数理解析研究所講究録
第 754 巻 1991 年 94-103

Using Maximal Independent Sets to Solve Problems in Parallel

Takayoshi Shoudai
Department of Control Engineering and Science
Kyushu Institute of Technology, Iizuka 820, Japan
E-mail:shoudai@ces.kyutech.ac.jp

Satoru Miyano
Research Institute of Fundamental Information Science
Kyushu University 33, Fukuoka 812, Japan
E-mail:miyano@rifis.sci.kyushu-u.ac.jp

Abstract

We consider two kinds of problems: the maximal vertex-induced subgraph problem for a given graph propery π and the minimal set cover problem. We give a unified scheme for parallelizing these problems using the maximal independent set parallel algorithm.

1 Introuduction

The authors have shown that bounded degree maximal subgraph problems are in NC [8] by employing the NC algorithms for the maximal independent set problem (MIS) [3, 4, 5]. This paper extends the technique developed in [8] and shows a way of employing the parallel algorithms for MIS to solve two kinds of problems in which maximal or minimal solutions are searched.

The first problem is to find a maximal set of vertices which induces a subgraph satisfying a given graph property π. The other is the minimal set cover problem that is, given a collection $C = \{c_1, ..., c_m\}$ with $c_i \subset S = \{1, ..., n\}$, to find a collection $C' \subseteq C$ such that every element in S is contained in some $c \in C'$ but no proper subcollection $C'' \subset C'$ does not have this property.

These problems are easily solved in polynomial time by straightforward greedy sequential algorithms. However, these algorithms are hardly parallelizable since they are P-complete [7]: It is shown in [6] that the lexicographically first maximal subgraph problem for a given property π is P-complete if π is hereditary, nontrivial and polynomial-time testable. The same fact also holds for the greedy minimal set cover algorithm.

For the maximal subgraph problem, we need some restrictions on the property to solve the problem in NC. A graph property π is called *local* if the diameter of any minimal graph violating π is bounded by some constant. For such local propery π, we consider the problem of finding a maximal vertex-induced subgraph which satisfies π and, simultaneously, whose maximum vertex degree is at most Δ, where Δ is a given constant. We prove that this problem can be solved in NC by using MIS if π is testable in NC.

For the minimal set cover problem, we also show an algorithm which employs an MIS algorithm. This algorithm can be implemented on an EREW PRAM in time $O(\alpha\beta(\log(n+m))^2)$ using a polynomial number of processors, where $\alpha = \max\{|c_i| \mid i = 1, ..., m\}$ and $\beta = \max\{|d_j| \mid j = 1, ..., n\}$ with $d_j = \{c_i \mid j \in c_i\}$. This implies that if $\alpha\beta = O((\log(n+m))^k)$ then the problem is solvable in NC.

The algorithms for these problems are described by a scheme which applies MIS repeatedly. Thus we do not directly deal with parallelization of the problems. Our concern is how to employ an MIS algorithm to solve problems in parallel.

2 Maximal subgraph problem for a local property

Let π be a property on graphs. We say that a graph $G = (V, E)$ is a *minimal graph violating* π with respect to vertices if G violates π and the vertex-induced subgraph $G[U]$ of U satisfies π for every subset U of V with $U \neq V$. The property π is called *local* with respect to vertices if $\lambda(\pi) = \sup\{\text{diameter}(G) \mid G \text{ is a minimal graph violating } \pi \text{ with respect to vertices}\}$ is finite.

Remark 1 A minimal graph violating a property π with respect to vertices must be connected if π is local.

A property π on graphs is called *hereditary* with respect to vertices if for every graph $G = (V, E)$ satisfying π, the vertex-induced subgraph $G[U]$ also satisfies π for every subset $U \subseteq V$.

Theorem 1 Let π be a graph property which is local and hereditary with respect to vertices. Then a maximal subgraph of a graph $G = (V, E)$ which satisfies π and whose maximum degree is at most Δ can be computed on an EREW PRAM in time $O(\Delta^{\lambda(\pi)}T_\pi(n)(\log n)^2)$ using a polynomial number of processors, where $T_\pi(n)$ is the time needed to decice whether a graph with n vertices satisfies π.

Proof. For subsets W and U of vertices with $W \cap U = \emptyset$, let $E_U^W = \{\{v,w\} \subseteq W \mid dist_{G[U\cup\{v,w\}]}(v,w) \leq \lambda(\pi) \text{ with } v \neq w\}$ and $N_U(w) = \{u \in U \mid dist_{G[U]}(u,w) \leq \lambda(\pi) - 1\}$, where $dist_G(\{v,w\})$ is the length of the shortest path between v and w in G. Then let $H_U^W = (W, E[W] \cup E_U^W)$. The required set U of vertices is computed together with a set W of vertices such that $W \cap U = \emptyset$. Initially let $W = V$ and $U = \emptyset$. At each iteration of the algorithm, a maximal independent set I of H_U^W is computed and added to U while vertices which induce a graph violating π or make the degree of some vertex greater than Δ are deleted from W together with I. This is iterated $\Delta^{\lambda(\pi)}$ times. Formally the algorithm is described as follows:

1 **begin** /* $G = (V, E)$ is an input */
2     $W \leftarrow V$; $U \leftarrow \emptyset$;
3   **while** $W \neq \emptyset$ **do**
4   **begin**
5   Find a maximal independent set I of H_U^W;
6             $U \leftarrow U \cup I$;
7             $W \leftarrow W - I$;
8             $W \leftarrow W - \{w \in W \mid G[U \cup \{w\}] \text{ violates } \pi \text{ or } deg(G[U \cup \{w\}]) > \Delta\}$
9   **end**
10 **end**

We show that this algorithm computes a maximal subset U whose induced subgraph satisfies π and maximum degree is at most Δ.

Let $W_0 = V$ and $U_0 = \emptyset$. Then the graph $H_{U_0}^{W_0}$ is the same as $G = (V, E)$. Therefore in the first iteration, a maximal independent set of G is computed at line 5. For $i = 1, \ldots, \Delta^{\lambda(\pi)}$, let U_i, I_i and W_i be the contents of variables U, I and W at the end of ith iteration, respectively.

Obviously, $W_i \cap U_i = \emptyset$ for $i = 0, \ldots, \Delta^{\lambda(\pi)}$. We assume that the induced subgraph $G[U_{i-1}]$ satisfies π and the maximum degree of $G[U_{i-1}]$ is at most Δ.

Let $\{w, u\}$ be an edge in E with $w \in W_i$ and $u \in U_i$. Line 8 deletes every vertex which is adjacent to more than Δ vertices in U_i or adjacent to a vertex v in U_i with $deg_{G[U_i]}(v) = \Delta$. Therefore u is adjacent to at most Δ vertices in U_i and $deg_{G[U_i \cup \{w\}]}(u) \leq \Delta$. Moreover, $|N_{U_i}(w)| \leq \Delta^{\lambda(\pi)-1}$. Hence, for each w in W_i, we see that

$$A_i(w) = \sum_{u \in N_{U_i}(w)} deg_{G[U_i \cup \{w\}]}(u) \leq \Delta^{\lambda(\pi)}.$$

To show that W becomes empty within $\Delta^{\lambda(\pi)}$ iterations of the while-loop, it suffices to prove that

$$A_i(w) > A_{i-1}(w)$$

for each w in W_i. Since w is not in the maximal independent set I_i of $H_{U_{i-1}}^{W_{i-1}}$ computed by line 5, w is adjacent to a vertex v in $I_i \subseteq W_{i-1}$ via an edge $\{w, v\}$ in $E[W_{i-1}]$ or $E_{U_{i-1}}^{W_{i-1}}$.

Case 1. $\{w, v\} \in E[W_{i-1}]$: Then $\{w, v\}$ is an edge in $G[U_i \cup \{w\}]$. Hence $deg_{G[U_i \cup \{w\}]}(v) \geq 1$. Since $v \in N_{U_i}(w)$ and $v \notin N_{U_{i-1}}(w)$, we see that $A_i(w) \geq A_{i-1}(w) + deg_{G[U_i \cup \{w\}]}(v) > A_{i-1}(w)$.

Case 2. $\{w, v\} \in E_{U_{i-1}}^{W_{i-1}}$: Then there is a path $w, u_1, \ldots, u_{k-1}, v$ with $k \leq \lambda(\pi)$ and $u_j \in U_{i-1}$ $(j = 1, \ldots, k-1)$ in $G[U_{i-1} \cup \{w, v\}]$. Since $v \in W_{i-1}$, $W_{i-1} \cap U_{i-1} = \emptyset$ and $w \neq v$, we see $v \notin U_{i-1} \cup \{w\}$. Hence $\{v, u_{k-1}\}$ is not an edge in $G[U_{i-1} \cup \{w\}]$. On the other hand, v is in U_i and u_{k-1} is in $U_{i-1} \subseteq U_i$. Hence $\{v, u_{k-1}\}$ is an edge in $G[U_i \cup \{w\}]$. Therefore $deg_{G[U_i \cup \{w\}]}(u_{k-1}) > deg_{G[U_{i-1} \cup \{w\}]}(u_{k-1})$. Since $u_{k-1} \in N_{U_{i-1}}(w) \subset N_{U_i}(w)$, we see that $A_i(w) > A_{i-1}(w)$.

We now show that $deg(G[U_i]) \leq \Delta$ and $G[U_i]$ satisfies π.

Claim 1. $deg(G[U_i]) \leq \Delta$.

Proof. For a vertex u in U_{i-1}, if u is adjacent to a vertex w in I_i via an edge in E, then no other vertex in I_i is adjacent to u since I_i is also an independent set with respect to $E_{U_{i-1}}^{W_{i-1}}$. Therefore the degree of u in $G[U_{i-1} \cup I_i]$ remains to be at most Δ since $deg(G[U_{i-1} \cup \{w\}]) \leq \Delta$ by the algorithm. For a vertex u in I_i, $deg_{G[U_{i-1} \cup I_i]}(u)$ is at most Δ since u is adjacent to at most k vertices in U_{i-1} and since I_i is an independent set with respect to $E[W_{i-1}]$. Hence

$deg_{G[U_{i-1} \cup I_i]}(u) \le \Delta$.

Claim 2. $G[U_i]$ satisfies π.

Proof. We assume that $G[U_i]$ does not satisfy π. Then, there is a minimal subset $S \subseteq U_i$ such that $G[S]$ violates π. Since $S \subseteq U_i$ and $U_i = U_{i-1} \cup I_i$, we see that $S = (S \cap U_{i-1}) \cup (S \cap I_i)$. The set $S \cap I_i$ contains at least two vertices since if $S \cap I_i$ consists of only one vertex then line 8 deletes the vertex at the last iteration. Therefore there are two distinct vertices v, w such that $\{v, w\} \in E$ or there are at most $\lambda(\pi) - 1$ vertices in $S \cap U_{i-1}$ which construct a path between v and w since diameter$(G[S]) \le \lambda(\pi)$. For each case, $\{v, w\}$ are in $E[W_{i-1}]$ or $E_{U_{i-1}}^{W_{i-1}}$ since $v, w \in I_i \subset W_{i-1}$. It contradicts the fact that $v, w \in S \cap I_i \subset I_i$ and I_i is a maximal independent set with respect to $E[W_i] \cup E_{U_{i-1}}^{W_{i-1}}$. Hence $G[U_i]$ satisfies π.

Since only vertices which violate the property π or the condition of maximum degree Δ are deleted from W and since π is hereditary, the resulting set U is a maximal subset which induces a subgraph satisfying π when W becomes empty.

MIS can be solved on an EREW PRAM in $O((\log n)^2)$ time using a polynomial number of processors [5]. It is not hard to see that the steps other than MIS can also be implemented on an EREW PRAM in $O((\log n)^2)$ time using a polynomial number of processors. Hence the total algorithm can be implemented using the same amount of time and processors.□

Remark 2 At line 8 of the algorithm, for each $w \in W$, it is sufficient to decide whether $G[N_U(w) \cup \{w\}]$ satisfies π and $deg(G[N_U(w) \cup \{w\}]) \le \Delta$. Therefore, the time needed to compute line 8 depends only on Δ and $\lambda(\pi)$.

Finding a maximal subgraph of maximum degree k takes $O(k^2(\log n)^2)$ time using a polynomial number of processors [8]. This is a special case of Theorem 1 for π = "maximum degree k", $\lambda(\pi) = 2$ and $\Delta = k$. For a graph of maximum degree Δ and π = "k cycle free", it takes $O(\Delta^{\lfloor k/2 \rfloor}(\log n)^2)$ time to find a maximal subgraph satisfing π of maximum degree Δ since $\lambda(\pi) = \lfloor k/2 \rfloor$.

3 Solving the minimal set cover problem using MIS

Let $C = \{c_1, ..., c_m\}$ be a family of subsets of a finite set $S = \{1, ..., n\}$. A subset S' of S is called a *hitting set* for C if $c_i \cap S' \ne \emptyset$ for all $i = 1, ..., m$. A subset S'' of S is called a *co-hitting* *set* if $c_i \not\subseteq S'$ for all $i = 1, ..., m$. We say that C is a *set cover* if $\bigcup_{i=1}^{n} c_i = S$.

It should be noticed that S' is a hitting set for C and only if $S - S'$ is a co-hitting set for C. Therefore, S' is a minimal hitting set for C if and only if $S - S'$ is a maximal co-hitting set for C.

The problem of finding a hitting set is closely related to the set cover problem. For a family $C = \{c_1, ..., c_m\}$ with $\bigcup_{i=1}^{n} c_i = \{1, ..., n\}$, let

$$d_j = \{c_i \mid j \in c_i \in C\}$$

for $j = 1, ..., n$. Then each d_j is not empty. Let $D = \{d_1, ..., d_n\}$ and $C' \subseteq C$ be a minimal hitting set for D. Then $d_j \cap C' \neq \emptyset$ for each $j = 1, ..., n$. Therefore there is some $c_i \in d_j \cap C'$. Thus $j \in c_i$. Hence C' is a set cover of $\{1, ..., n\}$ and also can be seen that C' is minimal.

Theorem 2 Let $C = \{c_1, ..., c_m\}$ be a family of distinct subsets of a finite set $S = \{1, ..., n\}$. Let $\alpha = \max\{|c_i| \mid i = 1, ..., m\}$ and $\beta = \max\{|d_j| \mid j = 1, ..., n\}$, where $d_j = \{c_i \mid j \in c_i\}$. Then a minimal hitting set for C can be computed on an EREW PRAM in time $O(\alpha\beta(\log(n+m))^2)$ using a polynomial number of processors with respect to n and m.

Hence, if $\alpha\beta = O((\log(n+m))^k)$, then a minimal hitting set can be computed in NC.

Proof. We consider the following algorithm that finds a maximal co-hitting set for C_0:

/* A family $C_0 = \{c_1, ..., c_m\}$ with $c_i \subseteq S_0 = \{1, ..., n\}$ for $i = 1, ..., m$ is given. */
/* We assume that $S_0 = \bigcup_{c \in C_0} c$ and $|c_i| \geq 2$ for $i = 1, ..., m$. */

begin
  $S \leftarrow S_0$; $C \leftarrow C_0$;
  $W \leftarrow \emptyset$; /* W gets a maximal co-hitting set */
  while $S \neq \emptyset$ **do**
    begin
      $E \leftarrow \emptyset$;
      par $c \in C$ **do**
        begin
          Choose two distinct vertices v, w from $c \cap S$;
          Add the edge $\{v, w\}$ to E

11 **end**;
12 Find a maximal independent set I of the graph $G = (S, E)$;
13 $W \leftarrow W \cup I$;
14 $S \leftarrow S - I$;
15 $U \leftarrow \{u \in S \mid c \cap S \subseteq W \cup \{u\}$ for some $c \in C\}$;
16 **par** $c \in C$ **do if** $c \cap U \neq \emptyset$ **then** delete c from C;
17 $S \leftarrow S - U$;
18 $V \leftarrow S - \bigcup_{c \in C} c$
19 $W \leftarrow W \cup V$
20 $S \leftarrow S - V$;
21 **end**
22 **end**

The variable W gets a maximal co-hitting set. Let I_i, C_i, U_i, W_i and S_i be the contents of the variables I, C, U, W and S just after the ith iteration of the while-loop, respectively. For convenience, let $W_0 = \emptyset$ and $U_0 = \emptyset$. Let $U_i^* = U_0 \cup \cdots \cup U_i$. We also let E_i be the set of edges constructed during lines 7-11. Then from the algorithm we can easily see that S_0, S_{i-1} and W_i are represented as the following disjoint unions (Figure 1):

(1) $S_i \cup W_i \cup U_i^* = S_0$.

(2) $S_{i-1} = I_i \cup U_i \cup V_i \cup S_i$.

(3) $W_i = W_{i-1} \cup I_i \cup V_i$.

Claim 1. For $c \in C_i$, $c \cap S_i$ contains at least two elements.

Proof. By the assumption on the input, Claim 1 obviously holds for $i = 0$. Assume that the claim holds for i and $S_{i+1} \neq \emptyset$. Let c be in C_i. Then $c \cap U_i = \emptyset$ from line 16 and $c \cap V_i = \emptyset$ from line 18. Therefore from (2) we see that $c \cap S_i = c \cap (S_{i-1} - I_i)$. If $c \cap S_i = \emptyset$, then $U_i = S_{i-1} - I_i$ from line 15. This yields $S_i = \emptyset$ from line 17. This is a contradiction since S_i is assumed not empty. On the other hand, if $c \cap S_i = \{u\}$, then $c \cap S_i \subseteq W_{i-1} \cup I_i \cup \{u\}$. This means that u is in U_i and, therefore, $c \cap U_i \neq \emptyset$, a contradiction. Thus $|c \cap S_i| \geq 2$.

Claim 2. W_i is a co-hitting set for C_0.

Proof. We assume that $S_{i-1} \neq \emptyset$. Obviously, $W_0 = \emptyset$ is a co-hitting set for C_0. Assume that W_{i-1} is a co-hitting set for C_0. Let c be in C_0.

Case 1. $c \notin C_i$: c was deleted during the jth iteration for some $1 \leq j \leq i$. Then $c \cap U_j \neq \emptyset$. Hence there is u in $c \cap U_j \subseteq U_i^*$. By (1) u is not in W_i. Therefore we have $c \not\subseteq W_i$.

Case 2. $c \in C_i$: c is also in C_{i-1}. Then by Claim 1 there are v, w in $c \cap S_{i-1}$ with $v \neq w$ and $\{v, w\} \in E_i$. Since I_i is an independent set, $v \notin I_i$ or $w \notin I_i$. Since W_{i-1} is a co-hitting set for C_0, we have $c \not\subseteq W_{i-1}$. Since no element in S_{i-1}, hence no element in I_i, is in W_{i-1}, v or w is not in $W_{i-1} \cup I_i$. Therefore $c \not\subseteq W_{i-1} \cup I_i$. On the other hand, $c \cap V_i = \emptyset$ by line 18. Therefore $c \not\subseteq W_{i-1} \cup I_i \cup V_i = W_i$.

Claim 3. For any $u \in U_i$, there is $c \in C_{i-1}$ suth that $c \subseteq W_i$.

Proof. By line 15, for $u \in U_i$ there is $c \in C_{i-1}$ such that $c \cap (S_{i-1} - I_i) \subseteq W_{i-1} \cup I_i \cup \{u\}$. Then $c \cap S_{i-1} \subseteq W_{i-1} \cup I_i \cup \{u\}$. Note that for $c \in C_{i-1}$ we have $c \cap U_{i-1}^* = \emptyset$ by line 16. Then

$$\begin{array}{rcll} c & = & c \cap (S_{i-1} \cup W_{i-1} \cup U_{i-1}^* & \text{(by (1))} \\ & = & (c \cap S_{i-1}) \cup (c \cap W_{i-1}) \cup (c \cap U_{i-1}^*) & \\ & \subseteq & W_{i-1} \cup I_i \cup \{u\}. & (\text{by } c \cap U_{i-1}^* = \emptyset) \end{array}$$

Let t be the integer such that $S_t = \emptyset$. Then by (1) $S_0 = W_t \cup U_t^*$. From Claim 2 W_t is a co-hitting set. Claim 3 asserts that for any $u \in U_t^*$ there is some c with $c \subseteq W_t \cup \{u\}$. Therefore W_t is a maximal co-hitting set for C_0.

Claim 4. $t \leq \alpha\beta$.

Proof. For $u \in S_i$, we define

$$B_i(u) = \{v \mid u \neq v \text{ and } \{u, v\} \subseteq c \cap S_i \text{ for some } c \in C_i\}.$$

It is easy to see that $|B_i(u)| \leq \alpha\beta$. Then it suffices to show that

$$|B_i(u)| < |B_{i-1}(u)|$$

for each $u \in S_i$. If $u \in S_i$, then u is not in I_i from line 14. Since I_i is a maximal independent set, there is v with $\{u, v\} \in E_i$. Therefore $\{u, v\} \subseteq c \cap S_{i-1}$ for some $c \in C_{i-1}$. Hence v is in $B_{i-1}(u)$. However, v is not in S_i since v is in I_i. Therefore v is not in $B_i(u)$.

As in the proof of Theorem 1, the part of finding a maximal independent set can be implemented on an EREW PRAM in $O((\log(n+m))^2))$ time using polynomially many processors with respect to n and m. The other steps can also be implemented with at most the same amount of time and processors.□

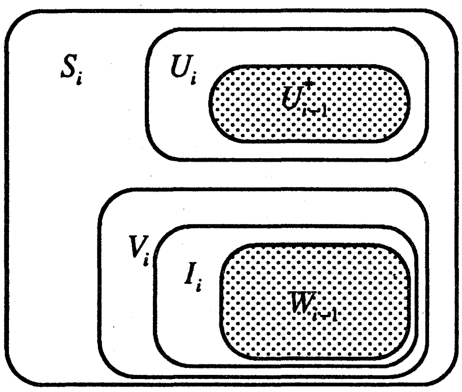

Figure 1: Relation between S_i, I_i, V_i, W_{i-1} and U_i^*

The following corollary is obtained in a straightforward way from Theorem 2:

Corollary 1 Let $C = \{c_1, ..., c_m\}$ be a family of subsets of a finite set $S = \{1, ..., n\}$ such that $S = \bigcup_{i=1}^{m} c_i$. Let $\alpha = \max\{|c_i| \mid i = 1, ..., m\}$ and $\beta = \max\{|d_j| \mid j = 1, ..., n\}$, where $d_j = \{c_i \mid j \in c_i\}$. Then a minimal set cover for S can be computed on an EREW PRAM in time $O(\alpha\beta(\log(n+m))^2)$ using a polynomial number of processors with respect to n and m.

Hence, if $\alpha\beta = O((\log(n+m))^k)$, then a minimal set cover can be computed in NC.

Remark 3 An NC approximation algorithm for the set cover problem is shown in [1]. But it should be noted here that their algorithm does not produce a minimal set cover.

4 Conclusion

We have shown that parallel MIS algorithms are useful to solve the minimal set cover problem and the maximal subgraph problem for a property "local and of degree at most Δ". However, the idea of using MIS does not seem to work for other properties, for example, "acyclic", "planar", which are not local. MIS locates at an interesting position in the NC hierarchy. It is in NC^2 but unlikely to belong to classes such as AC^1 and DET shown in [2]. It is not difficult to see that the algorithms shown in this paper can be transformed to NC^1-reductions to MIS. Hence the results in this paper give some new problems NC^1-reducible to MIS.

References

[1] B. Berger, J. Rompel and P.W. Shor, Efficient NC algorithm for set cover with application to learning and geometry, *Proc. 30th IEEE FOCS* (1989) 54-59.

[2] S.A. Cook, Taxonomy of problems with fast parallel algorithms, *Inform. Contr.* **64** (1985) 2-22.

[3] M. Goldberg and T. Spencer, A new parallel algorithm for the maximal independent set problem, *SIAM J. comput.* **18** (1989) 419-427.

[4] R.M. Karp and A. Wigderson, A fast parallel algorithm for the maximal independent set problem, *J. Assoc. Comput. Mach.* **32** (1985) 762-773.

[5] M. Luby, A simple parallel algorithm for the maximal independent set problem, *Proc. 17th ACM STOC* (1985) 1-10.

[6] S. Miyano, The lexicographically first maximal subgraph problems: P-completeness and NC algorithms, *Math. Systems Theory* **22** (1989) 47-73.

[7] S. Miyano, S. Shiraishi and T. Shoudai, A list of P-complete problems, RIFIS-TR-CS-17, Research Institute of Fundamental Information Science, Kyushu University, 1989 (revised in December, 1990).

[8] T. Shoudai and S. Miyano, Bounded degree maximal subgraph problems are in NC, *Proc. Toyohashi Symposium on Theoretical Computer Science* (1990) 97-101.