Abstract. A fast parallel algorithm that computes a vertex colouring with a constant number of colours is presented. The algorithm works for a wide class of graphs, including graphs of fixed degree or of fixed genus.

It can be realized simultaneously within uniform Boolean depth $O(\log^2 n)$ and polynomial size.

An application of this colouring algorithm yields an $O(\log^2 n)$ depth computation of maximal independent sets, which considerably improves the known $O(\log^4 n)$ depth algorithm for a great class of graphs.

1. Introduction

An independent vertex set in an undirected graph $G = (V,E)$ is a subset of the vertex set $V$, which contains only pairwise nonadjacent vertices, and a vertex colouring is a partition of $V$ into independent sets. Since vertex colouring is a very central problem for the design of graph algorithms, it is of great interest to find efficient parallel algorithms constructing a vertex colouring with a small number of colours. While a great number of sequential algorithms attacking this problem are known, only very little is done in the field of parallel algorithms.

We present a method for finding a vertex colouring within $O(\log^2 n)$ Boolean depth. This algorithm colours graphs of a special $(p,q)$-type with a fixed number of colours. Degree bounded graphs as well as graphs of fixed genus are examples for special $(p,q)$-types. For instance, the planar graphs
are coloured with eight colours.

Related problems like finding a maximal independent set, minimal dominating set or minimal vertex cover are discussed in the last chapter. They are proved to be log-depth-reducible to the problem of finding a colouring with any fixed number of colours. Thus we obtain fast parallel algorithms solving these problems for a wide class of graphs. This includes an improvement of the \( O(\log^4 n) \) depth algorithm for the maximal independent set problem proposed by Karp/Wigderson [KaWi].

Furthermore, it should be noted that our algorithm generates a maximal independent set which is, up to a constant factor, maximum with respect to its cardinality.

The heart of the colouring algorithm is a graph partition procedure that simplifies the \((p,q)\)-type of a given graph until the \((1,1)\)-type is reached. But \((1,1)\)-type graphs are totally disconnected and therefore one-colourable.

2. A Graph Partition

Given a graph \( G = (V, E) \), the procedure presented below generates a partition \( P(G) = \{G(V_0), G(V_1)\} \) of \( G \) into two induced subgraphs \( G(V_0) \) and \( G(V_1) \). The partition bases upon a vertex partition \( V = V_0 \cup V_1 \) and will be used for the type reduction introduced in the next chapter.

Partition Procedure

Input: \( G = (V,E) \).

i) For each pair \( u,v \) of vertices compute \( \text{dist}(u,v) \), the minimum length of a path between \( u \) and \( v \) (or \( \infty \) if there is no path between \( u \) and \( v \)).

ii) Choose a root \( v_i \) for each connected component of \( G \).

iii) Define layers of \( G \):

\[
L_{ij} = \{ v : \text{dist}(v_i,v) = j \},
\]

\[
L_j = \bigcup_{i} L_{ij}
\]

iv) Compute the vertex partition by setting

\[
V_0 = \bigcup_{j \text{ even}} L_j, \quad V_1 = \bigcup_{j \text{ odd}} L_j.
\]

Output: \( P(G) = \{G(V_0), G(V_1)\} \).