On the distributed decision-making complexity of the minimum vertex cover problem


<http://www.numdam.org/item?id=ITA_1996__30_5_431_0>
ON THE DISTRIBUTED DECISION-MAKING COMPLEXITY
OF THE MINIMUM VERTEX COVER PROBLEM (*) (**)

by Pierluigi CRESCENZI and Luca TREVISAN (1)

Communicated by AUZIELLO

Abstract. — In this paper we study the problem of computing approximate vertex covers of a graph on the basis of partial information in the distributed decision-making model proposed by Deng and Papadimitriou [1]. In particular, we show an optimal algorithm whose competitive ratio is equal to \( p \), where \( p \) is the number of processors.

Résumé. — Dans cet article nous étudions le problème de trouver des couvertures approchées d’arêtes d’un graphe en partant d’une information partielle dans le modèle de prise de décision distribuée proposé par Deng et Papadimitriou [1]. En particulier nous donnons un algorithme optimal dont le taux de concurrence est \( p \) où \( p \) est le nombre de processeurs.

1. INTRODUCTION

The minimum vertex cover (MVC) problem consists of finding, given a graph \( G \), a minimum cardinality set of nodes \( V' \) such that, for any edge \((u, v)\), either \( u \in V' \) or \( v \in V' \). This is a well-studied problem which appeared in the first list of NP-complete problems presented by Karp [4]. A straightforward approximation algorithm, based on the idea of a maximal matching, was successively developed by Gavril (according to [2]) with a performance ratio no greater than 2. Several other approximation algorithms are presented in the lecture notes of Motwani [5]. In this paper we analyse the complexity of finding approximate solutions for the MVC problem in the framework of distributed decision-making with incomplete information.
In particular, we assume that the vertex cover is chosen by independent processors, each knowing only a part of the graph and acting in isolation. More specifically, we assume that the adjacency list of each node of the graph is known by only one processor which has to decide whether the node should belong to the vertex cover. We then want to develop distributed algorithms that always produce feasible solutions (that is, vertex covers) and achieve, in the worst case, a reasonable competitive ratio (that is, the ratio of the cardinality of the solution computed by the algorithm to the optimum cardinality should be as small as possible). In this paper, we show that a simple double-matching algorithm which essentially performs Gavril's algorithm first on the "bridge" edges and then on the "inner" edges achieves a competitive ratio equal to $p$ where $p$ is the number of processors. We also show, by means of a quite involved counting technique, that this algorithm is optimal, that is, no distributed algorithm can achieve a ratio smaller than $p$. These results fit into a more general context in which an optimization problem has to be solved in a distributed fashion and neither a centralized control nor a complete information are available (see [3] for several applications). Moreover, it has been argued that this kind of results "can be seen as part of a larger project aiming at an algorithmic theory of the value of information" [8]. Intuitively, this theory should allow to compare in terms of competitive ratios two different information regimes, that is, two different ways of distributing the input among the processors.

2. THE MODEL

We consider a distributed system formed by $p$ non-communicating processors $P_1, \ldots, P_p$. Given an instance $I$ of an optimization problem $\Pi$, we assume that $I$ is encoded as a set of objects $I = \{w_1, \ldots, w_n\}$. The instance is "distributed" among the processors according to a certain criterion, so that the processors $P_i$ receives a subset $I_i \subseteq I$ and computes a partial solution $S_i$ which depends only on $I_i$ and $i$. A measure function $u$ gives the value $u(S_1, \ldots, S_p, I)$ of the partial solutions $S_1, \ldots, S_p$ (without loss of generality, we assume that $u$ is undefined whenever $S_1, \ldots, S_p$ do not result into a feasible solution).

More formally, an information regime (for a $p$-processor system) is a function $\mathcal{R}$ that, given an instance $I$, returns a $p$-tuple of subinstances $I_1, \ldots, I_p$. The $p$-tuple $I_1, \ldots, I_p$ is also called a distributed instance. A decision strategy $\mathcal{A}$ is a $p$-tuple of algorithms $A_1, \ldots, A_p$. The competitive ratio of a strategy $\mathcal{A}$ with respect to a regime $\mathcal{R}$ is defined as follows.
THE DISTRIBUTED DECISION-MAKING COMPLEXITY

\[ R(A, \mathcal{R}) = \max_I \max \left\{ \frac{u(A_1(I_1), \ldots, A_p(I_p))}{\text{opt}(I)}, \frac{\text{opt}(I)}{u(A_1(I_1), \ldots, A_p(I_p))} \right\} \]

where \( I_1, \ldots, I_p = \mathcal{R}(I) \) and \( \text{opt}(I) \) is the value of an optimum solution for \( I \). Observe that \( R(A, \mathcal{R}) \geq 1 \) for any strategy \( A \) and any information regime \( \mathcal{R} \), and that \( R(A, \mathcal{R}) \) is as close to one as the solutions computed by \( A \) are close to the optimum.

This definition can be easily generalized to a family \( \mathcal{F} \) of information regimes as follows:

\[ R(A, \mathcal{F}) = \max_{\mathcal{R} \in \mathcal{F}} R(A, \mathcal{R}). \]

The competitive ratio of a problem \( \Pi \) in a \( p \)-processors system, with respect to a family \( \mathcal{F} \) is

\[ R(\Pi, \mathcal{F}) = \min_A R(A, \mathcal{F}). \]

It has been argued that such a ratio is a reasonable measure of the value of the information that has been distributed to the processors according to the information regimes in \( \mathcal{F} \).

For any optimization problem, two approaches are interesting within such a framework:

1. Fix a natural family of “homogeneous” information regimes, usually those with fewer redundancy, and try to characterize as tightly as possible the competitive ratio (that is, the value of information with respect to such a distribution scheme).

2. Consider several (families of) different information regimes and show the existence of trade-offs between redundancy and competitiveness.

In this paper we will study the MVC problem within this framework following the first approach.

2.1. The distributed MVC problem

In the following we will identify a graph \( G = (V, E) \) with the set \( L \) of its adjacency lists and we will consider the family \( \mathcal{F}_{\text{part}} \) of information regimes that partition \( L \) into disjoint sets \( L_1, \ldots, L_p \).

A distributed strategy \( A = A_1, \ldots, A_p \) for the MVC problem is a \( p \)-tuple of algorithms with the following property. For any graph \( G = (V, E) \), for
any information regime \( \mathcal{R} \), such that \( \mathcal{R}(G) = L_1, \ldots, L_p \), \( A_i(L_i) \subset V_i \) and \( A(G, \mathcal{R}) = \bigcup_{i=1}^{p} A_i(L_i) \) is a vertex cover for \( G \), where \( V_i \) denotes the set of nodes whose adjacency lists are in \( L_i \).

The competitive ratio of \( A \) with respect to the family \( \mathcal{F}_{\text{part}} \) is

\[
R(A, \mathcal{F}) = \max_{\mathcal{R} \in \mathcal{F}_{\text{part}}} \frac{|A(G, \mathcal{R})|}{\text{opt}(G)}.
\]

In the following sections we will show that \( \min_{A} R(A, \mathcal{F}_{\text{part}}) = p. \)

3. THE UPPER BOUND

Recall that Gavril’s algorithm looks for a maximal matching in the graph and then returns both the endpoints of any edge in the matching. It is easy to see that such a set of nodes is a vertex cover and that its cardinality is at most twice the cardinality of the minimum cover.

In the next theorem we will apply the same idea first to edges “shared” by two processors and then to the remaining edges.

Input: \( L_i \)
\( \{V_i \) denotes the set of nodes whose adjacency lists are in \( L_i \}\)

begin
  \( C_i := \emptyset; \ B_i := \emptyset; \)
  for each edge \((u, v)\) such that \( u \in V_i \) and \( v \notin V_i \) do
    if \( u \notin C_i \) and \( v \notin B_i \) then
      begin
        \( C_i := C_i \cup \{u\}; \)
        \( B_i := B_i \cup \{v\}; \)
      end;
  for each edge \((u, v)\) such that \( u, v \in V_i \) do
    if \( u \notin C_i \) and \( v \notin C_i \) then
      \( C_i := C_i \cup \{u, v\}; \)
  return \( C_i \)
end.

Figure 1. – The double-matching algorithm.
THEOREM 1: For any $p \geq 2$, a strategy $A$ for a $p$-processor system exists whose competitive ratio is at most $p$.

Proof: Consider the strategy $A = \{A_1, \ldots, A_p\}$ where algorithm $A_i$ is described in figure 1, and assume that the edges are picked in any specified order in the boldface for loops (e.g. in lexicographic order).

Let $G$ be an input graph and $\mathcal{R}$ be an information regime in $\mathcal{F}_{\text{part}}$. We denote by $A_i^{(1)}(L_i)$ and $A_i^{(2)}(L_i)$ the set of nodes included in $C_i$ during the first and the second for instruction, respectively. Clearly, all edges "seen" by processor $P_i$ are covered by the set $A_i^{(1)}(L_i) \cup A_i^{(2)}(L_i) \cup B_i$. Then, in order to prove that $A(G, \mathcal{R})$ is a vertex cover for $G$ it suffices to show that, for any $i$,

$$B_i \subseteq \bigcup_{k=1}^{p} A_k^{(1)}(L_k).$$

(1)

The proof is by induction on the number $b$ of bridge edges, that is, edges whose endpoints "belong" to different processors (observe that each $B_i$ contains only endpoints of bridge edges). If $b = 0$, then the proof is trivial. Suppose that we have $b + 1$ bridge edges and that $(u, v)$ is the last of these edges in the lexicographic order with $u \in V_i$ and $v \in V_j$. Let $\tilde{L}_i$ and $\tilde{L}_j$ denote the adjacency lists obtained from $L_i$ and $L_j$, respectively, by deleting the edge $(u, v)$, moreover, let $A_i^{(1)}(\tilde{L}_i)$ and $\tilde{B}_i$ (respectively, $A_j^{(1)}(\tilde{L}_j)$ and $\tilde{B}_j$) be the sets computed by the algorithm on input $\tilde{L}_i$ (respectively, $\tilde{L}_j$).

By induction hypothesis,

$$\tilde{B}_i, \tilde{B}_j \subseteq \tilde{A}^{(1)} = \bigcup_{k=1}^{p} A_k^{(1)}(L_k) \cup A_i^{(1)}(\tilde{L}_i) \cup A_j^{(1)}(\tilde{L}_j).$$

We shall now prove that $B_i \subseteq A^{(1)}$ (the proof for $B_j$ is similar). To this aim, we distinguish the following two cases.

1. $u \in A_i^{(1)}(\tilde{L}_i) \lor v \in \tilde{B}_i$: in this case $B_i = \tilde{B}_i \subseteq \tilde{A}^{(1)} \subseteq A^{(1)}$.

2. $u \notin A_i^{(1)}(\tilde{L}_i) \land v \notin \tilde{B}_i$: in this case $B_i = \tilde{B}_i \cup \{v\}$ and $u \notin \tilde{B}_j$ (since $u \notin \tilde{A}^{(1)}$ and $\tilde{B}_j \subseteq \tilde{A}^{(1)}$). If $v \in A_j^{(1)}(\tilde{L}_j)$ then, clearly, $B_i \subseteq \tilde{A}^{(1)} \subseteq A^{(1)}$.

We have thus shown that $A(G, \mathcal{R})$ is a vertex cover. In order to prove that its competitive ratio is at most $p$, let $n_k = \sum_{i=1}^{p} |A_i^{(k)}(L_i)|$ for $k = 1, 2$. 

vol. 30, n° 5, 1996
Clearly, an index \( i \) must exist such that \( |A_i^{(1)}(L_i)| \geq n_1/p \). This set \( A_i^{(1)}(L_i) \) then corresponds to a set of at least \( n_1/p \) disjoint edges. Moreover, the set \( \bigcup_{i=1}^p A_i^{(2)}(L_i) \) corresponds to another set of \( n_2/2 \) disjoint edges. From (1) it also follows that the union of these two sets is still a set of disjoint edges. That is, \( G \) contains a matching of at least \( n_1/p + n_2/2 \) edges. Thus, any vertex cover for \( G \) must contain at least \( n_1/p + n_2/2 \geq (n_1 + n_2)/p \) nodes, that is,

\[
\frac{|A(G, R)|}{\text{opt}(G)} \leq \frac{n_1 + n_2}{(n_1 + n_2)/p} = p.
\]

We can conclude that the competitive ratio of \( \mathcal{A} \) is at most \( p \). \( \square \)

4. THE LOWER BOUND

In order to prove that the result of the previous section is tight, let us first show that, for any strategy \( \mathcal{A} \), an information regime \( R \in \mathcal{F}_{\text{part}} \) exists such that \( R(\mathcal{A}, R) \geq 2 \). Let \( K_{i,j}^{n,n} \) denote the distributed instance in which the complete bipartite graph \( K_{n,n} \) with vertex classes \( U \) and \( W \) is distributed in the following way: \( V_i = U, V_j = W, \) and all other processors know nothing (recall that \( R(K_{n,n}) = L_1, \ldots, L_p \) and, for any \( k \), \( V_k \) is the set if nodes whose adjacency lists are in \( L_k \)). Then, for any strategy \( \mathcal{A} \), either \( P_i \) or \( P_j \) has a choose all its nodes when running \( \mathcal{A} \) with input \( K_{i,j}^{n,n} \) (otherwise, an uncovered edge exists). Without loss of generality, we can assume that \( P_i \) chooses all its nodes. Let us then consider the new distributed instance in which the vertices in \( W \) are pairwise connected, thus forming a clique of order \( n \). Clearly, \( P_i \) still chooses all its nodes since its subinstance is not changed. Moreover, \( P_j \) is also forced to choose at least \( n - 1 \) of its nodes. The optimum solution then contains \( n \) nodes while the solution computed by \( \mathcal{A} \) contains at least \( 2n - 1 \) nodes. That is, the competitive ratio is at least 2.

In order to increase the above lower bound, we will show in the next theorem how to find, for any strategy \( \mathcal{A} \), and for infinitely many \( n \), a distributed instance \( G \) in which a processor \( P_j \) knows \( n \) nodes and the other processors \( P_i \) share at least \( (p - 1)(n - 1) \) nodes which are all connected to the \( n \) nodes of \( P_j \). Moreover, each \( P_i \) with \( i \neq j \) chooses all its nodes when running strategy \( \mathcal{A} \) with input \( G \). We can then modify the instance by pairwise connecting all nodes of \( P_j \). The optimum thus contains \( n \) nodes while the solution computed by \( \mathcal{A} \) contains at least \( p(n - 1) \) nodes. That is, the competitive ratio is at least \( p \).
In order to prove the theorem, we need the following technical result which intuitively states that, given two sequences of integers to be interpreted both as "values" and as "indices", the sum of the two sequences must be sufficiently large if, for any element of one sequence whose value is "small", this element is the starting index of a subsequence of "large" values in the other sequence.

**Lemma 1:** Let \( a_1, \ldots, a_N, b_1, \ldots, b_N \) be \( 2N \) nonnegative numbers such that, for any \( n \),
1. \( 0 \leq a_n, b_n \leq N \).
2. If \( a_n < n - 1 \), then \( b_k \geq n \) for \( k = a_n + 1, \ldots, n - 1 \).
3. If \( b_n < n - 1 \), then \( a_k \geq n \) for \( k = b_n + 1, \ldots, n - 1 \).

Then

\[
\sum_{n=1}^{N} (a_n + b_n) \geq 2 \sum_{n=1}^{N} (n - 1). \tag{2}
\]

**Proof:** We proceed by induction on \( N \). For \( N = 1 \) the proof is trivial since both \( a_1 \) and \( b_1 \) are nonnegative.

Assume that (2) has been proven for any \( N' < N + 1 \) and let \( a_1, \ldots, a_{N+1}, b_1, \ldots, b_{N+1} \) be \( 2(N + 1) \) nonnegative numbers satisfying the hypothesis of the lemma. Let us consider the case in which both \( a_{N+1} \) and \( b_{N+1} \) are smaller than \( N \) (the other cases are proved similarly). Then \( a_{N+1} = N - h \) and \( b_{N+1} = N - k \) with \( h, k > 0 \). From the hypothesis it follows that

\[
a_{N-k+1}, \ldots, a_N = N + 1 \quad \text{and} \quad b_{N-h+1}, \ldots, a_N = N + 1.
\]

For any \( n \) with \( N - k + 1 \leq n \leq N \) and for any \( m \) with \( N - k + 1 \leq m \leq N \), let us define \( a'_n = N \) and \( b'_m = N \). The \( 2N \) numbers \( a_1, \ldots, a_{N-k}, a'_{N-k+1}, \ldots, a'_N, b_1, \ldots, b_{N-h}, b'_{N-h+1}, \ldots, b'_N \) clearly still satisfy the hypothesis of the lemma. This, in turn, implies that

\[
\sum_{n=1}^{N+1} (a_n + b_n) = \sum_{n=1}^{N} (a'_n + b'_n) + (h + k) + (a_{N+1} + b_{N+1}) \\
\geq 2 \sum_{n=1}^{N} (n - 1) + (h + k) + (a_{N+1} + b_{N+1}) \\
= 2 \sum_{n=1}^{N+1} (n - 1)
\]
where the inequality is due to the inductive hypothesis. The lemma thus follows.

We are now in a position to prove the main result of this section.

**Theorem 2:** For any strategy \( \mathcal{A} = \{A_1, \ldots, A_p\} \), and for any integer \( N_0 \), a graph \( G \), an information regime \( \mathcal{R} \in \mathcal{F}_{\text{part}} \), an index \( j \), and an integer \( n_0 > N_0 \) exist such that

1. \( |V_j| = n_0 \).
2. \( \sum_{i \neq j} |V_i| \geq (p - 1)(n_0 - 1) \).
3. For any \( i \neq j \), each vertex in \( V_i \) is connected to each vertex in \( V_j \).
4. For any \( i \neq j \), \( A_i(L_i) = V_j \).

**Proof:** For any \( i, j, m, \) and \( n \), let \( K_{i,j}^{m,n} \) denote the distributed instance in which the complete bipartite graph \( K_{m,n} \) with vertex classes \( U \) and \( W \) is distributed in the following way: \( V_i = U \), \( V_j = W \), and all other processors know nothing. For any integer \( n \), let \( c_{i,j}^n \) be the maximum \( m \) such that \( P_i \) with input \( K_{i,j}^{m,n} \) chooses all its nodes (see fig. 2 where the black nodes have been chosen and the white nodes may or may not have been chosen).

![Figure 2. - The definition of \( c_{i,j}^n \) (Diagrams showing the distribution of nodes)](image)

Observe that, for any \( i, j, \) and \( n \), if \( c_{i,j}^n = m < n - 1 \) then \( c_{j,i}^k \geq n \) for \( k = m + 1, \ldots, n - 1 \): from Lemma 1, we have that for any \( i, j, \) and \( N \), the following inequality holds:

\[
\sum_{n=1}^{N} (c_{i,j}^n + c_{j,i}^n) \geq 2 \sum_{n=1}^{N} (n - 1).
\]
It then follows that

\[ \sum_{n=1}^{N} \sum_{j=1}^{p} \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^n = \sum_{i,j=1 \atop i < j}^{p} \sum_{n=1}^{N} (c_{i,j}^n + c_{j,i}^n) \geq p(p-1) \sum_{n=1}^{N} (n-1). \quad (3) \]

Assume now that an \( N_0 \) exists such that, for any \( n > N_0 \) and for any \( j \),

\[ \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^n < (p-1)(n-1) \]

and let

\[ \sigma = \sum_{n=1}^{N_0} \sum_{j=1}^{p} \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^n, \]

Then, for any \( N > N_0 \), we have that

\[ \sum_{n=1}^{N} \sum_{j=1}^{p} \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^n = \sigma + \sum_{n=N_0+1}^{N} \sum_{j=1}^{p} \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^n \]

\[ \leq \sigma + \sum_{n=N_0+1}^{N} \sum_{j=1}^{p} [(p-1)(n-1) - 1] \]

\[ = \sigma + p(p-1) \sum_{n=N_0+1}^{N} (n-1) - (N-N_0)p \]

which, for \( N \) sufficiently large, contradicts (3).

Thus, for any integer \( N_0 \), an index \( j \) and an integer \( n_0 > N_0 \) exist such that

\[ \sum_{i=1 \atop i \neq j}^{p} c_{i,j}^{n_0} \geq (p-1)(n_0 - 1). \]
The distributed instance $G$ is then defined as a star of bipartite graphs in which processor $P_j$ knows $n_0$ nodes and each processor $P_i$ with $i \neq j$ knows $c_{i,j}^{n_0}$ nodes which are all connected to each node of $P_j$ (2). Clearly, this graph satisfies the theorem. 

As a consequence of the above theorem, we then have the following result, which states the optimality of the upper bound shown in the previous section.

**Corollary 1:** $R(MVC, \mathcal{F}_{\text{part}}) \geq p$.

5. CONCLUSION AND OPEN PROBLEMS

We studied the problem of computing approximate vertex covers of a graph on the basis of partial information. We showed an optimal algorithm whose competitive ratio is equal to the number of processors.

In this paper we fixed a particular family of information regimes: it would be interesting to consider other families of information regimes and find trade-offs between competitive ratio and redundancy.

The algorithm given in Section 3 runs in polynomial time, even if the lower bound holds for algorithms of unbounded complexity. In general, however, the competitive ratio of a problem may increase if we restrict ourselves to polynomial-time algorithms. Investigating the relationship between time complexity and competitive ratio may be an interesting direction for further research.

In particular, in the case of the minimum vertex cover problem, given full information, no polynomial time algorithm is known to achieve a better asymptotic performance ratio than 2: it would be interesting to find a non-trivial information regime where a competitive ratio equal to 2 is achievable in polynomial time.

ACKNOWLEDGEMENTS

We wish to thank Alberto Marchetti-Spaccamela for several helpful discussions.

---

(2) To be more precise, we should assume that, for any $k$, each vertex assigned to processor $P_k$ has a "name" depending on the value of $k$, such that no two distinct processors own two nodes with the same name. For instance, we can assume that if processor $P_k$ has $n_k$ nodes, then their identity is $k, p + k, \ldots, (n_k - 1)p + k$. 

Informatique théorique et Applications/Theoretical Informatics and Applications
REFERENCES