) problem for π that is stated as follows: given a graph $G = (V, E)$, find the lexicographically first maximal subset U of V whose vertex-induced subgraph satisfies π. We say that π is *nontrivial* on a graph family $\mathcal{G}$ if infinitely many graphs in $\mathcal{G}$ satisfy π and some graph in $\mathcal{G}$ violates π, and π is *hereditary* on vertex-induced subgraphs if all vertex-induced subgraphs also satisfy π. We say that a graph $G = (V, E)$ is a *minimal graph violating* π with respect to vertices if G violates π and the vertex-induced subgraph $G[U]$ satisfies π for every subset U of V with $U \neq V$.

Theorem 9. *Let* π *be a property on graphs satisfying the following conditions;*

- π *is nontrivial,*
- π *is hereditary on vertex-induced subgraphs,*
- π *is satisfied by all independent edges, and*
- π *is* NC^{s-1} *testable for some positive integer* s.

Then the LFMS problem for π *on* $\mathcal{G}_{\log^k n}$ *is in* NC^{k+s}, *and the problem on* $\mathcal{G}_{\mathrm{poly}}$ *is* P*-complete.*

Proof. First, we consider the algorithm for the LFMS problem. For subsets W and U of vertices with $W \cap U = \emptyset$, let $E_U^W = \{\{v, w\} \mid v \neq w, v, w \in W$ and there is $u \in U$ such that $\{v, u\} \in E$ and $\{w, u\} \in E\}$. Then let $H_U^W = (W, E[W] \cup E_U^W)$,

where $E[W] = \{\{u, v\} \in E \mid u, v \in W\}$. Then the algorithm is described as follows:

1. **begin** /* $G = (V, E)$ is an input */
2. $W := V$; $U := \emptyset$;
3. **while** $W \neq \emptyset$ **do begin**
4. $I := \{v \mid \mathrm{d}^-_{\vec{H}^W_U}(v) = 0\}$;
5. $U := U \cup I$;
6. $W := W - I$;
7. $W := W - \{w \in W \mid G[U \cup \{w\}]$ violates $\pi\}$;
8. **end**
9. **output** U;
10. **end**

Since π is hereditary and satisfied by all independent edges, π is satisfied by all independent set. By the definition of H^W_U, for each vertex in U, at most one of its neighbors is set into U from W in an iteration. Using the same techniques in the proof of Theorem 5 and the proof of [18, Theorem 3], we can show the correctness of the algorithm. To evaluate the number of the iterations, we use the potential function argument. The potential function $\phi(W, U)$ is defined by $\mathrm{MDTS}(H^W_U)$. Let W_i and U_i, respectively, be the contents of the variables W and U before the ith iteration. Then, it is easy to see that $\phi(W_1, U_1) = \phi(V, \emptyset) = \mathrm{MDTS}(G)$, and $\phi(W_i, U_i) > 0$ if and only if $W_i \neq \emptyset$. We now show that $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) \geq 1$ for each $i > 0$ by induction. Lemma 2 shows that $\phi(W_1, U_1) - \phi(W_2, U_2) \geq 1$, since each root of a tree T with $|T| = \mathrm{MDTS}(G)$ is set into I_1 in step 4 and deleted from W_1 in step 6. Here, fix an iteration $i > 1$ of the while-loop. We assume that $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) < 1$, and derive contradictions. Let T be a directed tree in $H^{W_{i+1}}_{U_{i+1}}$ with $|T| = \mathrm{MDTS}(H^{W_{i+1}}_{U_{i+1}})$. We first assume that T contains no edge in $E^{W_{i+1}}_{U_{i+1}}$. Then T must be the directed tree in $H^{W_i}_{U_i}$. This implies that $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) = 0$, and $|T| = \mathrm{MDTS}(H^{W_i}_{U_i})$. Thus, Lemma 2 shows that the root of T has no indegree in $H^{W_i}_{U_i}$, or the root must be set into I_i and deleted from W_i in the ith loop. This contradicts that the root of T exists in the $(i+1)$st loop. Thus, T contains at least one edge in $E^{W_{i+1}}_{U_{i+1}}$. We first assume that T contains only one edge $e = \{v, w\}$ in $E^{W_{i+1}}_{U_{i+1}}$. By construction of $E^{W_{i+1}}_{U_{i+1}}$, there is a vertex $u \in U$ such that $\{v, u\} \in E$ and $\{w, u\} \in E$. Two cases arise. First, u is set into U in the ith step. Then, T with u must be a directed tree in $H^{W_i}_{U_i}$. This contradicts the assumption that $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) < 1$. Second, u is set into U in i'th step with $i' < i$. Then, T must be a directed tree in $H^{W_i}_{U_i}$. This also contradicts the assumption. Thus, $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) \geq 1$ when T contains only one edge in $E^{W_{i+1}}_{U_{i+1}}$. Using a simple induction on the number of the edges, we can get $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) \geq 1$ when T contains two or more edges in $E^{W_{i+1}}_{U_{i+1}}$. Thus, the number of the iterations is bounded above by $\phi(W_1, U_1) = \mathrm{MDTS}(G)$. Hence, the algorithm solves the LFMS problem for π

in $O(T(n)\mathrm{MDTS}(G))$ time using polynomial number processors on an EREW PRAM for any input graph G, where $T(n)$ is the time to test whether a given graph satisfies π with polynomial number processors on an EREW PRAM. This completes the proof of the first part of the theorem.

The LFMS problem on graphs (with no restrictions) for π is P-complete under the log space reductions [15, Theorem 5]. Using the same log space reduction in Theorem 7, we can show the P-completeness of the LFMS problem for π on $\mathcal{G}_{\mathrm{poly}}$. This completes the proof of the second part of the theorem. □

5 Concluding Remarks

Toda noticed that we can use the maximum length of a directed path in $\vec{G}$ as another measure [19]. We can derive the same results using this measure as using the MDTS.

The further work is to completely characterize the class NC^k using the value measured. To do this, we must show that the LFMIS problem on the graphs with value k is not only in NC^{k+1}, but also NC^{k+1}-complete.

Acknowledgment

I indebted to Professor Seinosuke Toda and Professor Zhi-Zhong Chen for many helpful comments. My thanks also to Dr. Koichi Yamazaki for discussions and encouragement. I also wish to thank the referees for pointing out an error in an earlier draft.

References

1. N. Alon, L. Babai, and A. Itai. A Fast and Simple Randomized Parallel Algorithm for the Maximal Independent Set Problem. *Journal of Algorithms*, 7:567–583, 1986.
2. Z.-Z. Chen. Personal communication. 1997.
3. S.A. Cook. A Taxonomy of Problems with Fast Parallel Algorithms. *Information and Control*, 64:2–22, 1985.
4. K. Diks, O. Garrido, and A. Lingas. Parallel algorithms for finding maximal k-dependent sets and maximal f-matchings. In *ISA '91 Algorithms*, pages 385–395. Lecture Notes in Computer Science Vol. 557, Springer-Verlag, 1991.
5. H. Gazit and L. Miller. An Improved Parallel Algorithm That Computes the BFS Numbering of a Directed Graph. *Information Processing Letters*, 28:61–65, 1988.
6. M. Goldberg and T. Spencer. A New Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal on Computing*, 18(2):419–427, 1989.
7. M. Goldberg and T. Spencer. Constructing a Maximal Independent Set in Parallel. *SIAM J. Disc. Math.*, 2(3):322–328, 1989.
8. R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to Parallel Computation.* Oxford University Press, 1995.
9. F. Harary. *Graph Theory.* Addison-Wesley, 1972.
10. K. Iwama and C. Iwamoto. α-Connectivity: A Gradually Nonparallel Graph Problem. *Journal of Algorithms*, 20:526–544, 1996.

11. R.M. Karp and V. Ramachandran. Parallel Algorithms for Shared-Memory Machines. In J. van Leeuwen, editor, *The Handbook of Theoretical Computer Science, vol. I: Algorithms and Complexity*, pages 870–941. MIT Press, 1990.
12. R.M. Karp and A. Wigderson. A Fast Parallel Algorithm for the Maximal Independent Set Problem. *Journal of American Computing Machinery*, 32(4):762–773, 1985.
13. D.C. Kozen. *The Design and Analysis of Algorithms*. Springer-Verlag, 1992.
14. M. Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal on Computing*, 15(4):1036–1053, 1986.
15. S. Miyano. The Lexicographically First Maximal Subgraph Problems: P-Completeness and NC Algorithms. *Mathematical Systems Theory*, 22:47–73, 1989.
16. C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.
17. D. Pearson and V.V. Vazirani. Efficient Sequential and Parallel Algorithms for Maximal Bipartite Sets. *Journal of Algorithms*, 14:171–179, 1993.
18. T. Shoudai and S. Miyano. Using Maximal Independent Sets to Solve Problems in Parallel. *Theoretical Computer Science*, 148:57–65, 1995.
19. S. Toda. Personal communication. 1997.

This article was processed using the LaTeX macro package with LLNCS style