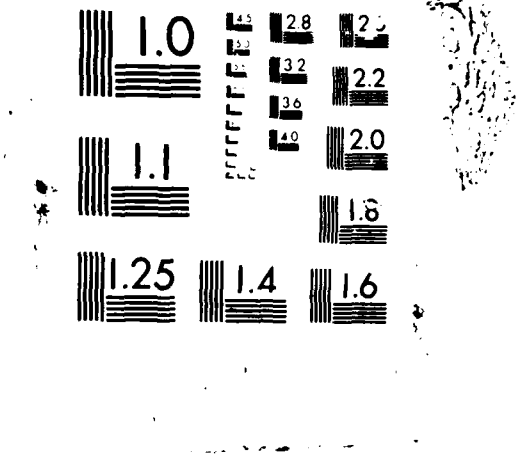
AD-A176 290 PARALLEL (DELTA + 1) COLORING OF CONSTANT-DEGREE GRAPHS (U) MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR COMPUTER SCIENCE A V GOLDBERG ET AL. DEC 86 1/1

UNCLASSIFIED VLSI-MEMO-86-355 N00014-80-C-0622 F/G 12/1 NL


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.8
3.2
3.6
4.0

VLSI Memo No. 86-355
December 1986

PARALLEL (Δ + 1) COLORING OF CONSTANT-DEGREE GRAPHS

Andrew V. Goldberg and Serge A. Plotkin

Abstract

This paper presents parallel algorithms for coloring a constant-degree graph with a maximum degree of Δ in (Δ + 1) colors and for finding a maximal independent set in a constant-degree graph. Given a graph with n vertices, the algorithms run in $O(\lg^* n)$ time on EREW PRAM with O(n) processors. The algorithms use only local communication and achieve the same complexity bounds when implemented in the distributed model of parallel computation.

Microsystems Research Center Room 39-321 | Massachusetts Institute of Technology | Cambridge Massachusetts 02139 | Telephone (617) 253-8138

Acknowledgements

This work was supported in part by a Fannie and John Hertz Foundation Fellowship and by the Defense Advanced Research Projects Agency under contracts numbered N00014-80-C-0622 and N00014-75-C-0661.

Author Information

Laboratory for Computer Science, MIT, Cambridge, MA 02139; Goldberg: Room NE43-342, (617) 253-5889; Plotkin: Room NE43-238, (617) 253-8861.

Parallel $(\Delta+1)$ Coloring of Constant-Degree Graphs

Andrew V. Goldberg*
Serge A. Plotkin†
Laboratory for Computer Science
MIT
Cambridge MA 02139

September 24, 1986

Abstract

This paper presents parallel algorithms for coloring a constant-degree graph with a maximum degree of Δ in $(\Delta+1)$ colors and for finding a maximal independent set in a constant-degree graph. Given a graph with n vertices, the algorithms run in $O(\lg^* n)$ time on EREW PRAM with $O(n)$ processors. The algorithms use only local communication and achieve the same complexity bounds when implemented in the distributed model of parallel computation.

Keywords: parallel algorithms, graph algorithms, graph coloring, maximal independent set.

1 Introduction

Both the maximal independent set (MIS) problem and the problem of vertex-coloring of a graph with a maximum degree of Δ in $(\Delta+1)$ colors (denoted by VC throughout this paper) are known to be in the class NC of problems that can be solved in polylogarithmic time using a polynomial number of processors. Polylogarithmic MIS algorithms are described in [7,8] and polylogarithmic graph coloring algorithms are described in [6]. The best previous parallel MIS algorithm is due to Luby [8]. The deterministic version of this algorithm runs in $O(\lg^2 n)$ EREW PRAM time using $O(n^2 m)$ processors. (Throughout this paper we use n to denote the number of vertices, and m to denote the number of edges in the input graph.) The total number of operations in this algorithm

*Supported by a Fannie and John Hertz Foundation Fellowship and by the Defense Advanced Research Projects Agency under the contract N00014-80-C-0622

†This work was supported in part by the Advanced Research Projects Agency under the Office of Naval Research contract N00014-75-C-0661

(given by the processor-time product) is $O(n^2 m \lg^2 n)$, which is a large number compared to $O(m)$ operations of the sequential algorithm. The graph coloring algorithms described by Karloff in [6] make use of Luby's MIS algorithm as a subroutine and therefore have a large processor-time product as well.

In this paper we present algorithms for the MIS and VC problems that work well for constant-degree graphs. For these graphs, the algorithms run in $O(\lg^* n)$ time using $O(n)$ processors, and the number of sequential operations of the algorithms is almost linear, namely $O(n \lg^* n)$. Our approach is a generalization of the deterministic coin-flipping technique of Cole and Vishkin [3]. Although their technique, like ours, can be viewed as an iterative improvement of coloring, it works only for directed chains whereas our method works for any constant-degree graph.

The algorithms presented in this paper can be implemented in the distributed model of computation [4,2], where processors have fixed connections determined by the input graph. The algorithms in the distributed model achieve the same $O(\lg^* n)$ bound as in the EREW PRAM model. Awerbuch has recently shown [1] using Ramsey theory that $\Omega(\lg^* n)$ time is required in the distributed model to find a maximal independent set on a chain. Our algorithms are therefore optimal (to within a constant factor) in the distributed model.

2 Definitions and Notation

This section defines the MIS and VC problems, describes the assumptions about the computational model, and introduces the notation used throughout the paper.

In this paper we use n to denote the number of vertices and m to denote the number of edges in the graph. We use Δ to denote the maximum degree of the graph.

The two problems discussed in this paper are the maximal independent set (MIS) and vertex-coloring (VC) problems. The MIS problem is, given an undirected graph $G = (V, E)$, to find a maximal set of vertices $I \subseteq V$ such that no two vertices in I are adjacent. A valid coloring of a graph is an assignment of a color to each vertex such that no two neighboring vertices have the same color. The vertex-coloring (VC) problem is to find a valid coloring that uses at most $\Delta + 1$ colors.

We use the following standard notation:

$$\begin{aligned} \lg x &= \log_2 x \\ \lg^{(1)} x &= \lg x \\ \lg^{(i)} x &= \lg \lg^{(i-1)} x \\ \lg^* x &= \min\{i \mid \lg^{(i)} x \leq 2\} \end{aligned}$$

We assume an exclusive-read, exclusive-write (EREW) PRAM model of computation where each processor is capable of executing simple word and bit operations. The word width is assumed to be $\lceil \lg n \rceil$. The word operations we use include bit-wise boolean operations, integer comparisons, and unary-to-binary conversion. In addition, each processor has a unique identification number (PE-ID).

3 The Algorithms

This section presents three algorithms. Given a graph with a constant degree Δ, the first algorithm colors it into a constant number of colors. The second algorithm uses this coloring and finds a maximal independent set in the graph. The third algorithm iteratively applies the second one to find an MIS in the graph and recolors the graph using only $\Delta+1$ colors. All these algorithms run in $O(\lg^* n)$ time on an EREW PRAM.

We first describe an $O(\lg^* n)$ time algorithm for coloring a constant-degree graph using a constant number of colors. The algorithm accepts as input a graph $G = (V, E)$ whose maximum degree is a constant Δ. Let C be an initial coloring of the graph. One possible way to obtain an initial coloring is to assign each vertex v a color C_v that is equal to the unique identification number of the processing element associated with v (denoted by PE-ID(v) in figure 1).

The main idea of the algorithm is to iteratively improve the coloring by constructing, for every vertex, a list of differences between the vertex and its neighbors and using this list as a new color for the vertex. Each element of the list is an ordered pair that consists of the index of a position in which the color of the vertex differs from the colors of one of its neighbors and the value of the bit at this position. The index can be represented by a string of bits of length $\lceil \lg L \rceil$ where L is the length of the representation of the current colors. Since the maximum degree is a constant, the length of the new representation is $O(\lg L)$.

The algorithm is shown in figure 1 and works as follows. Starting from a valid coloring C, we iteratively recolor the graph, each time reducing the number of colors. For each vertex v and each

```
PROCEDURE Color-Constant-Degree-Graph
L ← ⌈lg n⌉
for all v ∈ V in parallel set C_v ← PE-ID(v)   ;;; initial coloring
while L > Δ⌈lg L + 1⌉ for all v ∈ V in parallel do
      N_v ← {w | (v, w) ∈ E}
      for all w_k ← N_v(k), 1 ≤ k ≤ |N_v| do   ;;; find the index of diff. bit
          i_k ← min{i | C_v(i) ≠ C_{w_k}(i)}
          b_k ← C_v(i_k)
      end
      for all |N_v| < k ≤ Δ do
          i_k ← 0
          b_k ← C_v(0)
      end
      C_v ← i_1 b_1 i_2 b_2 ... i_Δ b_Δ   ;;; calculate new color
      L ← Δ⌈lg L + 1⌉
end
```

Figure 1: The Coloring Algorithm for Constant-degree Graphs

neighbor w of v we find an index i_w of the first bit in which C_v and C_w differ and construct a pair $\langle i_w, C_v(i_w)\rangle$. Such an index always exists because of the validity of the initial coloring. The new color of v is constructed by concatenating these pairs computed for all the neighbors of v.

Theorem 1 *The algorithm* Color-Constant-Degree-Graph *produces a valid coloring of a constant-degree graph in a constant number of colors in* $O(\lg^* n)$ *time on EREW PRAM.*

Proof: First we prove by induction that the coloring computed by the algorithm is valid, and afterwards we prove the upper bound on the execution time.

Each processor has a unique identification number and therefore the initial coloring is valid. We must show that if the coloring at the beginning of an iteration is valid, then the coloring calculated at the end of the iteration is also valid. Since the initial coloring is valid, for each pair of adjacent vertices v and w there exists at least one index i_w such that the colors C_v and C_w differ in bit position i_w. For each vertex v, the algorithm constructs a new color that can be viewed as a list of pairs of the form $\langle i_w, C_v(i_w)\rangle$, one pair per neighbor. In order for these lists to constitute a valid coloring, any two neighboring vertices v and w must have at least one different pair. Assume that the list constructed by the vertex v is

$$(\langle i_1, C_v(i_1)\rangle\langle i_2, C_v(i_2)\rangle \ldots \langle i_\Delta, C_v(i_\Delta)\rangle)$$

and the list constructed by the vertex w is

$$(\langle j_1, C_w(j_1)\rangle\langle j_2, C_w(j_2)\rangle \ldots \langle j_\Delta, C_w(j_\Delta)\rangle).$$

If for some k, $i_k \neq j_k$, we are done. On the other hand, if for all $1 \leq k \leq \Delta$, $i_k = j_k$, by the algorithm there exists a pair $\langle i_k, C_v(i_k)\rangle$ where i_k is an index of some bit position in which the colors C_v and C_w differ. Therefore, in this case, $C_v(i_k) \neq C_w(j_k)$. Hence, the constructed lists are always different for any two neighbors.

Each iteration of the algorithm is dominated by the computation of the index in which the colors of two neighboring vertices differ. Vishkin [3] shows how to find the minimum such index in constant time using word operations described in the previous section.

Now we show that the algorithm terminates in at most $O(\lg^* n)$ iterations. Let L_k denote the number of bits in the representation of colors after k iterations. For $k = 1$ we have

$$\begin{aligned} L_1 &= \Delta\lceil \lg L + 1\rceil \\ &\leq 2\Delta\lceil \lg L\rceil \end{aligned}$$

if $\lceil \lg L\rceil \geq \lceil \lg \Delta\rceil + 2$.
Assume that for some $k-1$ we have

$$\begin{aligned} L_{k-1} &\leq 2\Delta\lceil \lg^{(k-1)} L\rceil \\ \lceil \lg^{(k)} L\rceil &\geq \lceil \lg \Delta\rceil + 2. \end{aligned}$$

Then

$$\begin{aligned} L_k &= \Delta\lceil \lg L_{k-1} + 1\rceil \\ &\leq \Delta(\lceil \lg(2\Delta \lg^{(k-1)} L)\rceil + 1) \\ &\leq 2\Delta\lceil \lg^{(k)} L\rceil \end{aligned}$$

Therefore, as long as $\lceil \lg^{(k)} L\rceil \geq \lceil \lg \Delta\rceil + 2$,

$$L_k \leq 2\Delta\lceil \lg^{(k)} L\rceil.$$

Hence, the number of bits (L_k) in the representation of colors decreases until, after $O(\lg^* n)$ iterations, it reaches the value of $2\Delta\lceil \lg \Delta + 2\rceil$ or less, which is constant. The algorithm terminates at this point. ∎

Given a coloring of a graph in a constant number of colors, we can directly find an MIS in the graph.

Theorem 2 *In a constant-degree graph, MIS can be found in $O(\lg^* n)$ time on EREW PRAM using $O(n)$ processors.*

Proof: First, we use the algorithm *Color-Constant-Degree-Graph* to color the input graph in a constant number of colors. After the graph is colored, we iterate over all the colors, selecting the remaining vertices of the current color, adding them to the independent set and deleting all their neighbors from the graph. It can be shown by induction on the number of colors that this procedure produces an MIS.

The running time of this MIS algorithm is dominated by the *Color-Constant-Degree-Graph* subroutine: each iteration takes constant time, and the number of iterations is equal to the total number of colors and therefore is a constant. ∎

The above MIS algorithm can be used to produce a $(\Delta+1)$ vertex-coloring of the graph.

Theorem 3 *A graph whose maximum degree is a constant Δ can be colored in $\Delta+1$ colors in $O(\lg^* n)$ time on EREW PRAM using $O(n)$ processors.*

Proof: The $\Delta+1$ coloring algorithm works by iteratively finding an MIS in the input graph, coloring the MIS with a new color, and deleting it from the graph. If a vertex v is not removed at the end of some iteration, at least one of its neighbors is removed during this iteration. Therefore, after each iteration, the maximum degree of the graph induced by the remaining vertices decreases. Hence, after at most $\Delta+1$ iterations, the algorithm terminates producing a $\Delta+1$ coloring of the graph.

Each iteration of the algorithm takes $O(\lg^* n)$ time (by theorem 2), and the number of iterations is constant. Therefore, the overall running time of the algorithm is $O(\lg^* n)$. ∎

4 Discussion and Further Results

In this paper we have described a new technique for deterministic symmetry-breaking and have shown how to apply this technique to vertex-coloring and maximal independent set problems.

The algorithms presented in this paper have good asymptotic behavior, but the constant factors are large because the procedure *Color-Constant-Degree-Graph* uses a large number of colors compared to the graph degree. In [5] we describe a somewhat more complicated algorithm that uses significantly less colors.

The techniques used in this paper can be further extended. In combination with other methods, these techniques can be used to obtain polylogarithmic time linear processor algorithms for finding an MIS and a $(\Delta+1)$ vertex-coloring in a wide class of graphs that includes trees, planar graphs, graphs with polylogarithmic thickness (i.e. graphs that are unions of polylogarithmic number of planar graphs), and graphs with polylogarithmic maximum degree; see technical memorandum [5] for details. The memorandum also gives an $O(\lg^* n)$ algorithm for 3-coloring of a rooted tree, an $O(\lg n \lg^* n)$ algorithm for 7-coloring of a planar graph, and proves an $\Omega(\lg n / \lg\lg n)$ lower bound on any algorithm in CRCW PRAM model that 2-colors a rooted tree using a polynomial number of processors. The same lower bound is proven for the general MIS problem.

5 Acknowledgment

We would like to thank Charles Leiserson for fruitful and stimulating discussions, and for his valuable comments on a draft of this paper.

References

[1] B. Awerbuch. 1986. Personal Communication.

[2] B. Awerbuch. Complexity of network syncronization. *Journal of the Association for Computing Machinery*, 32(4):804–823, October 1985.

[3] R. Cole and U. Vishkin. Deterministic coin tossing and accelerating cascades: micro and macro techniques for designing parallel algorithms. In *Proc. 18th ACM Simp. on Theory of Computing*, pages 206–219, 1986.

[4] R. G. Gallager, P. A. Humblet, and P. M Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Transactions on Programming Languages and Systems*, 5(1):66–77, January 1983.

[5] A. Goldberg and S. Plotkin. *Efficient Parallel Coloring and Maximal Independent Set Algorithms.* Technical Report, MIT, 1986. (To appear).

[6] H. J. Karloff. *Fast Parallel Algorithms for Graph-Theoretic Problems: Matching, Coloring, Partitioning.* PhD thesis, University of California, Berkeley, 1985.

[7] R. M. Karp and A. Wigderson. A fast parallel algorithm for the maximal independent set problem. In *Proc. 16th ACM Simp. on Theory of Computing*, pages 266–272, 1984.

[8] M. Luby. A simple parallel algorithm for the maximal independent set problem. In *Proc. 17th ACM Simp. on Theory of Computing*, pages 1–10, 1985.

END

3-87

DTIC