# Coloring Inductive Graphs On-line

Sandy Irani*
Computer Science Division
University of California
Berkeley, CA 94720.

July 24, 1996

**Abstract**

In this paper we consider the problem of on-line graph coloring. In an instance of on-line graph coloring, the nodes are presented one at a time. As each node is presented, its edges to previously presented nodes are also given. Each node must be assigned a color, different from the colors of it neighbors, before the next node is given. Let $A(G)$ be the number of colors used by algorithm $A$ on a graph $G$ and $\chi(G)$ the chromatic number of $G$. The performance ratio of an on-line graph coloring algorithm for a class of graphs $\mathcal{C}$ is $max_{G \in \mathcal{C}}(A(G)/\chi(G))$. We consider the class of $d$-inductive graphs. A graph $G$ is $d$-inductive if the nodes of $G$ can be numbered so that each node has at most $d$

*This research was supported by an IBM Graduate Fellowship.

edges to higher-numbered nodes. In particular, planar graphs are 5-inductive, and chordal graphs are $\chi(G)$-inductive. First Fit ($FF$) is the algorithm that assigns each node the lowest numbered color possible. We show that if $G$ is $d$-inductive then $FF$ uses $O(d \log n)$ colors on $G$. This yields an upper bound of $O(\log n)$ on the performance ratio of $FF$ on chordal and planar graphs. $FF$ does as well as any on-line algorithm for $d$-inductive graphs: we show that for any $d$ and any on-line graph coloring algorithm $A$, there is a $d$-inductive graph that forces $A$ to use $\Omega(d \log n)$ colors to color $G$. We also examine on-line graph coloring with lookahead. An algorithm is on-line with lookahead $l$, if it must color node $i$ after examining only the first $l + i$ nodes. We show that for $l < \frac{n}{\log n}$ the lower bound of $d \log n$ colors still holds.

# 1 Introduction

In an *on-line* problem, the input is obtained incrementally and partial output must be produced before the entire problem instance is known. If the problem to be solved is an optimization problem, the question is what kind of solution quality can be obtained given the fact that each part of the solution is produced without knowledge of the entire input. We examine the problem of on-line graph coloring: the graph is presented one node at a time, and when a node is presented, the edges from that node to all previously presented nodes are also given. Each node must be assigned a color, different from the colors of its neighbors, before the next node is presented. The object is to minimize the number of colors used. On-line graph coloring can be applied to processor assignment and register allocation problems [1], [9]. We measure an on-line algorithm in comparison to the optimal off-line algorithm. This notion was first introduced by Sleator and Tarjan and presented a way to do a worst case analysis of on-line algorithms [10].

Let $A(G)$ be the number of colors used by algorithm $A$ on the graph $G$. Let $\chi(G)$ be the chromatic number of $G$, the minimum number of colors required to color $G$ off-line. We are interested in finding an on-line graph coloring algorithm $A$, that for a class of graphs $\mathcal{C}$, minimizes

$$max_{G \in \mathcal{C}} \frac{A(G)}{\chi(G)}.$$

This is the *performance ratio* of $A$ for the class $\mathcal{C}$.

A fundamental issue of great importance is lookahead: what is it worth to know the future? It may be possible to delay a decision at some extra cost. In the case of processor allocation the cost might be delaying the scheduling of a task. In the case of register allocation, the price might be keeping extra registers to store values temporarily before they are placed in their assigned register. In either case, it is useful to know, given lookahead $l$, how much better an on-line algorithm can do. The only previous work on on-line algorithms with lookahead was done by Graham, Chung and Saks in the area of dynamic location problems (1-server problems with excursions) [2]. They characterize all graphs on which a dynamic location problem can be solved optimally with a fixed lookahead.

We examine on-line graph coloring for the class of inductive graphs. A $d$-inductive graph has the property that the nodes can be assigned distinct numbers in such a way that each node is adjacent to at most $d$ higher-numbered nodes. First Fit ($FF$) is the most natural on-line algorithm a practitioner would think of: it assigns to each node the lowest-numbered color possible (i.e. the lowest numbered color such that the node is not already adjacent to any nodes of that color). We show that $FF$ will use $O(d \log n)$ colors on any $d$-inductive graph with $n$ nodes. Karloff has a proof of the upper bound for First Fit that was discovered independently [5]. We show that this bound is tight: for any on-line graph coloring algorithm $A$, there is a $d$-inductive graph on which $A$ uses $\Omega(d \log n)$ colors. Since $d + 1$ is an upper bound on the chromatic number of any $d$-inductive graph, this yields a bound for any $d$ of $\Omega(\log n)$ on the performance ratio for any on-line algorithm on $d$-inductive graphs. The upper bound on the number of colors used yields an upper bound on the performance ratio for graphs where $d$ and the chromatic number are closely related. For example, since planar graphs are 5-inductive and chordal graphs are $\chi(G)$-inductive, the number of colors used by $FF$ is $O(\log n)$ for planar graphs and $O(\chi(G) \log n)$ for chordal graphs, which yields a performance ratio of $O(\log n)$ for both classes. The bound for planar graphs is tight to within a constant factor because one can show that for any on-line algorithm $A$, there is a tree $T$ such that $A(T) = \Omega(\log n)$. Since trees are also chordal, the result also yields a tight lower bound on the performance ratio for any on-line algorithm on chordal graphs.

When lookahead is added to the model, our results indicate that a substantial amount of lookahead is required to give an on-line algorithm any advantage. We show that even with lookahead $\frac{n}{\log n}$, an on-line algorithm still requires $\Omega(d \log n)$ colors to color a $d$-inductive graph. For lookahead $l > \frac{n}{\log n}$ we can do better, because we can color a $d$-inductive graph on-line in $\Theta( \ min\{d \log n, \frac{dn}{l}\})$ colors. It is surprising that the advantage of lookahead was not simply a tradeoff in $l$ , but rather produced a threshold effect. In addition, the point at which lookahead becomes an advantage is quite high.

The following is an outline of the techniques used in the proofs. The upper bound is proved by attaching a price to each node. Nodes to which $FF$ assigns higher-numbered colors cost more than nodes that are assigned lower numbered colors. The price of a node is a lower bound on the number of nodes in the graph. We use the constraint that the graph must be $d$-inductive to lower bound the price of a node that gets assigned a high color. Therefore we show that if $FF$ uses many colors, the graph must be large.

For the lower bound, the adversary presents a graph such that each node does not have more than $d$ edges to higher-numbered nodes (thus ensuring that the graph is $d$-inductive). Conceptually, the adversary divides the graph into phases. We show that the adversary can construct the graph in such a way that before every phase, he can determine all of the edges for nodes presented in the coming phase. After the $p^{th}$ phase, any on-line coloring algorithm has used at least $p$ colors to color the graph. To get the bound with lookahead, the adversary inserts $l$ independent nodes in between the nodes of each phase. Thus when the algorithm is coloring nodes in phase $p$ it cannot see any of the nodes in phase $p+1$. As long as the number of "independent" nodes does not dominate the total number of nodes, the same lower bound holds as in the case without lookahead.

## 2 Related Work

The on-line model is a natural way to view many problems that arise in practice. Such problems that have been studied in the past include on-line bin packing, operations on data structures and embedding process graphs on networks. Faigle, Kern and Turan discuss several on-line problems, rephrased as partitioning an independence system [3].

There has also been considerable work on on-line graph coloring. In evaluating an on-line graph coloring algorithm, keep in mind that the worst possible performance ratio is $n$. Lovasz, Saks and Trotter have shown an algorithm that has a performance ratio of $o(n)$ [8]; their algorithm achieves a performance ratio of $O(\frac{n}{log^* n})$ on all graphs. This is close to the best any on-line algorithm can do for general graphs. It has been shown that for any on-line algorithm $A$, and integer any $k$, there is a graph on at most $k(2^k - 1)$ nodes with chromatic number $k$ on which $A$ will use $2^k - 1$ colors [11]. This yields a lower bound of $\Omega(\frac{n}{(\log n)^2})$ for the performance ratio of any on-line algorithm on general graphs. Note that this lower bound does not say anything if we restrict our attention to the class of graphs where $k$ is fixed and the number of nodes in the graph can be arbitrarily large. Vishwanathan has a randomized on-line algorithm for coloring graphs where the chromatic number is a constant but the size of the graph can grow [12]. His algorithm achieves a competitive ratio of $O(n/\sqrt{\log n})$ colors.

Researchers have considered on-line coloring of more restricted classes of graphs. One can show that bipartite graphs can be colored on-line using $O(\log n)$ colors. This bound is matched by a lower bound of $\Omega(\log n)$ for any on-line algorithm on trees. Kierstead and Trotter have shown an on-line algorithm that achieves an optimal performance ratio of 3 on interval graphs [7]. Kierstead has shown that $FF$ has a constant performance ratio on the class of interval graphs [6]. Gyarfas and Lehel have shown that $FF$ achieves a constant performance ratio on split graphs, complements of bipartite graphs, and complements of chordal graphs [4]. Although $FF$ does well on some restricted classes of graphs, one can show that it does quite poorly in general. In fact, there is a bipartite graph on $2k$ nodes on which $FF$ will use $k$ colors. The graph is simply the complement of a perfect matching on $2k$ nodes.

# 3 Upper Bound

A graph $G$ is $d$-inductive if the nodes of $G$ can be ordered in such a way that each node has at most $d$ edges to higher-numbered nodes. Such an ordering on the nodes is called an *inductive order*. An inductive order is not necessarily unique for a graph. Note that if the nodes were presented in the reverse inductive order, then First Fit would use at most $d+1$ colors to color the graph. An inductive order of $G$ defines an orientation of the edges called an *inductive orientation* obtained by orienting the edges from the higher-numbered nodes to the lower numbered nodes. Notice that in an inductive orientation, the in-degree of each node is bounded by $d$. In fact the upper bound of $O(d \log n)$ for First Fit holds for any graph whose edges can be oriented so that the in-degree of every node is no more than $d$. However, any such graph is $2d$-inductive (the average degree is no more than $2d$), so using the weaker assumption can only yields an improvement of a constant factor.

**Theorem 1** *If $G$ is a $d$-inductive graph on $n$ nodes, then $FF$ uses $O(d \log n)$ colors to color $G$.*

The analysis will make use of a fixed inductive orientation of the graph. Note that the order in which the adversary presents the nodes need not be an inductive order. The adversary would like to present a graph on which $FF$ uses as many colors as possible and has as few nodes as possible. The nodes are numbered in the order they are presented. Examine the graph just after the adversary has presented node $j$. Let $N(j)$ denote the neighborhood of node $j$ when it is first presented. Suppose node $j$ gets color $c$. Then for every color in $\{1, \ldots, c-1\}$, there is a node in $N(j)$ that has been assigned that color. Fix a set of nodes $S_j$ such that :

- For every $i \in S_j$, $i < j$.
- $S_j \subset N(j)$
- For every $k \in \{1, \ldots, c-1\}$ there is exactly one node in $S_j$ that has been assigned the color $k$.

- $|S_j| = c - 1$

Notice that if we remove the edges between $j$ and lower numbered nodes not in $S_j$, then $FF$ will produce the same coloring. Let $S_j^{in}$ be those nodes in $S_j$ that are incident to edges directed into $j$, and $S_j^{out}$ those nodes on $S_j$ that are incident to edges directed out of $j$ in the inductive orientation. We know that $|S_j^{in}| \leq d$. Let $d_j = d - |S_j^{in}|$. Then $|S_j^{out}| = c - 1 - (d - d_j)$. Note that there can be at most $d_j$ edges from higher-numbered nodes directed into $j$. Associate with each node $j$ a value $P_j$ given by

$$P_j = 1 + \sum_{k \in S_j^{out}} \frac{P_k}{d_k}$$

Note that if $d_k = 0$, then there can not be any edges from higher-numbered nodes into node $k$. This implies that $k \notin S_j^{out}$ for every node $j$ where $j > k$. Thus the above sum is well defined.

Intuitively, if node $j$ gets color $c$, then $P_j$ is the number of nodes used in the graph to force node $j$ to get color $c$, or the "price" of node $j$. The constraint is that no node can have in-degree greater than $d$. $d_j$ denotes the number of additional edges that can be directed into node $j$ after it is first presented. We can think of these as "slots" for edges directed into $j$. So how many nodes were used to force $j$ to get the color $c$? One for the node $j$ itself. In addition, $j$ must be adjacent to nodes assigned colors $\{1, \ldots, c-1\}$. This is the set $S_j$. A slot is used up from every node $i$ in $S_j^{out}$ (i.e. if there is an edge from $j$ into $i$). Since $P_i$ is the price of node $i$ , and node $j$ uses up one of its $d_i$ slots, we add $P_i/d_i$ to the price of node $j$. To formalize this, we need to prove that if the price of a node is $P_i$ , then there are in fact at least $P_i$ nodes in the graph. Also we need to prove that nodes that are assigned higher-numbered colors, have a higher price.

**Lemma 2** *For any node $j$, $P_j \leq n$.*

**Proof:** We prove the claim by bounding partial sums defined as follows:

$$P_j^{(i)} = 1 + \sum_{k \in S_j^{out} \cap \{1, \ldots i\}} \frac{P_k}{d_k}$$

We show that for all $1 \leq i \leq n$:

$$\sum_{k=i}^{n} P_k^{(i-1)} \leq n$$

This implies the claim because $P_i^{(i-1)} = P_i$ and

$$P_i \leq P_i + \sum_{k=i+1}^{n} P_k^{(i-1)} = \sum_{k=i}^{n} P_k^{(i-1)} \leq n$$

By induction on $i$:

For $i = 1$: $P_1^{(0)} = \ldots = P_n^{(0)} = 1$. So

$$\sum_{k=1}^{n} P_k^{(0)} = n$$

By the induction hypothesis, we know that

$$\sum_{k=i}^{n} P_k^{(i-1)} \leq n$$

Suppose $i \in S_k^{out}$, then

$$P_k^{(i)} = 1 + \sum_{l \in S_k^{out} \cap \{1,\ldots,i\}} \frac{P_l}{d_l} = \frac{P_i}{d_i} + P_k^{(i-1)}$$

If $i \notin S_k^{out}$, then $P_k^{(i)} = P_k^{(i-1)}$

For every $k$ such that $i \in S_k^{out}$ there is an edge from $k$ into $i$. There are at most $d_i$ such edges, because there are at most $d_i$ edges from higher-numbered nodes into $i$. So we have

$$\sum_{k=i+1}^{n} P_k^{(i)} \quad \leq \quad d_i \left( \frac{P_i}{d_i} \right) + \sum_{k=i+1}^{n} P_k^{(i-1)}$$

$$= P_i^{(i-1)} + \sum_{k=i+1}^{n} P_k^{(i-1)}$$

$$= \sum_{k=i}^{n} P_k^{(i-1)} \leq n$$

$\square$

**Lemma 3** *If $i$ is colored with color $c$ then*

$$\textit{(1) if } d_i \neq 0, \textit{ then } \frac{P_i}{d_i} \geq \frac{1}{d}\left(\frac{d+1}{d}\right)^{c-1}$$

$$\textit{(2) if } d_i = 0, \textit{ then } P_i \geq \left(\frac{d+1}{d}\right)^{c-1-d}$$

**Proof:**

By induction on $c$.

- Case 1: $d_i \neq 0$. Each node in $S_i^{out}$ is colored with a different color in the range $1, \ldots c-1$. If a node $j$ is in $S_i^{out}$, then $d_j \neq 0$. Thus, by the induction hypothesis, we can lower bound $\frac{P_i}{d_i}$ as follows:

$$\frac{P_i}{d_i} \geq \frac{1}{d_i}\left[1 + \sum_{l \in S_i^{out}} \frac{1}{d}\left(\frac{d+1}{d}\right)^{l-1}\right]$$

$$\geq \frac{1}{d_i}\left[1 + \sum_{l=1}^{|S_i^{out}|} \frac{1}{d}\left(\frac{d+1}{d}\right)^{l-1}\right]$$

$$= \frac{1}{d_i}\left(\frac{d+1}{d}\right)^{|S_i^{out}|} = \frac{1}{d_i}\left(\frac{d+1}{d}\right)^{c-1-|S_i^{in}|} = \frac{1}{d_i}\left(\frac{d+1}{d}\right)^{c-1-d+d_i}$$

Since $d_i$ is an integer between 1 and $d$, this expression is minimized for $d_i = d$.

$$\frac{P_i}{d_i} \geq \frac{1}{d}\left(\frac{d+1}{d}\right)^{c-1}$$

- Case 2: $d_i = 0$. Similarly to above, we get

$$P_i \geq \left(\frac{d+1}{d}\right)^{c-1-d+d_i} = \left(\frac{d+1}{d}\right)^{c-1-d}$$

□

The two claims yield the theorem because if $FF$ colors a node with color $c$, then the size of the graph is at least $\frac{1}{d}\left(\frac{d+1}{d}\right)^{c-1-d}$.

$$c \leq log_{\frac{d+1}{d}} dn + d + 1 = O(d \log n)$$

# 4 Lower Bound

The following theorem shows that no on-line algorithm can perform asymptotically better than $FF$. Michael Saks has a simplified version of the proof of Theorem 4, but unfortunately, the simpler proof does not generalize for the case when lookahead is added to the model. We present a proof of the lower bound from which the lower bound with lookahead follows easily.

**Theorem 4** *For every on-line graph coloring algorithm $A$, and for every $d$, there is a family of $d$-inductive graphs $\mathcal{G}$ such that for every $n \geq d^{\,3}$, there is a $G \in \mathcal{G}$ where $G$ has $n$ nodes and $A(G) = \Omega(d \log n)$.*

The nodes are numbered from 1 to $n$ in the order they are presented. The adversary presents a graph such that the number of edges from any node to higher-numbered nodes is at most $d$. To do this, he must ensure that after a node is presented, at most $d$ more nodes will be adjacent to it. This ensures that $G$ is $d$-inductive; an inductive order of $G$ is the order in which the nodes are presented. We introduce the notion of *slots* which indicates for a particular node how many more edges can be attached to it without violating this condition. When a node is first presented, it has $d$ slots. As each additional node adjacent to it appears, the number of slots decreases by one. When a node has no slots left, the adversary cannot add any more edges incident to that node. If an on-line algorithm is coloring the nodes of a graph, then the number of slots of color $c$ is simply the total number of slots belonging to nodes that have been assigned color $c$.

The adversary works in phases starting with phase 1. He adds $X_p$ nodes in the $p^{th}$ phase and determines all the edges for these $X_p$ nodes at the beginning of a phase. The total number of edges added during the $p^{th}$ phase is $(p-1)X_p$. If the adversary plans to have $c$ phases, then he chooses the $X_p$'s as follows: $X_c = 2$.

$$X_{p-1} = \left\lceil \left(\frac{d+1}{d}\right) X_p + 2 \right\rceil$$

The adversary's goal is to force the algorithm to use $p$ different colors by the end of the $p^{th}$ phase. If he succeeds, then by the end of the last phase, the algorithm will have used at least $c$ colors to color the entire graph. To bound the total number of nodes in the graph, we first bound the number of nodes in each phase. Using the fact that $X_c \leq 3$ and $X_{p-1} \leq ((d+1)/d)X_p + 3$,

$$X_p \leq 3 \sum_{i=0}^{c-p} \left(\frac{d+1}{d}\right)^i \leq 3d \left(\frac{d+1}{d}\right)^{c-p+1}.$$

The total number of nodes in the graph is the sum over all phases of the number of nodes added in each phase:

$$\begin{aligned} n &\leq 3d \sum_{p=1}^{c} \left(\frac{d+1}{d}\right)^{c-p+1} \\ &= 3d \sum_{j=0}^{c-1} \left(\frac{d+1}{d}\right)^{j+1} \\ &\leq 3d^2 \left(\frac{d+1}{d}\right)^{c+1}. \end{aligned}$$

Since $n \geq d^{\,3}$, $c = \Omega(d \log n)$.

First we describe the adversary's strategy against First Fit and then we describe how the adversary alters the strategy when playing against an arbitrary on-line graph coloring algorithm $A$. We then prove that $A$ can do no better than First Fit.

If the adversary is playing against First Fit, then in the first phase, he presents $X_1$ independent nodes. First Fit assigns them all the color 1. In the second phase, the adversary presents $X_2$ nodes each of which is adjacent to a node that has been assigned color 1. First Fit will assign these nodes the color 2. In general, during phase $p$, every new node will be adjacent to $p-1$ nodes each of which is assigned a different color in $\{1, \ldots, p-1\}$. The adversary will only be able to do this if there are enough slots of colors $1, \ldots, p-1$. If he succeeds, then every node in phase $p$ will be assigned the color $p$. Let $FF_i(p)$ denote the number of slots of color $i$ remaining after phase $p$. We prove that the $X_j$'s are chosen in such a way that that following lemma holds:

**Lemma 5** *For any phase $p \leq c$, and for every $i \leq p$,*

($A$) $FF_{i-1}(p) \geq 2d + FF_i(p)$ *and* $FF_p(p) = dX_p$

($B$) *Every node in phase $p$ is assigned the color $p$.*

**Proof:** By induction on $p$. Assuming the claim is true for every phase through phase $p$, we prove the claim for phase $p+1$. Since for every $i \leq p$, color $i$ has at least $dX_p$ slots and $dX_p \geq X_{p+1}$, there are enough slots of colors $1, \ldots, p$ so that every node in phase $p+1$ can be adjacent to nodes that have been assigned colors $1, \ldots, p$. First Fit will assign each new node the color $p+1$, which proves Part B of the lemma. In addition, the same number of slots of each color is used up, so we have for $i \leq p$, $FF_{i-1}(p+1) \geq 2d + FF_i(p+1)$. Each of the new nodes is assigned the color $p+1$ and each new node has $d$ slots, so $FF_{p+1}(p+1) = dX_{p+1}$. The number of slots of color $p$ after phase $p+1$ is $dX_p - X_{p+1}$, because $X_{p+1}$ slots were used up in phase $p$. Since $dX_p \geq 2d + (d+1)X_{p+1}$, $FF_p(p+1) \geq 2d + FF_{p+1}(p+1)$. □

Now suppose the adversary plays against an arbitrary algorithm $A$. Let $A_i(p)$ denote the number of slots that algorithm $A$ has of color $i$ after phase $p$. We name each color according to the number of slots it has where color 1 has the most slots (i.e. $A_i(p) \geq A_{i+1}(p)$ ). If the relative number of slots of the colors changes, the colors are renamed accordingly. When the adversary plays against algorithm $A$, his goal is to keep $A$'s distribution of slots among colors as spread out as First Fit's distribution. Specifically, after each phase $p$, he wants to maintain the following two properties:

- **Property 1:** $\sum_{j=1}^{i} A_j(p) \leq d + \sum_{j=1}^{i} FF_j(p)$ for any $i \leq p$
- **Property 2:** $\sum_{j\geq 1} A_j(p) = \sum_{j\geq 1} FF_j(p)$

If both Properties hold after phase $p$, $A$ has used at least $p$ colors because the total number of slots $A$ has is more than the number of slots it has in the first $p-1$ colors after phase $p$. That is, since, $FF_p(p) \geq (d+1)$,

$$\begin{aligned}\sum_{i\geq 1} A_i(p) &= \sum_{i\geq 1} FF_i(p) \\ &> d + \sum_{i=1}^{p-1} FF_i(p) \geq \sum_{i=1}^{p-1} A_i(p)\end{aligned}$$

If the adversary has maintained both properties after phase $p-1$, he wants to choose the edges for the nodes in phase $p$ such that no matter what valid coloring $A$ uses on the nodes of phase $p$, both properties will be maintained. Notice that when the adversary plays against First Fit, he adds exactly $(p-1)X_p$ edges and $X_p$ nodes to the graph in phase $p$. Thus the total number of slots increases by $dX_p - (p-1)X_p$. The adversary adds the same number of edges and nodes per phase when playing against any algorithm. This ensures Property 2 because the change in the total number of slots is the same for both $A$ and First Fit. The adversary then must decide where to place these edges in order to maintain Property 1.

We define the following variables that determine the amount of "slack" in between $A$'s distribution and First Fit's. $s_i$ is the slack variable for color $i$. The slack variables are chosen in such a way that Algorithm $A$ could add $(d+1)s_i$ more slots to color $i$ for every $i \leq p-1$ and Property 1 would still hold after phase $p-1$.

For $i = 1$ to $p-1$,

$$s_i \leftarrow min_{p-1\geq k\geq i} \left\{ X_p, \left\lceil \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-1} s_j \right\}$$

If $s_{p-1} < X_p$ then for $i \geq p$,

$$s_i \leftarrow \min \left\{ X_p, \left\lfloor \frac{A_i(p-1)}{d+1} \right\rfloor \right\}$$

Otherwise, $s_i = 0$ for all $i \geq p$.

We prove the following lemma

**Lemma 6** *If Properties 1 and 2 hold after phase $p-1$ and the slack variables are chosen as indicated above, then for all $i$ where $1 \leq i \leq p-1$, $s_i \geq 0$ and*

$$d + \sum_{j=1}^{i} FF_j(p) \geq \sum_{j=1}^{i} [A_j(p-1) - X_p + (d+1)s_j]$$

**Proof:** Since $FF_j(p-1) - X_p = FF_j(p)$ for all $i \leq p-1$, we just have to prove $s_i \geq 0$ and

$$d + \sum_{j=1}^{i} FF_j(p-1) \geq \sum_{j=1}^{i} [A_j(p-1) + (d+1)s_j].$$

Since all variables are integers, this follows if $s_i \geq 0$ and

$$\left\lfloor \sum_{j=1}^{i} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rfloor - \sum_{j=1}^{i} s_j \geq 0.$$

The proof is by induction on $i$. We prove that for any $k$ where $p-1 \geq k \geq i$,

$$\left\lfloor \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rfloor - \sum_{j=1}^{i} s_j \geq 0 \tag{1}$$

Property 1 ensures that the claim is true for $i = 0$. Now assume the claim is true for $i$. Since $s_{i+1}$ is chosen to be the minimum over $k$ of the value in Line 1, then $s_{i+1}$ is just the minimum of a set of non-negative numbers and hence, $s_{i+1} \geq 0$. Furthermore, since for any $k$ where $p-1 \geq k \geq i+1$,

$$\left\lfloor \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rfloor - \sum_{j=1}^{i} s_j \geq s_{i+1}$$

we have that

$$\left\lfloor \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rfloor - \sum_{j=1}^{i+1} s_j \geq 0$$

□

The adversary chooses the edges for the next phase as follows: For $1 \leq i \leq p-1$, $X_p - s_i$ of the nodes in phase $p$ are adjacent to nodes of color $i$. For $i \geq p$, $s_i$ of the nodes in phase $p$ are adjacent to nodes of color $i$. If $\sum_{i=1}^{p-1} s_i > \sum_{i \geq p} s_i$, the adversary then adds $\sum_{i=1}^{p-1} s_i - \sum_{i \geq p} s_i$ extra edges between any node in phase $p$ and any node from phase $1, \ldots, p-1$.

When the adversary plays against First Fit, $(p-1)X_p$ edges are used in phase $p$, thus Property 2 holds if the adversary uses the same number of edges against algorithm $A$. Exactly $(p-1)X_p$ edges have been added in phase $p$ against $A$, if $\sum_{i=1}^{p-1} s_i - \sum_{i \geq p} s_i \geq 0$.

If $s_{p-1} = X_p$, the claim clearly follows because $\sum_{i \geq p} s_i = 0$. Otherwise, if $s_i < X_p$, then from from the choice of the $s_i$'s,

$$
\begin{aligned}
\sum_{i \geq p} s_i &\leq \sum_{i \geq p} \frac{A_i(p-1)}{(d+1)} \\
\sum_{i=1}^{p-1} s_i &\geq \sum_{i=1}^{p-1} \frac{FF_i(p-1) - A_i(p-1)}{(d+1)} \\
\sum_{i=1}^{p-1} s_i - \sum_{i \geq p} s_i &\geq \sum_{i \geq 1} \frac{FF_i(p-1) - A_i(p-1)}{(d+1)} = 0
\end{aligned}
$$

To prove that Property 1 holds after phase $p$, we first evaluate how the number of slots of each color has changed after algorithm $A$ has colored all the nodes added in phase $p$. We verify that the claim holds before reordering the colors according to their new number of slots. Then we verify that the claim holds after the colors are renumbered. For each color $i \leq p-1$, $X_p - s_i$ slots are used up and at most $ds_i$ new slots are added.

$$
\begin{aligned}
\sum_{j=1}^{i} A_j(p) &\leq \sum_{j=1}^{i} [A_j(p-1) - (X_p - s_j) + ds_j] &(2) \\
&= \sum_{j=1}^{i} [A_j(p-1) - X_p + (d+1)s_j] &(3)
\end{aligned}
$$

Line 3 and Lemma 6 imply that Property 1 holds before the colors have been renumbered according to their new number of slots. We use the following lemma to prove that Property 1 holds after the colors have been renumbered.

**Lemma 7** *Suppose $j < i$ and $A_j(p) < A_i(p)$ before the colors are reordered, then*

*(1) if $i \leq p-1$, then for every $l \leq i-1, A_i(p) \leq A_l(p-1) - X_p + (d+1)s_l$*

*(2) if $i > p-1$, then for every $l \leq p-1, A_i(p) \leq A_l(p-1) - X_p + (d+1)s_l$*

Lemma 7 together with Line 3 implies that for $i \leq (p-1)$, $\sum_{j=1}^{i} A_j(p) \leq \sum_{j=1}^{i} A_j(p-1) - X_p + (d+1)s_j$ even after the colors are reordered according to their new number of slots. Thus using Lemma 6, Property 1 for $i \leq p-1$ holds after renumbering. Property 1 holds for $i = p$ because First Fit and $A$ have the same total number of slots, and First Fit has only used colors $\{1, \ldots, p\}$,

$$\sum_{i=1}^{p} A_i(p) \leq \sum_{i \geq 1} A_i(p) = \sum_{i \geq 1} FF_i(p) = \sum_{i=1}^{p} FF_i(p)$$

Thus Property 1 still holds after the colors are reordered.

**Proof** of Lemma 7: We prove that if $j < i$ and if $A_j(p) < A_i(p)$ before the colors are reordered, then

$$(1) \text{ for } i \leq p-1,\ A_i(p) \leq A_{i-1}(p-1) - X_p + (d+1)s_{i-1}$$

$$(2) \text{ for } i > p-1,\ A_i(p) \leq A_{p-1}(p-1) - X_p + (d+1)s_{p-1}$$

In addition we prove that for every $i \leq p-1$,

$$A_i(p-1) - X_p + (d+1)s_i \leq A_{i-1}(p-1) - X_p + (d+1)s_{i-1}.$$

There are five cases.

- Case 1: $s_i = 0$. Since $A_i(p-1) \leq A_{i-1}(p-1)$, then it is clear that

  $$A_i(p-1) - X_p + (d+1)s_i \leq A_{i-1}(p-1) - X_p + (d+1)s_{i-1}.$$

  If $s_i = 0$ then all $X_p$ nodes in phase $p$ are adjacent to a node of color $i$, and color $i$ can not gain more slots relative to color $j$. Thus, it can not be the case that for $j < i$, $A_j(p) < A_i(p)$.

- Case 2a: $i \leq p-1$ and $s_{i-1} = X_p$.

$$\begin{aligned} A_i(p) &\leq A_i(p-1) - (X_p - s_i) + ds_i \\ &\leq A_i(p-1) - X_p + (d+1)s_i \\ &\leq A_{i-1}(p-1) - X_p + (d+1)X_p \\ &= A_{i-1}(p-1) - X_p + (d+1)s_{i-1} \end{aligned}$$

- Case 2b: $i \leq p-1$, $s_{i-1} < X_p$ and $s_i > 0$. $s_{i-1}$ was chosen so that for some $k$ where $i-1 \leq k \leq p-1$,

$$0 = \left\lceil \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-1} s_j$$

If $k \geq i$, then $s_i = 0$ because $s_i$ is chosen so that

$$s_i \leq \left\lceil \sum_{j=1}^{k} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-1} s_j.$$

So we have that

$$0 = \left\lceil \sum_{j=1}^{i-1} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-1} s_j \tag{4}$$

From the choice of the $s_i$'s,

$$s_i \leq \left\lceil \sum_{j=1}^{i} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-1} s_j \tag{5}$$

Subtracting Line 4 from Inequality 5,

$$s_i \leq \left\lceil \frac{FF_i(p-1) - A_i(p-1)}{d+1} \right\rceil.$$

Rearranging,

$$A_i(p-1) + (d+1)s_i \leq FF_i(p-1) + d \tag{6}$$

Using Lemma 5, we can upper bound $FF_i(p-1)+d$ by $FF_{i-1}(p-1)-d$. To upper bound $FF_{i-1}(p-1)$, we know from the choice of the $s_i$'s,

$$0 \leq \left\lceil \sum_{j=1}^{i-2} \frac{FF_j(p-1) - A_j(p-1)}{d+1} \right\rceil - \sum_{j=1}^{i-2} s_j \tag{7}$$

Subtracting Line 7 from Line 4 and rearranging,

$$FF_{i-1}(p-1) - d \leq A_{i-1}(p-1) + (d+1)s_{i-1} \tag{8}$$

Putting Inequalities 6 and 8 together,

$$A_i(p-1) + (d+1)s_i \leq A_{i-1}(p-1) + (d+1)s_{i-1}$$

We can now bound the number of slots of color $i$ after phase $p$.

$$\begin{aligned} A_i(p) &\leq A_i(p-1) - (X_p - s_i) + ds_i \\ &\leq A_{i-1}(p-1) - X_p + (d+1)s_{i-1} \end{aligned}$$

- Case 3a: $i \geq p$ and $s_{p-1} = X_p$. Since at most $dX_p$ slots can be added to any color,

$$\begin{aligned} A_i(p) &\leq A_i(p-1) + dX_p \\ &\leq A_{p-1}(p-1) + dX_p \\ &= A_{p-1}(p-1) - X_p + (d+1)s_{p-1} \end{aligned}$$

- Case 3b: $i \geq p$ and $s_{p-1} < X_p$.

$$s_i = \left\lfloor \frac{A_i(p-1)}{d+1} \right\rfloor$$

$$A_i(p-1) \leq (d+1)s_i + d$$

To bound the number of slots of color $i$ after phase $p$, $X_p - s_i$ nodes can be added and $s_i$ slots are used up. Thus,

$$\begin{aligned} A_i(p) &\leq A_i(p-1) + d(X_p - s_i) - s_i \\ &\leq dX_p + d \\ &\leq FF_p(p) + d \leq FF_{p-1}(p) - d \\ &= FF_{p-1}(p-1) - X_p - d \end{aligned}$$

Since $s_{p-1} < X_p$, from the argument in Case $2b$,

$$FF_{p-1}(p-1) - d \leq A_{p-1}(p-1) + (d+1)s_{p-1} \tag{9}$$

and

$$A_i(p) \leq A_{p-1}(p-1) - X_p + (d+1)s_{p-1}$$

□

# 5 On-line Coloring with Lookahead

Suppose the on-line model is slightly altered by allowing the algorithm to see the next $l$ nodes before assigning a color to the present node. The adversary presents the nodes one at a time. The nodes are numbered in the order in which the adversary presents them. When each node is presented, its edges to previously presented nodes are also given. Vertex $i$ must be assigned a color before node $i + l + 1$ is presented. In this case, we say that the graph is colored on-line with *lookahead $l$.*

**Theorem 8** *If $G$ is a $d$-inductive graph on $n$ nodes , then $G$ can be colored on-line with lookahead $l$ using $O(\ min\{d \log n, \frac{dn}{l}\})$ colors.*

**Proof:** If $d \log n < (d+1)\frac{n}{l}$, then ignore the lookahead and use $FF$ to color the graph. By Theorem 1, $FF$ will use $O(d \log n)$ colors.

If $d \log n \geq (d+1)\frac{n}{l}$, then divide the nodes into $\frac{n}{l}$ groups of $l$ consecutive nodes. (Consecutive in the order presented). The algorithm can see the subgraph induced by the nodes in an entire group before having to assign a color to the first node in the group. Since the subgraph induced by each group is $d$-inductive, it can be colored using $d+1$ colors. At most $d+1$ colors are used for every group, and at most $(d+1)\frac{n}{l}$ total colors are used. □

The following theorem shows that this is asymptotically the best possible.

**Theorem 9** *For every on-line graph coloring algorithm $A$ with lookahead $l$, and for every $d$, there is a family of $d$-inductive graphs $\mathcal{G}$ such that for every $n \geq d^3$, there is a $G \in \mathcal{G}$ where $G$ has $n$ nodes and $A(G) = \Omega(\ min\ \{d \log n, \frac{dn}{l}\})$*

We first prove the weaker bound of $\Omega(\ min\ \{d \log n, \frac{n}{l}\})$ . Let $c = min\{\frac{n}{l}, d \log n\}$. The graph is chosen exactly according to the strategy for Theorem 4 except that the adversary inserts $l$ independent nodes after each phase. This means that the algorithm with lookahead $l$ cannot see any of the nodes in a subsequent phase. The adversary does not count the slots from the $l$ independent nodes in determining where to add edges for the next phase. Thus, when phase $p-1$ is over, the adversary can already determine the edges for the phase $p$ nodes. This implies that the adversary can force an algorithm with lookahead $l$ to use $c$ colors. The number of nodes in the graph is bounded by $2n$.

To prove the stronger bound we alter the adversary strategy in Theorem 4 so that there are $\Theta(\log n)$ phases. After the $p^{th}$ phase, any on-line algorithm will have used $\Omega(pd)$ colors. Then when playing against an algorithm with lookahead, the adversary adds $l$ nodes after each phase, but there are now only $O(\log n)$ phases. Let $d' = \lfloor d/2 \rfloor$. In phase $p$, the adversary presents $X_p$ $d'$-cliques. Now each node has at least $d'$ slots. In the proof we attribute exactly $d'$ slots to each node when it first arrives. If the size of the graph is $n$, then the adversary chooses the number of phases, $c$, to be $min\{\frac{n}{l}, \log(n/d^2)\}$. He chooses the $X_i$'s as follows: $X_c = 4d'^2$ and $X_i = 2X_{i+1} + 4d'$. $O(n)$ nodes are used in the phases, and at most $n$ nodes are used between phases. Since $d'$ colors are used per phase, $A$ will use $\Omega(\ min\ \{d \log n, \frac{dn}{l}\})$ colors.

The nodes within a clique are all adjacent to the same set of nodes. When the adversary plays against First Fit, each node in each clique presented in phase $p$ is adjacent to nodes assigned colors $\{1, \ldots, (p-1)d'\}$. First Fit uses

$d'$ colors per phase. Let $\hat{FF}_i(p)$ be the number of slots of color $i$ First Fit has after phase $p$ under the new strategy.

**Lemma 10** *For any phase $p \leq c$, and for every $i \leq p-1$ and for any $1 \leq j, k \leq d'$,*

$(A)$ $\hat{FF}_{(i-1)d'+j}(p) \geq 4d'^2 + \hat{FF}_{id'+k}(p)$ *and* $\hat{FF}_{(p-1)d'+k}(p) = d'X_p$

$(B)$ $\hat{FF}_{id'+k}(p) = \hat{FF}_{id'+j}(p)$

$(C)$ *Each node in a clique presented in phase $p$ is assigned a color in $\{(p-1)d'+1, \ldots, pd'-1, pd'\}$.*

**Proof:** Similar to lemma 5. □

The adversary would like to use the same strategy as before in choosing the edges for the upcoming phase. However, the distribution $FF_i(p)$ had the property that for every $i \leq p$, $FF_{i-1}(p) \geq FF_i(p) + 2d$. Now $\hat{FF}_i(p)$ has the property that the colors can be grouped into blocks of $d'$ consecutive colors. Colors within a block have the same number of slots, and a color from a lower block has at least $4d'^2$ more slots than a color from a higher block. In order to be able to use the same method as before, we define a new distribution $\bar{FF}$. $\bar{FF}$ has the property that $\bar{FF}_i(p) \geq 4d' + \bar{FF}_{i+1}(p)$. It is defined as follows:

$$\bar{FF}_i(p) = \hat{FF}_i(p) + 4d'(d' - i \pmod{d'})$$

Furthermore we have the property that

$$\hat{FF}_j(p) \leq \bar{FF}_j(p) \leq [4d'^2 + \hat{FF}_j(p)] \tag{10}$$

Because $\bar{FF}_i(p) \geq 4d'^2$, for all $i \leq pd'$,

$$\begin{aligned}
\sum_{j=1}^{\lfloor pd'/2 \rfloor} \bar{FF}_j(p) &\leq \sum_{j=1}^{\lfloor pd'/2 \rfloor} [\hat{FF}_j(p) + 4d'^2] && (11)\\
&\leq \sum_{j=1}^{\lfloor pd'/2 \rfloor} \hat{FF}_j(p) + \sum_{j=\lfloor pd'/2 \rfloor+1}^{pd'} 4d'^2 && (12)\\
&\leq \sum_{j \geq 1} \hat{FF}_j(p) && (13)
\end{aligned}$$

Now when playing against an algorithm $A$, we would like to ensure the following two properties:

- **Property 1':** $\sum_{j=1}^{i} A_j(p) \leq (2d'-1) + \sum_{j=1}^{i} \bar{FF}_j(p)$ for any $i \leq pd'$
- **Property 2':** $\sum_{j\geq 1} A_j(p) = \sum_{j\geq 1} \hat{FF}_j(p)$

From the two properties and the bound from line 13, we know that $A$ has used at least at least $pd'/2$ colors after phase $p$. The slack variables for phase $p$ are chosen after phase $p-1$ as follows: For $i = 1$ to $pd'-1$,

$$s_i \leftarrow min_{pd'-1\geq k\geq i}\left\{X_p, \left\lceil \sum_{j=1}^{k} \frac{\bar{FF}_j(p) + d'X_p - A_j(p-1)}{2d'} \right\rceil - \sum_{j=1}^{i-1} s_j \right\}$$

If $s_{pd'-1} < X_p$ then for $i \geq pd'$,

$$s_i \leftarrow \min \left\{X_p, \left\lfloor \frac{A_i(p-1)}{2d'} \right\rfloor \right\}$$

Otherwise, $s_i = 0$ for all $i \geq pd'$.

We use the following lemma.

**Lemma 11** *If Property 1' and 2' hold after phase $p-1$ and the slack variables are chosen as indicated above, then for all $i$ where $1 \leq i \leq pd'-1$, $s_i \geq 0$ and*

$$(2d'-1) + \sum_{j=1}^{i} \bar{FF}_j(p) \geq \sum_{j=1}^{i} [A_j(p-1) - d'X_p + 2d's_j]$$

The proof for Lemma 11 is similar to the proof for Lemma 6. The adversary chooses the edges for phase $p$ so that if $i \leq pd'-1$, then $X_p - s_i$ cliques are adjacent to color $i$ nodes. (If a clique is adjacent to color-$i$ nodes, then all the nodes in the clique are adjacent to nodes that have been assigned the color $i$.) If $i > pd'$, then $s_i$ cliques in phase $p$ are adjacent to color-$i$ nodes. The adversary uses $d'^2(p-1)X_p$ slots in phase $p$ when playing against First Fit. He uses $d'(\sum_{j=1}^{pd'-1}(X_p - s_j) + \sum_{j\geq pd'} s_j)$ against algorithm $A$. The adversary then adds $d'(\sum_{j=1}^{pd'-1} s_j - \sum_{j\geq pd'} s_j - (d'-1)X_p)$ edges between any node in phase $p$ and any node from a previous phase. Property 2' follows from Lemma 12:

**Lemma 12** $\sum_{j=1}^{pd'-1} s_j - \sum_{j\geq pd'} s_j - (d'-1)X_p \geq 0$.

**Proof:** Let $i$ be the largest integer such that $i \leq pd'-1$ and $s_i < d'X_p$.

- Case 1: $i > d'(p-1)$. Let $i = d'(p-1)+l$.

$$\begin{aligned}
\sum_{j=1}^{i} s_j &\geq \sum_{j=1}^{i} \frac{\bar{F}F_j(p) + d'X_p - A_j(p-1)}{2d'} \\
\sum_{j\geq pd'} s_j &\leq \sum_{j\geq pd'} \frac{A_j(p-1)}{2d'}
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{j=1}^{i} 2d's_j - \sum_{j\geq pd'} 2d's_j \\
\geq\ &\sum_{j=1}^{i} [\bar{F}F_j(p) + d'X_p - A_j(p-1)] - \sum_{j\geq pd'} A_j(p-1) \\
\geq\ &\sum_{j=1}^{i} [\bar{F}F_j(p) + d'X_p] - \sum_{j\geq 1} A_j(p-1)
\end{aligned}$$

Since $\bar{F}F_j(p) + d'X_p = \bar{F}F_j(p-1)$ for $j \leq d'(p-1)$, and

$$\sum_{j\geq 1} A_j(p-1) = \sum_{j=1}^{d'(p-1)} \hat{F}F_j(p-1) \leq \sum_{j=1}^{d'(p-1)} \bar{F}F_j(p-1)$$

we have that

$$\begin{aligned}
&\sum_{j=1}^{i} 2d's_j - \sum_{j\geq pd'} 2d's_j \\
\geq\ &\sum_{j=d'(p-1)+1}^{d'(p-1)+l} [\bar{F}F_j(p) + d'X_p] \\
\geq\ &2d'lX_p
\end{aligned}$$

Since $s_j = X_p$ for $i < j < pd'$,

$$\sum_{j=1}^{pd'-1} s_j - \sum_{j \geq pd'} s_j \geq (d'-1)X_p$$

- Case 2: $i \leq d'(p-1)$ Since $s_j - X_p = 0$ for $d'(p-1) < j < pd'$, and $s_j = 0$ for $j \geq pd'$,

$$(\sum_{j=1}^{pd'-1} s_j - \sum_{j \geq pd'} s_j - (d'-1)X_p) = \sum_{j=1}^{d'(p-1)} s_j \geq 0.$$

□

To prove that Property 1' holds, for $i \leq pd'-1$, $d'(X_p - s_i)$ slots are used up and at most $s_i$ nodes are added to color $i$. Thus,

$$\begin{aligned} A_i(p) &\leq A_i(p-1) - d'(X_p - s_i) + d's_i \\ &= A_i(p-1) - d'X_p + 2d's_i \end{aligned}$$

Thus using Lemma 11, Property 1' holds before the colors are reordered.

We can prove the following lemma which is sufficient to prove that Property 1' holds after the colors have been renumbered.

**Lemma 13** *Suppose $j < i$ and $A_j(p) < A_i(p)$ before the colors are reordered, then*

*(1) if $i \leq pd'-1$, then for any $l \leq i-1, A_i(p) \leq A_l(p-1) - d'X_p + 2d's_l$*

*(2) if $i > pd'-1$, then for any $l \leq pd'-1, A_i(p) \leq A_l(p-1) - d'X_p + 2d's_l$*

The proof of Lemma 13 parallels the proof of Lemma 7.

# 6 Open Questions

The graph constructed by the adversary in the proof that $\Omega(d \log n)$ colors are needed to color $d$-inductive graphs is not necessarily chordal. Since trees are chordal, there is a lower bound of $\Omega(\log n)$ for the performance ratio on chordal graphs, but it remains open if for any $d$ and any on-line coloring algorithm $A$, there is a chordal graph with chromatic number $d$ such that $A$ uses $\Omega(d \log n)$ colors to color the graph.

Finally for the question of lookahead, is there a class of graphs for which less lookahead is required before the on-line performance improves? Or is there a general technique to extend lower bounds on the number of colors required without lookahead to the number of colors required with lookahead? Can one prove for more classes of graphs that if the lower bound for the number of colors required by an on-line algorithm to color a graph, $G$ in the class is $c(n, \chi(G))$, then for any $l \leq \frac{n}{c(n,\chi(G))}$, the lower bound still holds to within a constant factor?


# Acknowledgements

I would like to thank Dick Karp for suggesting the original problem, and Prabhakar Raghavan for the suggestion to extend it to on-line coloring with lookahead. Both of them also provided many helpful comments in preparing this manuscript. Special thanks also to Bill Burley for his careful reading of numerous versions of this manuscript and for his many insightful comments.


# References


[1] C.G. Chaitin. Register allocation and spilling via graph coloring. *Proceedings of Sigplan Symposium on Computer Construction*, Sigplan, Note 17, 6, pages 98–105, June 1982.

[2] F.K. Chung, R. Graham, M.E. Saks. A dynamic location problem for graphs. *Combinatorica*, Vol. 9, No. 2, pages 111-131, 1989.

[3] U. Faigle, W. Kern, G. Turan. On the performance of on-line algorithms for partitioning problems. *Acta Cybernetica* Vol. 9, pages 107–119, 1989.

[4] A. Gyarfas, J. Lehel. On-line and first fit colorings of graphs. *Journal of Graph Theory*, Vol. 12, No. 2, pages 217-227, 1988.

[5] H. Karloff. Personal communication.

[6] H.A. Kierstead. The linearity if first-fit coloring of interval graphs. *SIAM Journal of Discrete Mathematics*, Vol. 1, No. 4, November 1988, pages 526-530.

[7] H.A. Kierstead, W.A. Trotter. An extremal problem in recursive combinatorics *Congressus Numerantium 33*, pages 143–153, 1981.

[8] L. Lovasz, M.E. Saks, W.A. Trotter. An on-Line graph coloring algorithm with sublinear performance ratio. *Discrete Mathematics*, Special Volume on Graph Theory and Combinatorics , pages 319-326, 1988.

[9] D.A. Patterson. Reduced instruction set computers. *Communications of the ACM*, Vol. 28, No. 1, January 1985.

[10] D.D. Sleator, R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, Vol. 28, pages 202–208, February 1985.

[11] M. Szegedy. Personal communication, transmitted through [8].

[12] S. Vishwanathan. Randomized online coloring of graphs. *31st Annual Symposium on the Foundations of Computer Science*, 1990.