Information Processing Letters 45 (1993) 285–290
Elsevier

16 April 1993

An efficiently solvable graph partition problem to which many problems are reducible

Fănică Gavril
CEMA(28), P.O. Box 2250, Haifa, Israel

Communicated by J. Håstad
Received 7 April 1992
Revised 25 June 1992 and 21 December 1992

Abstract

Gavril, F., An efficiently solvable graph partition problem to which many problems are reducible, Information Processing Letters 45 (1993) 285–290.

The 2-Colors Graph Partition problem (2-CGP) is the following: given a graph $G(V, E)$ with colors blue, white, or blue and white assigned to its edges, find a partition A, B of V, if one exists, such that $G(A)$ contains no white edges and $G(B)$ contains no blue edges. 2-CGP is a generalization of the recognition problem of bipartite graphs and many other problems are reducible to it. We prove that 2-CGP is log-space complete for NLOGSPACE by proving that 2-CGP and 2-CNFSAT are log-space reducible to each other. We describe a parallel algorithm for 2-CGP requiring $O(\log n)$ time and $O(n^3/(\log_4 n)^{1.5})$ processors on a CRCW PRAM which translates into parallel algorithms with the same performance for 2-CNFSAT and 2-QBF Truth.

Keywords: Parallel algorithms; 2-Colors Graph Partition; NLOGSPACE; 2-CNF Satisfiability; 2-QBF Truth

1. Introduction

We consider finite graphs $G(V, E)$ with no parallel edges and no self-loops, where V is the set of vertices and E the set of edges; $n = |V|$. G is a *2-colored graph* if each edge is colored blue, white, or blue and white; an edge can have both colors. We say that an *edge is white* (*blue*) if it is colored white (blue, respectively). Thus, edges colored white and blue are said to be white as well as blue. An *alternating path* is a path whose edges are alternately white and blue; its length is the number of its edges. An *alternating path* is called *white* (*blue*) if it starts with a white (blue, respectively) edge.

The 2-*Colors Graph Partition* (2-CGP) problem is the following: Given a 2-colored graph $G(V, E)$, is there a partition (called *legal*) A, B of V, such that $G(A)$ contains no white edges and $G(B)$ contains no blue edges? 2-CGP is a generalization of the recognition problem of bipartite graphs: if every edge of G is colored both blue and white, then G is bipartite iff it has a legal partition. In the present paper we show that 2-CGP has very efficient algorithms and is amenable to straightforward reductions from many other well-known problems, helping to solve them efficiently.

Let W be a finite set of Boolean variables. A *literal* is a variable x of W or its negation $\bar{x}$. A

Correspondence to: F. Gavril, CEMA(28), P.O. Box 2250, Haifa, Israel.

clause c over W is a Boolean expression of the form $x_1 + x_2 + \cdots + x_k$ where each x_i is a literal over W and "+" represents OR. A *satisfying truth assignment* for a conjunction C of clauses over W is a TRUE/FALSE assignment to the variables of W satisfying each clause of C. The *2-CNF Satisfiability* (2-CNFSAT) problem asks if a given conjunction C of clauses, with exactly two distinct literals per clause, has a satisfying truth assignment. The *2-QBF Truth* (2-QBFTR) problem asks if a quantified Boolean formula $F = Q_1 x_1 \cdots Q_k x_k C$, where each Q_i is $\forall$ or $\exists$ and C is a conjunction of clauses with exactly two distinct literals per clause, is true. NLOGSPACE is the family of problems having log-space non-deterministic algorithms. A problem $P \in$ NLOGSPACE is *log-space complete* for NLOGSPACE if every problem in NLOGSPACE is log-space reducible to P. We prove that 2-CGP and 2-CNFSAT are log-space reducible to each other in linear time obtaining that 2-CGP, like 2-CNFSAT [8,9] is log-space complete for NLOGSPACE.

Linear time sequential algorithms for 2-CNFSAT which translate into algorithms for 2-CGP appear in [1,4]. We present for 2-CGP a parallel algorithm requiring O(log n) time and $O(n^3/(\log_4 n)^{1.5})$ processors on a CRCW PRAM which translates into parallel algorithms with the same performance for 2-CNFSAT and 2-QBFTR, improving on the algorithms in [1,3] which require O(log n) time and $O(n^4)$ processors. The algorithms in [1,3,4] can also be implemented in parallel, using Boolean matrix multiplications, with the same time performance: the algorithm in [4] by using an appropriate translation of Lemma 2 below, and the algorithms in [1,3] by the method described in [10].

2. Algorithms for 2-CGP, 2-CNFSAT and 2-QBFTR

Theorem 1. 2-CGP *and* 2-CNFSAT *are log-space reducible to each other in linear sequential time.*

Proof. Given an input C, W for 2-CNFSAT, the corresponding input $G_C(V, E)$ for 2-CGP is: the vertices of G_C are the literals over W; two literals x, $\bar{x}$ of the same variable are connected by a white edge and two literals appearing in the same clause are connected by a blue edge. C is satisfiable iff there exists a legal partition A, B of V: the elements of A are those to receive the value TRUE.

Given an input $G(V, E)$ for 2-CGP, the corresponding input C, W for 2-CNFSAT is: to each vertex $v \in V$ corresponds a Boolean variable $v \in W$; to every edge connecting u and v corresponds a clause $\bar{u} + \bar{v}$ if the edge is white and a clause $u + v$ if it is blue. A partition A, B of G is legal iff no clause $\bar{u} + \bar{v}$ has $u,v \in A$ and no clause $u + v$ has $u,v \in B$, that is, iff C is satisfied by assigning TRUE exactly to the elements of A.

The above reductions can clearly be done in linear sequential time and in logarithmic (in fact, constant) working space. □

Theorem 1 together with the results in [8,9] imply the following:

Corollary. 2-CGP, *as* 2-CNFSAT, *is log-space complete for* NLOGSPACE.

The linear-time sequential algorithms for 2-CNFSAT given in [1,4] translate by Theorem 1 into linear-time algorithms for 2-CGP. We describe a parallel algorithm for 2-CGP using Boolean matrix multiplications of adjacency matrices. The i,j-entry of a matrix M is denoted $M(i, j)$. The identity matrix is denoted I: $I(i, j)$ is 1 if $i = j$ and 0 if $i \neq j$.

Consider a 2-colored graph $G(V, E)$, with adjacency matrix M, as an input for 2-CGP. We separate G into two edge subgraphs, one for the white edges, with adjacency matrix W, and one for the blue edges, with adjacency matrix B. As known, there exists a path of length k from v_i to v_j iff $M^k(i, j) = 1$. Thus, there exists a path from v_i to v_j iff $(M + I)^n(i, j) = 1$. Let $WB = W \times B + I$, $BW = B \times W + I$, $NW = WB^n \times W$ and $NB = BW^n \times B$. As above, there exists a white (blue) odd alternating path from v_i to v_j iff $NW(i, j) = 1$ ($NB(i, j) = 1$, respectively). Let

$$X_1 = \{v_i \mid NW(i, i) = 1\},\ Y_1 = \{v_i \mid NB(i, i) = 1\},$$
$$X_2 = \{v_j \mid NW(i, j) = 1 \text{ for some } v_i \in Y_1,\ i \neq j\},$$

$Y_2 = \{v_j \mid NB(i, j) = 1 \text{ for some } v_i \in X_1, i \neq j\}$,

$X_3 = \{v_j \mid NB \times W(i, j) = 1$

$\qquad \text{for some } v_i \in X_1, i \neq j\}$,

$Y_3 = \{v_j \mid NW \times B(i, j) = 1$

$\qquad \text{for some } v_i \in Y_1, i \neq j\}$,

$X = X_1 \cup X_2 \cup X_3, \quad Y = Y_1 \cup Y_2 \cup Y_3$.

Clearly, X_1 is the set of vertices having white odd alternating paths to themselves, Y_2 is the set of vertices having blue odd alternating paths from vertices of X_1 and X_3 is the set of vertices having blue even alternating paths from vertices of X_1. Similarly, Y_1 is the set of vertices having blue odd alternating paths to themselves, X_2 is the set of vertices having white odd alternating paths from vertices of Y_1 and Y_3 is the set of vertices having white even alternating paths from vertices of Y_1.

Lemma 2. *Let $G(V, E)$ be an input for 2-CGP and let X, Y be defined as above. A legal partition of G exists iff $X \cap Y = \emptyset$. Furthermore, in any legal partition A, B of G the vertices of X must be assigned to B and the vertices of Y must be assigned to A.*

Proof. By definition, in any legal partition A, B of G, if a vertex u is assigned to A, then any vertex v connected to u by a white edge must be assigned to B, any vertex w connected to v by a blue edge must be assigned to A, and so on. Thus, if a vertex u is assigned to A then any vertex v having a white odd alternating path from u must be assigned to B and any vertex w having a white even alternating path from u must be assigned to A. Similarly, if a vertex u is assigned to B then any vertex v having a blue odd alternating path from u must be assigned to A and any vertex w having a blue even alternating path from u must be assigned to B. Thus, in any legal partition, a vertex having a white odd alternating path to itself cannot be assigned to A. Hence, the vertices of X_1 must be assigned to B implying that the vertices of Y_2 must be assigned to A and the vertices of X_3 must be assigned to B. Similarly, the vertices of Y_1 must be assigned to A those of X_2 to B and those of Y_3 to A. Thus, the vertices of X must be assigned to B and those of Y to A. In conclusion, if a legal partition exists, then $X \cap Y = \emptyset$.

Conversely, assume that $X \cap Y = \emptyset$. Assign the vertices of X to B and those of Y to A. The unassigned vertices have no white edges to vertices of A and no blue edges to vertices of B. Consider any unassigned vertex v; let $Y_4 = \{v\}$ and $X_4 = \emptyset$. Assign to X_4 the unassigned vertices connected to vertices of Y_4 by a white edge and, after that, assign to Y_4 the unassigned vertices connected to vertices of X_4 by a blue edge; return on this step until no more unassigned vertices are assigned to either X_4 or Y_4. Since X_4, Y_4 are assigned only vertices unassigned to X, Y, it follows that no two vertices connected by a white edge are assigned to Y_4 and no two vertices connected by a blue edge are assigned to X_4; assign the vertices of Y_4 to A and those of X_4 to B. Continuing in the same way with the remaining unassigned vertices, a legal partition is obtained. □

The parallel algorithm for 2-CGP computes the matrices NW, NB, finds the sets X_1, Y_1, X_2, Y_2, X_3, Y_3, X, Y, and tests that $X \cap Y = \emptyset$. Assuming that a legal partition exists, it assigns X to B and Y to A. It assigns the unassigned vertices as follows: For an unassigned vertex v, both assignments $v \in A$ and $V \in B$ lead to legal partitions, but each will force different sets of vertices to A and B. Consider the corresponding submatrices NW', B' on the unassigned vertices. Consider the matrix $N' = NW' \times B' + NW'$: $N'(i, j) = 1$ iff there is a white (odd or even) alternating path from v_i to v_j. For every vertex v_j let i_j be the maximal row index such that $N'(i_j, j) = 1$. For every unassigned vertex v_j, the algorithms assigns v_j to B if $NW'(i_j, j) = 1$ and to A if $NW'(i_j, j) = 0$. Let us prove that this partition is legal.

Consider two vertices v_j, v_r assigned to A and assume that they are connected by a white edge. Since $N'(i_j, j) = 1$ and $NW'(i_j, j) = 0$, it follows that there exists a white even alternating path from v_{i_j} to v_j which together with the white edge between v_j and v_r gives a white odd alternating path from v_{i_j} to v_r. Hence $N'(i_j, r) = 1$ and $i_r \geqslant i_j$. By symmetry $i_j \geqslant i_r$, therefore $i_j = i_r$. This implies that the white even alternating paths from v_{i_j} to v_j and from $v_{i_r} = v_{i_j}$ to v_r together with the white edge between v_j and v_r form a white odd alternating path from v_{i_j} to itself, thus $v_{i_j} \in X_1$ and this is a contradiction. Similarly, no two vertices assigned to B are connected by a blue edge.

The algorithm is on a CRCW PRAM. A parallel algorithm for Boolean matrix multiplication needs constant time and $O(n^3/(\log_4 n)^{1.5})$ processors based on the algorithm in [2] or $O(\log n)$ time and $O(n^{2.376})$ processors based on the algorithm of Coppersmith and Winograd. The nth power of a matrix can be computed in $\lceil \log n \rceil$ multiplications. Thus, the algorithm can be implemented in time $O(\log n)$ using $O(n^3/(\log_4 n)^{1.5})$ processors or in time $O(\log^2 n)$ using $O(n^{2.376})$ processors. It translates into an algorithm for 2-CNFSAT with the same performance.

Consider a 2-QBF formula $F = Q_1 x_1 \cdots Q_k x_k C$ and construct for C the graph G_C defined in Theorem 1. A vertex of G_C appearing with a universal (existential) quantifier is called universal (existential, respectively); its index is the index of its quantifier. Let X_0 be the set of existential vertices u having a white alternating path to a universal vertex v^u of a higher index. Let $Y_0 = \{\bar{u} \mid u \in X_0\}$; if an element $u \in X_0$ is assigned TRUE, it will determine the value of the universal vertex v^u, contradicting its universality. Thus, the elements of X_0 must be assigned to B and those of Y_0 to A. Let X_1, Y_1 be the sets defined before Lemma 2 and add X_0 to X_1, Y_0 to Y_1; let X_2, Y_2, X_3, Y_3, X, Y be based on the new sets X_1, Y_1. By Theorem 1 and Lemma 2, the vertices of X must be assigned to B and those of Y to A.

Lemma 3. *Let $F = Q_1 x_1 \cdots Q_k x_k C$ be a 2-QBF formula and let G_C be the graph defined above. F is true iff in G_C there is no alternating path between two universal vertices, $X \cup Y$ contains no universal vertex and $X \cap Y = \emptyset$.*

Proof. If F is true the above conditions are fulfilled, otherwise the value of a universal variable is predetermined or C is unsatisfiable.

Conversely assume that the above conditions are fulfilled. All the pairs of universal vertices u, $\bar{u}$ can be freely assigned as $u \in A$, $\bar{u} \in B$ or $\bar{u} \in A$, $u \in B$. For any such assignment, alternately, until no more possible, assign to B (A) the unassigned existential vertices connected to vertices of A (B) by a white (blue, respectively) edge. Continue in the same way with the unassigned existential vertices. Each such assignment is a satisfying truth assignment for C, thus F is true. □

The parallel algorithm for 2-QBFTR tests the conditions in Lemma 3 using Boolean matrix multiplications. The algorithm computes the matrices WB^n, BW^n, NW, NB, $N_1 = NW + WB^n$ and $N_2 = NB + BW^n$; $N_1(i, j) = 1$ ($N_2(i, j) = 1$) iff there is a white (blue, respectively) alternating path from v_i to v_j. The algorithm finds X_0, Y_0 using N_1, finds X_1, Y_1 defined before Lemma 2 adds X_0 to X_1, Y_0 to Y_1 and finds X_2, Y_2, X_3, Y_3, X, Y, based on the new sets X_1, Y_1. The condition in Lemma 3 that there is no alternating path between two universal vertices is tested on N_1 and N_2. The other two conditions are tested on X, Y. The algorithm has the same performance as the one for 2-CGP.

3. Other problems reducible to 2-CGP

This section presents examples of well-known problems which are log-space reducible to 2-CGP in linear sequential time.

A graph $G(V, E)$ is a split graph if there is a partition A, B of V such that in $G(A)$ no two vertices are adjacent and in $G(B)$ every two vertices are adjacent. To find out if a given graph G is a split graph, transform G into a completely connected graph H by adding the missing edges. Color the edges of H appearing in G by blue and those not appearing in G by white. G is a split graph iff there exists a partition A, B of V such that $H(A)$ contains no blue edges and $H(B)$ contains no white edges.

A graph $G(V, E)$ is semipartite [6] if the maximum matching cardinality is equal to the minimum node covering cardinality. To find out if a given graph G with a maximum matching M is semipartite, assign both colors white and blue to the edges of M and assign the color blue to the edges of $E-M$. Let X be the set of vertices adjacent to no edges of M. G is semipartite iff there exists a partition C, $V-C$ of V, with the vertices of X preassigned to $V-C$ such that $G(V-C)$ contains no blue edges and $G(C)$ contains no white edges. If such a partition exists, then C is a node covering – $G(V-C)$ containing no edges – every edge of M is incident to exactly one vertex of C and every vertex of C is adjacent to exactly one edge of M. Conversely, if G is semipartite and C is a minimum node covering, then $G(V-C)$ contains no edges and every edge of M has exactly one endvertex in C since $|C| = |M|$. Thus, $G(C)$ contains no white edges and $G(V-C)$ contains no blue edges.

Consider an $n \times m$ matrix M and let $C \subseteq \{1, 2, \ldots, n\} \times \{1, 2, \ldots, m\}$ be the set of its non-zero entries. To find out if there is a partition A, B of C such that no two elements of A (B) appear in the same row (column), construct a graph $G(V, E)$ whose vertices correspond to the elements of C and connect by white (blue) edges vertices corresponding to non-zero entries appearing in the same row (column, respectively). A partition of C, as requested, exists iff there exists a partition A, B of V such that $G(A)$ contains no white edges and $G(B)$ contains no blue edges.

A k-coloring of the line graph of a graph $G(V, E)$ is a family $F=\{S_1, \ldots, S_k\}$ of k disjoint subsets of E, no S_i containing two incident edges. An h-covering of the line graph of G is a family $H=\{C_1, \ldots, C_h\}$ of h disjoint subsets of E such that every two edges in C_j are incident. Denote $M=\bigcup_{S \in F} S$, $N=\bigcup_{C \in H} C$. A k-coloring F and an h-covering H are orthogonal [5,7] if $E=M \cup N$ and if $|S \cap C| = 1$ for every $S \in F$, $C \in H$. Given a graph G and a k-coloring F with $k \geqslant 3$, of its line graph, is there an h-covering, for any h, orthogonal to the given k-coloring? For $D \subseteq V$ denote $E(D)=\{e \mid e \in E,\ e$ is incident to some $v \in D\}$; denote $E(v)=E(\{v\})$.

Lemma 4. *An h-covering orthogonal to F exists iff G has a set of vertices D fulfilling*:

(a) $|E(v) \cap M| = k$ *for every* $v \in D$,
(b) $G(V-D)$ *contains no edge of* $E-M$,
(c) $G(D)$ *contains no edges of* M.

Proof. Assume that an h-covering H orthogonal to F exists. If $C \in H$ has $|C| = k$, then $C-M=\emptyset$ and $H-\{C\}$ is also orthogonal to F. Thus, we can assume that every $C \in H$ has $|C| > k \geqslant 3$. Every $C \in H$ is a set of $|C| \geqslant 4$ mutually incident edges, thus there exists a unique vertex $v_C \in V$ having $C \subseteq E(v_C)$. Let $D=\{v_C \mid C \in H\}$. Clearly, for every $V_C \in D$, $|E(v_C) \cap M| = |C \cap M| = k$. In addition, no $e \in E-M=N-M \subseteq N \subseteq E(D)$ is contained in $G(V-D)$. Assume that two vertices $v_C, v_B \in D$ are connected by an edge $e \in S_p$. Since $C \cap B = \emptyset$, at least one of them, say C, does not contain e. Then, the edge $f \in C \cap S_p$ is incident to v_C and e, contradicting the fact that S_p has no two incident edges; thus (c) is true.

Conversely, let D be a set fulfilling conditions (a)–(c). Let $H=\{C \mid C=E(v),\ v \in D\}$. For every pair C, $B \in H$ and every $e \in C \cap B$, we have $e \in E-M$ by (c) and we delete e from one of them. Thus, the elements of H are now disjoint, $E = M \cup (E-M) \subseteq M \cup E(D) = M \cup N$ by (b), and $|S \cap C| = 1$ for every $S \in F$, $C \in H$ by (a). Therefore, H is orthogonal to F. □

Let $P=\{v \mid v \in V,\ |E(v) \cap M| = k\}$, $Y=V-P$. In G color the edges of M by white and the edges of $E-M$ by blue. By Lemma 4, an h-covering orthogonal to F exists iff there exists a partition D, $V-D$ of V, with the vertices of Y preassigned to $V-D$, i.e., $D \subseteq P$, such that $G(D)$ contains no white edges and $C(V-D)$ contains no blue edges.

4. Conclusions

In the present paper we define a new problem called 2-CGP which has efficient sequential and parallel algorithms and to which many well-known problems are easily reducible. Moreover, we prove that 2-CGP is log-space complete for NLOGSPACE. We give examples of reductions for many known problems, including 2-CNFSAT and 2-QBFTR.

References

[1] B. Aspvall, M.F. Plass and R.E. Tarjan, A linear-time algorithm for testing the truth of certain quantified Boolean formulas, *Inform. Process Lett.* **8** (1979) 121–123.

[2] M.D. Atkinson and N. Santoro, A practical algorithm for Boolean matrix multiplication, *Inform. Process. Lett.* **29** (1988) 37–38.

[3] S.A. Cook and M. Luby, A simple parallel algorithm for finding a satisfying truth assignment to a 2-CNF formula, *Inform. Process Lett.* **27** (1988) 141–145.

[4] S. Even, A. Itai and A. Shamir, On the complexity of timetable and multicommodity flow problems, *SIAM J. Comput.* **5** (1976) 691–703.

[5] A. Frank, On chain and antichain families of a partially ordered set, *J. Combin. Theory Ser. B* **29** (1980) 176–184.

[6] F. Gavril, Testing for equality between maximum matching and minimum node covering, *Inform. Process. Lett.* **6** (1977) 199–202.

[7] F. Gavril, Algorithms for maximum k-colorings and k-coverings of transitive graphs, *Networks* **17** (1987) 465–470.

[8] N. Immerman, Nondeterministic space is closed under complementation, *SIAM J. Comput.* **17** (1988) 935–938.

[9] N.D. Jones, Y.E. Lien and W.T. Laaser, New problems complete for nondeterministic log space, *Math. Systems Theory* **10** (1976) 1–17.

[10] Z.-Z. Chen, A fast and efficient parallel algorithm for finding a satisfying truth assignment to a 2-CNF formula, *Inform. Process. Lett.* **43** (4) (1992) 191–193.