ON-LINE MAXIMUM INDEPENDENT SET IN CHORDAL GRAPHS

George CHRISTODOULOU *, Vassilis ZISSIMOPOULOS *

Abstract. In this paper we deal with the on-line maximum independent set and we propose a probabilistic $O(\log n)$-competitive algorithm for chordal and interval graphs, proving that the same ratio is a lower bound of the problem. The relation of the on-line maximum independent set with the on-line admission control, allows us to obtain as particular case, an $O(\log n)$-competitive algorithm for the on-line admission control in trees and lines. In addition to that, we propose a competitive algorithm for the on-line call admission of subtrees in trees.

1 Introduction

The Maximum Independent Set problem (MIS) is one of the most fundamental problems in graph theory. Given a graph $G = (V, E)$ with vertex set V and edge set E, the goal is to compute a subset of the vertices V', such that no two vertices in V' are joined by an edge and such that the cardinality of V' is maximized. In this paper we deal with an on-line version of MIS, where the graph G is not known in advance, but is revealed in an on-line manner by a malicious adversary to the on-line algorithm. The on-line algorithm has to take its decisions during this revealing procedure.

*Partially supported by the Special Research Grants Account of the University of Athens under Grant 70/4/5821

*Department of Informatics and Telecommmunications, University of Athens, 157 84 Athens, Greece, {gchristo,vassilis}@di.uoa.gr

There are many on-line graph models in the bibliography. In [9] G is not given in advance to the on-line algorithm. There is a set of rules R dealing with information about the value of some parameters of the final graph G (e.g. the maximum degree of G) and the manner in which G is revealed to the on-line algorithm (e.g. vertex by vertex). The adversary may terminate the revealing procedure at any time. In [23] the same model is applied, but for the on-line graph coloring problem. In the same work there are some variations of the model, which do not affect essentially the problem. In [21] an on-line model called *Known-Graph On-line Model* is defined. According to this model, a graph isomorphic to G is induced, but the identification of the vertices is not known. Finally in [22] two more on-line models are proposed (*multi-solutions model* and *inheritance model*). According to these, the algorithm can preserve a collection of independent sets. In each step, the current vertex can be added to up to a number of different sets.

Another interesting problem that arises in a number of applications in communication networks, is the *on-line admission control problem*. In this work, we focus on particular topologies, namely trees and lines. Given a tree (or a line) T, call requests (given by the endpoints of the path) to be satisfied are presented in an on-line manner to the on-line algorithm. The algorithm may accept or reject a request and the goal is to maximize the number of the accepted requests. In [4], [8] a $O(\log n)$-competitive randomized algorithm is presented for on-line admission control in trees and lines, where n is the order of T. In [5] a $O(\log d)$-competitive randomized algorithm is suggested, where d is the diameter of T.

Here, we propose a randomized $O(\log n)$-competitive algorithm for the on-line MIS in chordal graphs and we prove that the same ratio is a lower bound of the problem. We also relate the on-line admission control in trees to the on-line MIS in chordal graphs and achieve an $O(\log n)$ competitive ratio, where n is the order of the tree. In section 2 we present the on-line model that we consider for MIS. In section 3 we refer some preliminary notions from graph theory concerning chordal and interval graphs. Then, in section 4 we present the randomized algorithm for the on-line MIS in chordal graphs. Last, in section 5 we adapt the algorithm to the on-line admission control in trees.

2 The Model

We consider the following on-line model (see also [6]). The graph that is presented by the adversary to the on-line algorithm, is an induced subgraph of a graph that *is known in advance* to the on-line algorithm. The interesting part of this model is that it is parallel with the on-line admission control model, where the network is given in advance to the on-line algorithm.

More formally [6]:

Definition 1. *The graph $G = (V, E)$ is known to the on-line algorithm. The vertices $v \in V$ are presented one by one. The adversary may choose to terminate the sequence at any time. The on-line algorithm has to decide if he accepts v or rejects it. The benefit of the on-line algorithm is the cardinality of the accepted vertex set.*

As a performance measure for our algorithm, we will use *competitive analysis* [30]. The competitive ratio of an on-line MIS algorithm is the maximum over all sequences of vertices of the ratio of the optimal algorithm for a sequence of vertices to the performance of the on-line algorithm on the same sequence. Specifically, let $\text{ALG}(\sigma)$ be the cardinality of the accepted vertex set by the on-line algorithm, for a sequence of vertices σ and let $\text{OPT}(\sigma)$ be the cardinality of the accepted vertex set by an optimal offline algorithm for σ. The competitive ratio of ALG is the maximum over all σ of $\text{OPT}(\sigma)/\text{ALG}(\sigma)$. For the case of randomized on-line algorithms, let $\mathbf{E}[\text{ALG}(\sigma)]$ be the expected cardinality of the accepted vertex set by the ALG on a sequence σ. The competitive ratio of ALG is the maximum over all σ of $\text{OPT}(\sigma)/\mathbf{E}[\text{ALG}(\sigma)]$. This competitive ratio is called *oblivious* since the sequence σ is produced by the adversary independently of the random choices made by ALG. In this paper we consider an oblivious adversary.

3 Chordal Graphs

In this section we present some preliminary definitions and results that we will use later.

A *chord* is an edge that joins two non consecutive vertices of a cycle.

Definition 2. *A graph is chordal if each one of its cycles with length $l > 3$ has a chord.*

Every *induced subgraph* of a chordal graph is also a chordal graph.

For the rest of the paper we use G to denote a simple undirected graph, $V(G)$ and $E(G)$ to denote respectively the vertex set and the edge set of G, the cardinality of which is $n = |V(G)|$ and $m = |E(G)|$. A *clique* is a vertex set that induces a complete subgraph of G. A clique is *maximal*, if it is not a subclique of some other clique of G.

Definition 3. *Let F be a family of nonempty sets. The intersection graph of F is a graph the vertices of which correspond to the sets of F, while two vertices are adjacent if and only if the corresponding subsets intersect.*

An interesting subfamily of chordal graphs are the *interval graphs.*

Definition 4. *An interval graph is an intersection graph of a family of intervals of the real line.*

Theorem 1. *[19] Let G be an undirected graph and let $\mathcal{K}$ be the set of its maximal cliques and $\mathcal{K}_v$ the set of all maximal cliques that contain a vertex v of G. The following statements are equivalent:*

1. *G is a chordal graph.*
2. *G is the intersection graph of a family of subtrees of a tree.*
3. *There is a tree $T = (\mathcal{K}, \mathcal{E})$ the vertex set of which is the set of maximal cliques of G, such that each induced subgraph $T[\mathcal{K}_v]$ is connected.*

A tree that satisfies the third property of theorem 1 is called *clique tree* of G.

Definition 5. *The clique graph $K(G)$ of a chordal graph G is the intersection graph of maximal cliques of G.*

One may consider weights w_e on the edges $e \in E(K(G))$, such that $w_{u,v} = |u \cap v|$, where $u, v \in V(K(G))$ are the ends of an edge e of $K(G)$. We denote the weighted clique graph of G by $K_w(G)$

Theorem 2. *[7] The clique tree of a chordal graph G is a maximum weight spanning tree of the weighted clique graph $K_w(G)$.*

In [19] it is proved the *Clique Intersection Property* which is the following:

Theorem 3. *[19] For any pair of cliques $K, K' \in K(G)$, the set $K \cap K'$ is contained in every clique of the path (in the clique tree) with endpoints K and K' if and only if G is chordal.*

The clique tree of an interval graph has the following interesting property

Theorem 4. *[20] G is an interval graph if and only if G has a clique tree that is a simple path.*

4 The Algorithm

In the sequel, we focus on the on-line maximum independent set in chordal graphs. We make use of the on-line model mentioned in section 2.

The initial graph G presented to the on-line algorithm is chordal, as it is the final graph C, which is an induced subgraph of G. In the following we will use the term "vertex" when referring to a vertex of the chordal graph G and the term "node" when referring to a vertex of the clique tree $T(G)$ induced by G. This distinction is meaningful as a node $u \in T(G)$ denotes a clique of G.

The algorithm can be separated into two phases. The first is the vertex partitioning phase and the second is the selection phase.

Vertex Partitioning The aim of this phase is to classify the vertex set $V(G)$ into $O(\log n)$ disjoint classes, where $n = |V(G)|$. At the first step of VertexPartitioning, we construct a clique tree of G. This process takes linear time, given a perfect elimination order of G [31]. A perfect elimination ordering of G can be found in linear time [28], [32]. Then, VertexPartitioning procedure uses NodePartitioning procedure in order to realize the classification. Notice that a node u (line 4 of NodePartitioning procedure) always exists, because trees have 1/2-separators of length equal to 1 [33]. The removal of u from T, induces a forest, each subtree of which has at most $\frac{|V(T(G))|}{2}$ nodes.

At the end of this phase every vertex $v \in V(G)$ is marked with a label that denotes its class. The number of distinct labels is k with $k \leq \lceil \log |V(T(G))| \rceil$. The maximal cliques of G are at most n [18], where n is the order of the initial chordal graph G. Hence, in respect with the theorems 1 and 3, we have separated the vertices of G in at most $O(\log n)$ disjoint classes.

Procedure 1 NodePartitioning(T, l)

```
1: if T is a single node then
2:    mark T with label l
3: else
4:    find a 1/2-separator u in T
5:    mark u with label l
6:    remove u from T
7:    l ← l + 1
8:    for all subtrees T_s induced by the removal of u do
9:       NodePartitioning(T_s, l)
10:   end for
11: end if
```

Procedure 2 VERTEXPARTITIONING(G)

construct clique tree T from G
run NODEPARTITIONING($T, 1$)
for all nodes $u \in V(T)$ **do**
  for all unmarked vertices $v \in V(G) \cap u$ **do**
    mark v with the label of u
  end for
end for

Figure 1a) depicts the chordal graph G, Figure 1b) depicts the induced clique tree $T(G)$ after the NODEPARTITIONING procedure, while Figure 1c) depicts G after the VERTEXPARTITIONING procedure. The numbers denote the class of each node/vertex.

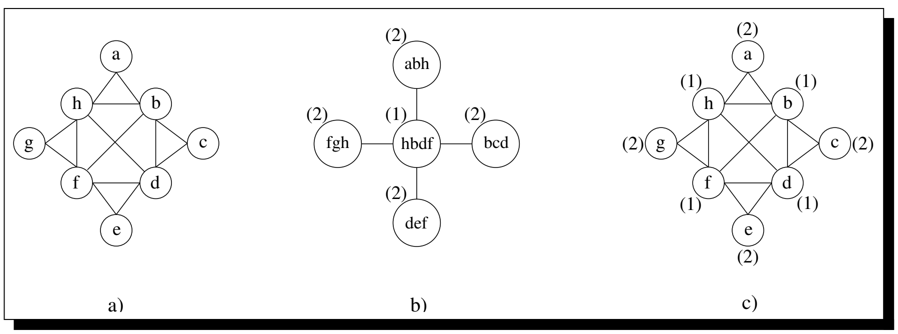

Figure 1: Separation of vertices into classes: a) a chordal graph G, b) the clique tree $T(G)$, c) partitioning of G in two classes

Selection Once we have classified the vertices of G, we need an on-line algorithm that chooses proper vertices to the Independent Set S. The on-line algorithm RANDMIS picks at random a label and accepts in a greedy way (using GREEDYSELECTION) only vertices with this label. The input σ is the sequence of the vertex arrivals, so σ_i is the i-th vertex of the sequence and its label is denoted by lab(σ_i).

Procedure 3 GREEDYSELECTION(v)

if v is not adjacent with any $u \in S$ **then**
  $S \leftarrow S \cup \{v\}$
end if

Algorithm 4 RandMIS(G, σ)

```
VertexPartitioning(G)
S ← ∅
pick a label l* uniformly at random from [1, ..., k]
for each arriving vertice σ_i of G do
    if lab(σ_i) = l* then
        GreedySelection(σ_i)
    end if
end for
```

Claim 1. *If the input sequence is restricted to vertices of a single class, then* GreedySelection *is 1-competitive.*

Proof. Let suppose that the selected class is class with label l. Let $u_1, \ldots, u_m$ be the nodes of $T(G)$ with label l. These nodes correspond to m cliques of G, after the VertexPartitioning procedure, that have no any vertex in common. The optimal offline algorithm can choose at most one vertex per clique u_i, because the maximum independent set of a clique is a singleton that contains any vertex of the clique. GreedySelection accepts exactly one vertex per clique. □

Lemma 1. *The* RandMIS *is $O(\log n)$-competitive, where n is the order of the chordal graph (for an oblivious adversary).*

Proof. For any input sequence σ of vertices of G, let RandMIS (σ) and ALG (σ) be the number of vertices accepted by the on-line and the offline algorithm respectively. Let c_l and o_l the number of vertices of class l accepted by the on-line and the offline algorithm respectively. By claim 1, $c_{l^*} = o_{l^*}$. Therefore

$$\begin{aligned}
\mathbf{E}[\textsc{RandMIS}(\sigma)] &= \sum_{l=1}^{\log n} \Pr[\text{chooses level } l] \cdot c_l \\
&\geq \sum_{l=1}^{\log n} \frac{1}{\log n} \cdot o_l \\
&= \frac{1}{\log n} \sum_{l=1}^{\log n} o_l \\
&= \frac{1}{\log n} \textsc{opt}(\sigma)
\end{aligned}$$

$\square$

The following lemma, suggests that there is no on-line algorithm that achieves better competitive ratio.

Lemma 2. $\lfloor \frac{\log(n+1)}{2} \rfloor$ *is a lower bound on the randomized (oblivious adversary) competitive ratio for the on-line maximum independent set in chordal graphs, where* n *is the order of the chordal graph.*

Proof. We will use the Yao's Principle in order to prove the lower bound. It is sufficient to produce a distribution on sequences of vertices σ, such that

$$\mathbf{E}[\textsc{opt}(\sigma)] > \frac{\log(n+1)}{2}$$

while for every on-line algorithm ALG, $\mathbf{E}[\textsc{alg}(\sigma)] \leq 1$.

For this purpose, we need firstly to construct a chordal graph G in the following way. We produce a complete binary tree B with n vertices. Then we recursively add edges in the following way: We add edges that join the root with any other vertex and we repeat this procedure for the two subtrees adjacent to the root and so on.

This chordal graph G, is in fact an interval graph, because its clique tree is a trail (theorem 4). We denote the vertex set of depth i with V_i and call them vertices of class i.

We consider G as the initial chordal graph of the problem. We also consider as sets of arriving vertices $V_i, 0 \leq i \leq \log(n+1) - 1$ where n is the order of G.

Note that every vertex in V_i is a vertex of class i and is adjacent to any leaf of the tree. Every vertex in V_i is adjacent with two vertices in V_{i+1}, with four vertices of V_{i+2} and so on.

Now, we produce the probability distribution as follows: Choose l in the set $\{1, 2, \ldots, \log n\}$ with probability $p_l = \frac{2^{-l}}{2-1/2^d}$, where $d = \log(n+1) - 1$ is the depth of the tree B.

Then, produce the arrivals of all the vertices in $V_1, V_2, \ldots, V_l$ and terminate the sequence.

Initially, we consider OPT in a random input σ. If this input terminates with vertices of class i, then OPT accepts exactly the vertices of class i, rejecting every vertex of class $j < i$.

Consequently,

$$
\begin{aligned}
\mathbf{E}[\textsc{opt}(\sigma)] &= \sum_{i=0}^{d} 2^i \cdot \frac{2^{-i}}{2-1/2^d} \\
&> \frac{\log(n+1)}{2}
\end{aligned}
$$

For every deterministic on-line algorithm ALG we claim the following:

Claim 2. *If* ALG *rejects a vertex v of V_i that is not adjacent to any already accepted vertex, then* $\mathbf{E}$[*the expected benefit of* ALG *by rejecting* v] $\leq 1 =$ *the benefit of* ALG *if it accepts* v.

Proof. Note that we have chosen a probability distribution such that Pr[ALG meets vertices of V_i| ALG has met vertices of V_{i-1}]$\leq \frac{1}{2}$. We will prove Claim 2 with induction on $i = d, d-1, \ldots, 0$. For the special case $i = d$, if the on-line algorithm rejects v, its expected benefit is 0, since there will not be other arrivals of vertices adjacent to v. Let's consider that the claim holds for the class j. We will prove that it also holds for the class $j-1$. If the algorithm rejects a vertex v of the class $j-1$, it hopes to the benefit of vertices of the class j, that are adjacent to v. Hence the expected benefit of ALG by rejecting v, is at most $\frac{1}{2}(\mathbf{E}[v_1] + \mathbf{E}[v_2])$, where v_1 and v_2 are the two adjacent vertices to v and $\mathbf{E}[v_i]$ is the maximum expected benefit of accepting or rejecting v_i. With probability $\frac{1}{2}$, ALG will meet both v_1 and v_2 and will have the choice to either accept them or to reject them. However, since v_1 and v_2 are class j vertices, we know by the induction hypothesis that $\mathbf{E}[v_1] = \mathbf{E}[v_2] \leq 1$, no matter if ALG accepts them or not.

□

Summarizing, ALG may accept or reject the first vertex. If it accepts it then ALG$(\sigma) = 1$. If it rejects it, then by Claim 2, $\mathbf{E}[\textsc{alg}(\sigma)] \leq 1$.

□

From the above lemmas the following main result for the on-line MIS in chordal graphs is induced.

Theorem 5. *The on-line MIS in chordal graphs has competitive ratio $O(\log n)$ (for oblivious adversary), where n is the order of the initial chordal graph.*

5 Applications

The specific on-line model for maximum independent set is particularly useful because it can be used to solve the on-line admission control. Instead of solving the on-line admission control or the on-line maximum edje-disjoint paths problem in a graph G, one may consider the intersection graph $I(G)$ which is induced by any possible request (path in G) and then solve the on-line maximum independent set in $I(G)$. In $I(G)$ two vertices are adjacent if the corresponding paths on G intersect.

We propose the following randomized algorithm for the on-line admission control problem in a tree T of order n.

Algorithm 5 RANDAC(T, σ)

1: $G \leftarrow I(T)$
2: RANDMIS(G, σ)

Corollary 1. *The* RANDAC *is $O(\log n)$-competitive, where n is the order of the tree T (for an oblivious adversary).*

Proof. The intersection graph $G = I(T)$ has $\binom{n}{2}$ vertices, as we consider any possible pair of vertices of T. Note that a pair of vertices of T denote a unique path in T. Consequently the competitive ratio c of RANDAC on T equals the competitive ratio of RANDMIS on G. Hence $c = O(\log \binom{n}{2}) = O(\log n)$ □

The algorithm RANDMIS partitions the node set of the clique tree in k classes. k becomes equal to $\log n$ in the case where the clique tree is a trail, otherwise $k < \log n$. As a matter of fact, we are not interested in 1/2-separators of the tree, but for the *minimum height elimination tree* of the clique tree. Hence the proposed here algorithm is $O(\log(\text{rank}(T)))$-competitive, where T is the clique tree and rank(T) the height of the minimum height elimination tree of T. According to [25] the following holds.

Proposition 1. *[25] If T is a tree on n vertices and of diameter D, then*

$$1 + \lfloor \log D \rfloor \leq \mathit{rank}(T) \leq 1 + \lfloor \log n \rfloor$$

and these bounds are tight.

The application of the algorithm in interval graphs is obvious. The interval graphs are intersection graphs in trails. Hence, we have the same result for the on-line admission control in trails.

As known by theorem 1, the family of chordal graphs is the family of the intersection graphs of subtrees in trees. This interesting property provides a straightforward algorithm for a generalization of on-line admission control in trees, where the calls are not just paths but subtrees. The problem is that the intersection graph may have superpolynomial number of vertices on $V(T)$, and in this case the competitive ratio is linear on $V(T)$.

In general the on-line MIS in chordal graphs is more general than the on-line admission control in trees. One can produce the former by the latter but not inversely. For example one can represent the tree as a chordal graph that considers every possible path. However, a chordal graph does not correspond to a network where every possible call is considered, but only a part of them.

6 Conclusion

In this paper we suggested an $O(\log n)$-competitive algorithm for the on- line maximum independent set problem in chordal and interval graphs. The specific on-line model can be applied to the on-line admission control in trees and lines and achieve competitive ratio $O(\log n)$. The same idea may also be applied in other intersection graph families (e.g. circular arc graphs) and produce results for the on-line admission control (e.g in rings).

References

[1] R. S. Anand and T. Erlebach. On-line algorithms for edge-disjoint paths in trees of rings. In *LATIN 2002: Theoretical Informatics, 5th Latin American Symposium, Cancun, Mexico, April 3-6, 2002, Proceedings*, volume 2286 of *Lecture Notes in Computer Science*, pages 584–597. Springer, 2002.

[2] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line load balancing with applications to machine scheduling and virtual circuit routing. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 623–631, 1993.

[3] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-competitive on-line routing. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science FOCS'93*, pages 32–40, 1993.

[4] B. Awerbuch, Y. Bartal, A. Fiat, and A. Rosen. Competitive non-preemptive call control. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1994.

[5] B. Awerbuch, R. Gawlick, T. Leighton, and Y. Rabani. On-line admission control and circuit routing for high performance computing and communication. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science FOCS'94*, pages 412–423, 1994.

[6] Y. Bartal, A. Fiat, and S. Leonardi. Lower bounds for on-line graph problems with application to on-line circuit and optical routing. In *STOC*, pages 531–540, 1996.

[7] P. A. Bernstein and N. Goodman. Power of natural semijoins. *SIAM J. Computing*, 10:751–771, 1981.

[8] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[9] M. Demange, X. Paradon, and V. T. Paschos. On-line maximum-order induces hereditary subgraph problems. In *SOFSEM 2000: Theory and Practice of Informatics, 27th Conference on Current Trends in Theory and Practice of Informatics, Milovy, Czech Republic, November 25 - December 2, 2000, Proceedings*, volume 1963 of *Lecture Notes in Computer Science*, pages 327–335. Springer, 2000.

[10] X. Deng, Y. Zhou, G. Li, and W. Zang. A 2-approximation algorithm for path coloring on trees of rings. In *Proceedings of the 11th Annual International Symposium on Algorithms and Computation ISAAC 2000*, pages 144–155, 2000.

[11] T. Erlebach. Approximation algorithms and complexity results for path problems in trees of rings. In Sgall et al. [29], pages 351–362.

[12] T. Erlebach and K. Jansen. Conversion of coloring algorithms into maximum weight independent set algorithms. In *ICALP Satellite Workshops*, pages 135–146, 2000.

[13] T. Erlebach and K. Jansen. The complexity of path coloring and call scheduling. *Theoretical Computer Science*, 255(1-2):33–50, 2001.

[14] T. Erlebach and K. Jansen. The maximum edge-disjoint paths problem in bidirected trees. *SIJDM: SIAM Journal on Discrete Mathematics*, 14, 2001.

[15] T. Erlebach, K. Jansen, C. Kaklamanis, M. Mihail, and P. Persiano. Optimal wavelength routing on directed fiber trees. *Theoretical Computer Science*, 221(1–2):119–137, 1999.

[16] J. A. Garay and I. S. Gopal. Call preemption in communication networks. In *IEEE INFOCOM '92, The Conference on Computer Communications, Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies, One World through Communications, May 4-8, 1992, Florence, Italy. IEEE*, volume 3 of *Lecture Notes in Computer Science*, pages 1043–1050, 1992.

[17] M. R. Garey, D.S. Johnson, G. L. Miller, and C. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM J. Algebraic Discrete Methods*, 1(2):216–227, 1980.

[18] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM J. Computing*, 1(2):180–187, 1972.

[19] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Combinatorial Theory*, 16:47–56, 1974.

[20] P. Gilmore and A.J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canadian Journal of Mathematics*, 15:539–548, 1964.

[21] M. Halldorsson. Online coloring known graphs. In *SODA: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 917–918, 1999.

[22] M. Halldorsson, K. Iwama, S. Miyazaki, and S. Taketomi. Online independent sets. In *Computing and Combinatorics, 6th Annual International Conference, COCOON 2000, Sydney, Australia, July 26-28, 2000, Proceedings*, volume 1858 of *Lecture Notes in Computer Science*, pages 202–209. Springer, 2000.

[23] M. Halldorsson and M. Szegedy. Lower bounds for on-line graph coloring. In *SODA: Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms*, pages 211–216, 1992.

[24] R. Kumar, R. Panigrahy, A. Russel, and R. Sundaram. A note on optical routing on trees. *Information Processing Letters*, 62:295–300, 1997.

[25] Y. Liang, S. K. Dhall, and S. Lakshmivarahan. Parallel approximate algorithms for node ranking of trees. In *Proceedings of the 2nd IEEE Symposium on Parallel and Distributed Processing*, pages 26–31, 1990.

[26] M. Mihail, C. Kaklamanis, and S. Rao. Efficient access to optical bandwidth - wavelength routing on directed fiber trees, rings, and trees of rings. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science FOCS'95*, pages 548–557, 1995.

[27] P. Raghavan and E. Upfal. Efficient routing in all-optical networks. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing STOC'94*, pages 134–143, 1994.

[28] D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976.

[29] J. Sgall, A. Pultr, and P. Kolman, editors. *Mathematical Foundations of Computer Science 2001, 26th International Symposium, MFCS 2001 Marianske Lazne, Czech Republic, August 27-31, 2001, Proceedings*, volume 2136 of *Lecture Notes in Computer Science*. Springer, 2001.

[30] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of ACM*, 28(2):202–208, 1985.

[31] J. Spinrad. *Efficient Graph Representations*. American Mathematical Society, 2003.

[32] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984.

[33] J. v. Leeuwen. *Graph Algorithms*, volume A of *Handbook of Theoretical Computer Science*. The MIT Press, 1990.

[34] G. Wilfong and P. Winkler. Ring routing and wavelength translation. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms SODA'98*, pages 333–341, 1998.