Parallel Complexity of the Lexicographically First Maximal Subgraph Problems on Restricted Graphs

Ryuhei Uehara

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

Abstract

Longest directed path length (LDPL) is defined by the length of the longest path of the directed acyclic graph of a given undirected graph. The LDPL of a graph *measures* the parallelization for the lexicographically first maximal subgraph (LFMS) problems. That is, the parallel complexity of the problems gradually increases as the value measured on a graph grows; the LFMS problems are in NC on graphs of LDPL $O(\log^k n)$, and P-complete on graphs of LDPL cn^ϵ. It is worth remarking that the problem to compute the LDPL is in NC^2. The LDPL on a random graph is also considered, and the threshold value for the LDPL l is $\frac{l}{n}$ for a random graph $G(n,p)$.
Keywords: Analysis of algorithms, NC algorithms, P-completeness, random graphs, 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. Karp and Wigderson first showed that the MIS problem is in the class NC [15]. Since then, much work has been devoted to the study of parallel complexity of maximality problems [17, 1, 9, 8, 7, 20, 21]. On the other hand, the lexicographically first maximal independent set (LFMIS) problem is a typical P-complete problem [5], and P-completeness of the lexicographically first maximal subgraph (LFMS) problems for some properties π was shown [18, 20] (see also [10] for a comprehensive reference). However, interestingly, the LFMIS problem on a random graph can be efficiently solved in parallel [6].

As noticed by Iwama and Iwamoto in [12], one of the approaches to making clear the boundary between the classes NC and P is to find (sufficient) conditions for problems to be in NC or to be P-complete. In [12], 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 graph gradually increases as the value measured on the graph grows. The measure is the *longest directed path length* (LDPL) of G, which is defined by the length of the longest directed path on the directed acyclic graph of G. The problem to compute the LDPL is in NC^3. This result is important since a "measure" is valid only if measuring the complexity of a problem is easier than solving the problem. The LFMIS problem on a graph of LDPL $O(\log^k n)$ is in NC^{k+1}, and the problem on a graph of LDPL cn^ϵ is P-complete, for any fixed positive constants c and ϵ. 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 a graph of LDPL $O(\log^k n)$ is in NC^{k+s}, and the problem on a graph of LDPL cn^ϵ is P-complete, for any fixed positive constants c and ϵ.

We next consider the average value of the LDPL on a random graph. (For the detail of a random graph, see [3, 2, 4] for example.) It seems to be similar to the graph property that contains a path of length l (see, e.g. [4, Figure 13.1]). However, interestingly, these properties are completely different, and so their threshold values are. Let $Q(l)$ be the graph property that the LDPL of the graph is at least l. For a random graph $G(n,p)$, the threshold value for the property $Q(l)$ is $\frac{l}{n}$ when $\lim_{n\to\infty} l = \infty$. (Cf. The threshold value of the graph property that a graph contains a path of length l is $n^{-(l+1)/l}$ [4, Figure 13.1].) The threshold value implies that the LDPL of a random graph $G(n,p)$ is $\Theta(np)$ with high

probability. Consequently, the LDPL of $G(n,p)$ is $\Theta(n)$ with high probability for any fixed positive real number $0 < p < 1$; and we must set $p = O(\frac{\log^k n}{n})$ for some positive constant k to be the LDPL of $G(n,p)$ is $O(\log^k n)$ with high probability.

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, $V = \{1, 2, \cdots, n\}$ 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 $1 \leq i < j \leq k, h$.) 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 $1 \leq i \leq k$, or there is an index $i' \geq 1$ such that $x_j = y_j$ for every $1 \leq j < i'$, 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 (DAG) 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 DAG $\vec{G}$ 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 [11, Theorem 16.2]). Especially, $\mathrm{d}^-{}_{\vec{G}}(1) = 0$ and $\mathrm{d}^+{}_{\vec{G}}(n) = 0$ for any DAG $\vec{G}$.

Lemma 1 **[6]** *Let v be a vertex with $\mathrm{d}^-{}_{\vec{G}}(v) = 0$. Then v is in the LFMIS of G.*

Now, we define the measure of the parallelization for the lexicographically first maximal problems. For $v_1, v_2, \cdots, v_k \in V$, $\alpha = (v_1, v_2, \cdots, v_k)$ is a *directed path* in $\vec{G} = (V, \vec{E})$ if $(v_i, v_{i+1}) \in \vec{E}$ for each $1 \leq i < k$. The *length* of α, denoted by $|\alpha|$, is defined by the number of edges in α. (In this case, $|\alpha| = k - 1$.) The *longest directed path length* (LDPL) of G, denoted by LDPL(G), is defined by the maximum of the length of a directed path in $\vec{G}$.

Lemma 2 *Let $\alpha = (v_1, v_2, \cdots, v_k)$ be a directed path in $\vec{G}$ with $|\alpha| = \mathrm{LDPL}(G)$. Then $\mathrm{d}^-{}_{\vec{G}}(v_1) = 0$.*

Proof. To derive contradictions, assume that $\mathrm{d}^-{}_{\vec{G}}(v_1) > 0$. Then there is a vertex u with (u, v_1) in $\vec{G}$. First we assume that u is in α. Then the directed path from v_1 to u with the arc (u, v_1) makes a cycle, which contradicts the acyclicness of $\vec{G}$. Thus, u is not in α. Then, adding u to α, we get a directed path longer than α. This contradicts that $|\alpha| = \mathrm{LDPL}(G)$. ∎

We use P to denote the class of all polynomial time computable problems, $NLOG$ to denote the class of all problems computable by a nondeterministic Turing machine using $O(\log^k n)$ space, and NC^k to denote the class of all problems computable by a uniform polynomial size circuit family of depth $O(\log^k n)$, where n is the size of its input, and k is some positive constant. The class NC is defined by $\cup NC^k$ (for further details, refer to [5, 14]). Although the P-completeness is defined via NC^1-reducibility in [5], we use the log space reducibility simply as in [18]: A problem 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 [15, 1, 17, 9, 8, 16], and the LFMIS problem is one of the fundamental P-complete problems [5, 18, 10].

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 [14] 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 [23] *The problem to compute* LDPL(G) *is in* NC^2.

Proof. We first consider two languages L_1 and L_2 defined as follows; $L_1^k = \{< G, k > |\ G$ has a path of length at least $k\}$, and $L_2^k = \{< G, k > |$ all the paths in G are of length at most $k\}$. It is not difficult to see that L_1 is in $NLOG$ and L_2 is in co-$NLOG$ or in $NLOG$ since co-$NLOG = NLOG$. Thus, the language $L^k = \{< G, k > |\ \mathrm{LDPL}(G) = k\} = L_1 \cap L_2$ is in $NLOG$ or in NC^2 since $NLOG \in NC^2$ (see e.g. [13]). For each i with $1 \leq i \leq n$, we can make n parallel queries to compute whether $< G, i >$ is in L^k. This establishes the lemma. ∎

Let n be a positive integer, and p be a positive real number with $0 < p < 1$ (and p may depend on n). The random graph $G(n,p)$ is a probability space over the set of graphs on the vertex set $\{1, 2, \cdots, n\}$ determined by $\Pr[\{i,j\} \in G] = p$, with these events mutually independent. Let Q be a graph theoretic property that is not destroyed by the addition of edges to a graph. A function $r(n)$ is called a *threshold function for* Q if $\lim_{n\to\infty} p(n)/r(n) = 0$ implies that almost no graph has property Q, and $\lim_{n\to\infty} p(n)/r(n) = \infty$ implies that almost every graph has property Q. In this paper, we concentrate the value of the LDPL on a random graph, so that, $Q(l)$ is defined by the graph property that the LDPL of the graph is at least l. We assume that $l \leq n$ without loss of generality.

3 Lexicographically first maximal independent set problem

3.1 An algorithm for the LFMIS problem

The parallel algorithm for the LFMIS problem is the following:

1: Set $G' := G$, and $I := \emptyset$.

2: While G' has at least one vertex, do: 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.

Theorem 4 [6] *The algorithm outputs the LFMIS of* G.

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

Theorem 5 *The LFMIS problem on a graph whose LDPL is bounded above by* $t(n)$ *can be solved in* $O(t(n))$ *time using* $O(n+m)$ *processors on an EREW PRAM.*

Proof. Fix an iteration. Let α be a directed path in $\vec{G'}$ with $|\alpha| = \mathrm{LDPL}(G')$, and v be the least vertex of α. Then, by Lemma 2, $\mathrm{d}^-_{\vec{G'}}(v) = 0$. Thus, v will be deleted from G' in step 2, consequently, LDPL(G') will decrease by at least one through step 2. It is easy to see that LDPL(G') $= 0$ if and only if G' contains no edges. When G' has no edges, all vertices in $\vec{G'}$ has no indegree, so that they will be deleted in step 2 of the algorithm. Thus, the number of the iterations of the while-loop is bounded above by $t(n) + 1$. 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 [18, 10]. In other words, the LFMIS problem on a graph of LDPL $n-1$ is P-complete. Using this observation, we have the following theorem.

Theorem 6 *The LFMIS problem on a graph of LDPL* cn^ϵ *is* P*-complete under* log *space reductions for any fixed positive constants* c *and* ϵ.

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 $f(n) = \left\lceil \left(\frac{n+1}{c}\right)^{\frac{1}{\epsilon}} \right\rceil$. 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 $nf(n)$ joins each copy of G in the following way: We remind that the vertex $n - 1$ in G has no outdegree, or $\mathrm{d}^+_{\vec{G}}(n-1) = 0$. For the vertex $n - 1$, E' also contains edges $\{in + n - 1, nf(n)\}$ for each i with $0 \leq i \leq f(n) - 1$. Clearly, this is a log space reduction, and G' is connected. Now, LDPL(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, LDPL(G') $\leq n + 1 = c(f(n))^\epsilon \leq c(nf(n) + 1)^\epsilon = cn'^\epsilon$. This completes the proof. ∎

4 Lexicographically first maximal subgraph problem for a local property

The P-completeness of the lexicographically first maximal subgraph (LFMS) problem for a hereditary property is proved by Miyano [18], and Shoudai and Miyano showed that the maximal subgraph problem for a local property is in NC [21]. 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 $U \neq V$. A set of edges is a *matching* if no pair of the edges in the set shares their endpoints.

Theorem 7 *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 matchings, and*
- *π is NC^{s-1} testable for some positive integer s.*

Then the LFMS problem for π on a graph of LDPL $O(\log^k n)$ is in NC^{k+s}, and the problem on a graph of LDPL cn^ϵ is P-complete, for any fixed positive constants c and ϵ.

Proof. First, we consider an algorithm for the LFMS problem. For subsets W and U of vertices with $W \cap U = \emptyset$, let $E^W_U = \{\{v, w\} \mid v, w \in W, v \neq w$, and there is $u \in U$ such that $\{v, u\} \in E$ and $\{w, u\} \in E\}$. That is, for each vertex u in U, all neighbors in W are joined to each other by the edges in E^W_U. Then let $H^W_U = (W, E[W] \cup E^W_U)$, where $E[W] = \{\{u, v\} \in E \mid u, v \in W\}$. Intuitively speaking, we want to find an independent set in $G[W]$ such that for each vertex u in U at most one neighbor of u is in the independent set. 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 sets. By the definition of H_U^W, for each vertex in U, at most one of its neighbors is set into U from W in an iteration. The algorithm in the proof of [21, Theorem 3] is the same as our algorithm except step 4. In step 4, the algorithm in the proof of [21, Theorem 3] finds an independent set. Thus, using the same techniques in the proof of [21, Theorem 3] together with the fact that the set $\{v \mid \mathrm{d}^-_{\vec{H}_U^W}(v) = 0\}$ is an independent set, we can show the correctness of the algorithm. Now, we show that LDPL(H_U^W) is decreased by at least one after one iteration. Hereafter, we use $\phi(W, U)$ to denote LDPL(H_U^W) for ease to read. 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 $\phi(W_1, U_1) = \phi(V, \emptyset) = \mathrm{LDPL}(G)$, 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. For each directed path α with $|\alpha| = \mathrm{LDPL}(G)$, its least vertex is set into I_1 in step 4 and deleted from W_1 in step 6 from Lemma 2. Thus, $\phi(W_1, U_1) - \phi(W_2, U_2) \geq 1$. 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 α be a directed path in $\vec{H}_{U_{i+1}}^{W_{i+1}}$ with $|\alpha| = \mathrm{LDPL}(H_{U_{i+1}}^{W_{i+1}})$. We first assume that α contains no arc in $\vec{E}_{U_{i+1}}^{W_{i+1}}$. Then α must be a directed path in $\vec{H}_{U_i}^{W_i}$. This implies that $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) = 0$, and $|\alpha| = \mathrm{LDPL}(H_{U_i}^{W_i})$. Thus, Lemma 2 shows that the least vertex in α has no indegree in $\vec{H}_{U_i}^{W_i}$, or the vertex must be set into I_i and deleted from W_i in the ith loop. This contradicts that the vertex exists in the $(i+1)$st loop. Thus, α contains at least one arc in $\vec{E}_{U_{i+1}}^{W_{i+1}}$. We first assume that α contains only one arc $a = (v, w)$ in $\vec{E}_{U_{i+1}}^{W_{i+1}}$. By construction of $E_{U_{i+1}}^{W_{i+1}}$, there is a vertex $u \in U_{i+1}$ such that $\{v, u\} \in E$ and $\{w, u\} \in E$. Moreover, since u was set into U in step 4, $(u, v), (u, w) \in \vec{E}$, or $u < v, w$. Two cases arise. First, u is set into U in the ith iteration. Then α with u must be a directed path in $\vec{H}_{U_i}^{W_i}$. 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 iteration with $i' < i$. Then α must be a directed path in $\vec{H}_{U_i}^{W_i}$. This also contradicts that α exists in the $(i+1)$st loop. Thus $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) \geq 1$ when α contains only one arc in $\vec{E}_{U_{i+1}}^{W_{i+1}}$. Using a simple induction on the number of the arcs, we can get $\phi(W_i, U_i) - \phi(W_{i+1}, U_{i+1}) \geq 1$ when α contains two or more arcs in $\vec{E}_{U_{i+1}}^{W_{i+1}}$. Thus, the number of the iterations is bounded above by $\phi(W_1, U_1) = \mathrm{LDPL}(G)$. Hence, the algorithm solves the LFMS problem for π in $O(T(n)\mathrm{LDPL}(G))$ 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 former half of the theorem.

The LFMS problem on graphs (with no restrictions) for π is P-complete under log space reductions as [18, Theorem 5]. Using the same log space reduction in Theorem 6, we can show the P-completeness of the LFMS problem for π on a graph of LDPL cn^ϵ. This completes the proof of the latter half of the theorem. ∎

5 LDPL on random graphs

In this section, we consider the average value of the LDPL on a random graph. That is, we consider the threshold value of the graph property $Q(l)$, that the LDPL of the graph is at least l. We only consider the case that l is an increasing function of n. It seems to be similar to the graph property that contains a path of length l (see, e.g. [4, Figure 13.1]). However, interestingly, these properties are completely different, and so their threshold values are. The reason is, intuitively, stated as follows: Let K_n be the complete graph with n vertices. Then the number of the *undirected* (vertex-disjoint) paths of length l in K_n is, clearly, $\binom{n}{l+1} l!$. On the other hand, the number of the *directed* paths on $\vec{K}_n$ is only $\binom{n}{l}$, since a path using specific l vertices is uniquely determined following their order on $\vec{K}_n$. This observation together with linearity of expectation implies the following proposition for a random graph $G(n, p)$.

Proposition 8 *Let $N(n,p,l)$ be the expected value of the number of the directed paths of length l on $\vec{G}(n,p)$ with $l \geq 1$. Then $N(n,p,l) = \binom{n}{l+1}p^l$.*

Let l_0 be the least integer that satisfies $N(n,p,l_0) \leq 1$. Intuitively speaking, l_0 gives the LDPL on $G(n,p)$ on the average case. Using the equation $\left(\frac{n}{l}\right)^l \leq \binom{n}{l}$ (see e.g. [19]), we get $1 \geq \binom{n}{l_0+1}p^{l_0} \geq \left(\frac{n}{l_0+1}p\right)^l$, consequently, $np - 1 \leq l_0$. On the other hand, using the equation $\binom{n}{l} \leq \left(\frac{en}{l}\right)^l$ and $N(n,p,l_0-1) > 1$, we also get $1 < \binom{n}{l_0}p^{l_0-1} \leq \left(\frac{en}{l_0}\right)^{l_0} p^{l_0-1}$, which is less than $\left(\frac{en}{l_0-1}\right)^{l_0-1} p^{l_0-1}$ for sufficiently large n (or l_0). This implies that $enp + 1 \leq l_0$. Intuitively speaking, the LDPL on $G(n,p)$ is nearly equal to np on the average case. We show that the average case occurs with high probability.

Lemma 9 *Let d be any positive integer with $d + l_0 \leq n$. Then $N(n,p,d+l_0) \leq (1-ep)^d \left(\frac{l_0+1}{l_0+2}\right)^d$.*

Proof. Using $\binom{n}{l_0}p^{l_0} \leq 1$ and $l_0 \geq np - 1$, we have

$$
\begin{aligned}
& N(n,p,l_0+d) \\
&= \binom{n}{l_0+d+1}p^{l_0+d} \\
&= \frac{n-l_0-1}{l_0+2}\frac{n-l_0-2}{l_0+3}\cdots\frac{n-l_0-d}{l_0+d+1}\binom{n}{l_0+1}p^{l_0}p^d \\
&\leq \frac{n-l_0-1}{l_0+2}\frac{n-l_0-2}{l_0+3}\cdots\frac{n-l_0-d}{l_0+d+1}\left(\frac{l_0+1}{n}\right)^d \\
&= \frac{n-l_0-1}{n}\frac{n-l_0-2}{n}\cdots\frac{n-l_0-d}{n} \\
&\quad \frac{l_0+1}{l_0+2}\frac{l_0+1}{l_0+3}\cdots\frac{l_0+1}{l_0+d+1} \\
&\leq \left(\frac{n-l_0-1}{n}\right)^d\left(\frac{l_0+1}{l_0+2}\right)^d \leq (1-ep)^d\left(\frac{l_0+1}{l_0+2}\right)^d.
\end{aligned}
$$

∎

Lemma 10 *We assume that $\frac{n}{p} = o(n^2)$. Let c_1 be any fixed real number with $0 < c_1 < 1$, and l_1 be $c_1 np$. Let ϵ be any given real number with $0 < \epsilon < 1$. Then for sufficiently large n, the LDPL of $G(n,p)$ is at least l_1 with probability greater than $1-\epsilon$.*

Proof. We consider the following algorithm:

1: Set $v := 1$;

2: Find the least vertex u such that $u > v$ and (v,u) is an arc in $\vec{G}(n,p)$;

3: If u does not exist or the length of the directed path produced by the computed vertices is l_1, halt;

4: Set $v := u$, and go to step 2.

We show that the algorithm finds a directed path of length l_1 with high probability. This implies that the LDPL of $G(n,p)$ is at least l_1. Let X be a random value defined to be the number of trials required to check whether (v,u) is an arc in $\vec{G}(n,p)$ in step 2. Let $C_1, C_2, \cdots, C_X$ denote the sequence of trials, where $C_i \in \{true, false\}$ denotes whether (v,u) is an arc in $\vec{G}(n,p)$ or not. Call the ith trial C_i a *success* if the arc (v,u) is in $\vec{G}(n,p)$. Later on, we will show that u does not exist in step 3 with very few probability. Here we consider the case that the algorithm halts because it finds a directed path of length l_1. In this case, C_X is always success. We divide the sequence into *epochs* as follows; epoch 1 begins with C_1 and ends with the first success; and epoch i with $i > 1$ begins with the trial following the ith success and ends with the trial on which we obtain the $(i+1)$st success. Define the random variable X_i, for $1 \leq i \leq l_1$, to be the number of trials in the ith epoch, so that $X = \sum_{i=1}^{l_1} X_i$. The random variable X_i is geometrically distributed with parameter p since the probability of success on any trial is p from the definition of a

random graph $G(n,p)$. Thus, the expected value of X_i is $\frac{1}{p}$, and its variance is $\frac{1-p}{p^2}$. Hence, the expected value of X is $\frac{l_1}{p} = c_1 n$, and its variance is $\frac{1-p}{p} c_1 n$. Thus, using Chebyshev's Inequality, we get

$$\Pr\left[\, |X - c_1 n| \geq \sqrt{\frac{(1-p)c_1 n}{(1-\epsilon)p}} \right] \leq 1 - \epsilon.$$

Here $\sqrt{\frac{(1-p)c_1 n}{(1-\epsilon)p}} < \sqrt{\frac{c_1}{1-\epsilon}}\sqrt{\frac{n}{p}} = o(n)$. Hence $c_1 n + \sqrt{\frac{(1-p)c_1 n}{(1-\epsilon)p}} < n$ for sufficiently large n. Consequently, the algorithm success to find a path of length l_1 with probability greater than $1-\epsilon$ for sufficiently large n. This completes the proof. ∎

We remind that $Q(l)$ is the graph property that the LDPL of a given graph is at least l.

Lemma 11 *For a given random graph $G(n,p)$ and a positive integer l, suppose $N(n,p,l) < 1$. Then $Q(l)$ is satisfied with probability at least $N(n,p,l)$.*

Proof. For any positive integer l' with $l' > l$, every directed path of length l' contains at least one directed path of length l. Thus, when $Q(l')$ is satisfied, so $Q(l)$ is. That is, when $Q(l)$ is not satisfied, $Q(l')$ is not satisfied, either. This implies that $\cup_{i \geq l} \Pr[Q(i) \text{ is satisfied}] = \Pr[Q(l) \text{ is satisfied}]$. Hence, the basic facts from statistics show that the probability that the graph has at least one directed path of length l is bounded above by the expected number of the directed path of length l. This completes the proof. ∎

Now we are ready to show the threshold function of $Q(l)$.

Theorem 12 *For $Q(l)$ with $l \geq 1$, $r(n,l) = \frac{l}{n}$ is the threshold function.*

Proof. Let l_2 be the least integer that satisfies $N(n,r,l_2) \leq 1$ on a random graph $G(n,r)$. Then $enr + 1 \geq l_2 \geq nr - 1$.

We then consider a random graph $G(n,p)$ such that $\lim_{n\to\infty} p(n,l)/r(n,l) = \infty$. Then for any positive fixed constant c_3 with $0 < c_3 < 1$, $c_3 np > enr + 1 \geq l_2$ for sufficiently large n. This implies that $\frac{n}{p} < \frac{n^2}{l_2} = \frac{n^2}{c_2 l} = o(n^2)$. Thus, from Lemma 10, the LDPL of $G(n,p)$ is at least l_2 with probability greater than $1-\epsilon$ for any fixed positive real number ϵ.

Next assume that $\lim_{n\to\infty} p(n,l)/r(n,l) = 0$. Let l_3 be the least integer that satisfies $N(n,p,l_3) \leq 1$ on a random graph $G(n,p)$. Then $enp+1 \geq l_3 \geq np-1$. For any fixed positive integer d, $(1+d)l_3 = (1+d)(np-1) < nr - 1 \leq l_2 < n$ for sufficiently large n. From Lemma 9, $N(n,p,l_3+dl_3) \leq (1-ep)^{dl_3}\left(\frac{l_3+1}{l_3+2}\right)^{dl_3}$. When p is a positive constant, $N(n,p,l_3+dl_3)$ converges to 0 since $N(n,p,l_3+dl_3) \leq (1-ep)^{dl_3}$. When $p = o(1)$, consequently, $l_3 = o(n)$, we have $N(n,p,l_3+dl_3) \leq \left(\frac{l_3+1}{l_3+2}\right)^{dl_3} = \frac{\left(1-\frac{1}{l_3+2}\right)^{d(l_3+1)}}{\leq} \left(\frac{l_3+2}{l_3+1}\frac{1}{e}\right)^d = \left(\left(1+\frac{1}{l_3+1}\right)\frac{1}{e}\right)^d$, which also converges to 0 for large d and n (or l_3). This together with Lemma 11 implies that the probability that the LDPL of $G(n,p)$ is at least l_3 converges to 0. Since $l_3 < l_2$, the LDPL of $G(n,p)$ is at least l_2 with probability less than ϵ for any fixed positive real number ϵ. ∎

6 Concluding Remarks

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 thank Professor Osamu Watanabe, who noticed an error for the parallel complexity of the LDPL problem in an early draft. I indebted to Professor Seinosuke Toda and Professor Zhi-Zhong Chen for their many helpful comments. In a preliminary version of this paper [22], I defined the maximum directed tree size (MDTS) and used it as a measure. The LDPL is better than the MDTS because the graph family characterized by the MDTS is properly contained in the graph family characterized by the LDPL. Professor Seinosuke Toda is the first man who noticed that we can use the LDPL as another measure. Professor Zhi-Zhong Chen gave many fruitful suggestion for the complexity of computing the measure. Especially, he noticed that the problem computing the MDTS is in NC^2. My thanks also to Dr. Koichi Yamazaki for discussions and encouragement.

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] N. Alon and J.H. Spencer. *The Probabilistic Method*. John Wiley & Sons, 1992.

[3] B. Bollobás. *Random Graphs*. Academic Press, 1985.

[4] G. Chartrand and L. Lesniak. *Graphs & Digraphs*. Chapman & Hall, 3rd edition, 1996.

[5] S.A. Cook. A Taxonomy of Problems with Fast Parallel Algorithms. *Information and Control*, 64:2–22, 1985.

[6] D. Coppersmith, P. Raghavan, and M. Tompa. Parallel Graph Algorithms That Are Efficient on Average. *Information and Computation*, 81:318–333, 1989.

[7] 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.

[8] M. Goldberg and T. Spencer. A New Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal on Computing*, 18(2):419–427, 1989.

[9] M. Goldberg and T. Spencer. Constructing a Maximal Independent Set in Parallel. *SIAM J. Disc. Math.*, 2(3):322–328, 1989.

[10] R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to Parallel Computation*. Oxford University Press, 1995.

[11] F. Harary. *Graph Theory*. Addison-Wesley, 1972.

[12] K. Iwama and C. Iwamoto. α-Connectivity: A Gradually Nonparallel Graph Problem. *Journal of Algorithms*, 20:526–544, 1996.

[13] D.S. Johnson. A Catalog of Complexity Classes. In J. van Leeuwen, editor, *The Handbook of Theoretical Computer Science, Vol. I: Algorithms and Complexity*, pages 69–161. Elsevier, 1990.

[14] 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. Elsevier, 1990.

[15] R.M. Karp and A. Wigderson. A Fast Parallel Algorithm for the Maximal Independent Set Problem. *Journal of the ACM*, 32(4):762–773, 1985.

[16] D.C. Kozen. *The Design and Analysis of Algorithms*. Springer-Verlag, 1992.

[17] M. Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal on Computing*, 15(4):1036–1053, 1986.

[18] S. Miyano. The Lexicographically First Maximal Subgraph Problems: P-Completeness and NC Algorithms. *Mathematical Systems Theory*, 22:47–73, 1989.

[19] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge, 1995. 1995.

[20] D. Pearson and V.V. Vazirani. Efficient Sequential and Parallel Algorithms for Maximal Bipartite Sets. *Journal of Algorithms*, 14:171–179, 1993.

[21] T. Shoudai and S. Miyano. Using Maximal Independent Sets to Solve Problems in Parallel. *Theoretical Computer Science*, 148:57–65, 1995.

[22] R. Uehara. A Measure of Parallelization for the Lexicographically First Maximal Subgraph Problem. In *23rd International Workshop on Graph-Theoretic Concepts in Computer Science (WG '97)*. Springer-Verlag, 1997. to appear.

[23] N.V. Vinodchandran. Personal communication. 1997.