Edge cover by bipartite subgraphs

Marie-Christine Plateau

CEDRIC-CNAM, 292 rue Saint-Martin, Paris, France

Leo Liberti

LIX, École Polytechnique, F-91128 Palaiseau, France

Laurent Alfandari

ESSEC, Av. Bernard Hirsch, BP105 95021, Cergy Pontoise, France

Key words: Bipartite subgraph, edge cover, integer linear programming, heuristics.

1 Introduction

We consider the following optimization problem.

MINIMUM BIPARTITE GRAPH COVER (MBGC). Given a connected undirected graph $G = (V, E)$ without loops or parallel edges, find a family $\{H_k = (A_k, B_k, E_k) \mid k \leq m\}$ of (not necessarily induced nor complete) connected bipartite subgraphs of G such that $E = \bigcup_{k \leq m} E_k$ and m is minimum.

Two related and reasonably well-studied problems are the MINIMUM BICLIQUE COVER (MBC), where H_k are required to be bicliques (i.e. complete bipartite subgraphs) [4,3,2,1] and the MINIMUM CUT COVER (MCC), where H_k are cutsets, namely not required to be connected. Both problems are **NP**-hard. To the best of our knowledge, whether the MBGC is **NP**-hard or not is currently unknown.

Let $G = (V, E)$ be an undirected graph. For $v \in V$, we denote by $\delta(v)$ the set of vertices u such that $\{v, u\} \in E$, and by $\bar{\delta}(v)$ the set of edges $e \in E$ adjacent to v. With respect to a set of edges $F \subset E$, $\delta_F(v)$ is the set of vertices adjacent to v using edges in F, and $\bar{\delta}_F(v)$ is the set of edges $e \in F$ adjacent to v.

Email addresses: mc.plateau@cnam.fr (Marie-Christine Plateau), liberti@lix.polytechnique.fr (Leo Liberti), alfandari@essec.fr (Laurent Alfandari).

Preprint submitted to CTW 2007 *27 February 2007*

2 Mathematical Formulation

2.1 Decision variables

Let $\bar{m}$ be an upper bound to m, for example $\bar{m} = \lceil \frac{n}{2} \rceil$. We consider four sets of binary variables and a set of continuous multicommodity flow variables:

$$\forall k \leq \bar{m} \quad y_k = \begin{cases} 1 & \text{if the } k\text{-th bipartite subgraph is in the cover} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$$\forall i \in V, j \in V, k \leq \bar{m} \quad e_{ij}^k = \begin{cases} 1 & \text{if edge } (i,j) \text{ belongs to the } k\text{-th bipartite subgraph} \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

$$\forall i \in V, k \leq \bar{m} \quad a_i^k = \begin{cases} 1 & \text{if node } i \text{ is in } A_k \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

$$\forall i \in V, k \leq \bar{m} \quad b_i^k = \begin{cases} 1 & \text{if node } i \text{ is in } B_k \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

$$\forall \{i,j\} \in E, k \leq \bar{m}, \quad u \in V, v \in V : u \neq v \quad f_{ij}^{uvk} \in [0,1] \tag{5}$$

where the continuous flow variables f identify a path connecting u and v in the k-th bipartite subgraph in order to ensure that bipartite subgraphs are connected, i.e., $f_{ij}^{uvk} = 1$ if edge $(i,j) \in E$ belongs to the path connecting u and v, 0 otherwise.

2.2 Objective function

The objective is to minimize the number of subgraphs in the cover:

$$\min \sum_{k=1}^{\bar{m}} y_k \tag{6}$$

2.3 Global covering constraints

The global constraints linking all subgraphs are the edge covering constraints:

$$\forall\ (i,j) \in E : i < j \quad \sum_{k=1}^{\bar{m}} e_{ij}^k \geq 1 \tag{7}$$

The remaining constraints are local constraints defining for each index k a valid subgraph (i.e., bipartite and connected).

2.4 Local Bipartite constraints

Constraints (8),(9),(10),(11) ensure that subgraphs are bipartite.

$$\forall k = 1, \ldots, \bar{m}, \forall i \in V : \quad a_i^k + b_i^k \leq y^k \tag{8}$$

The above constraints have a double function: first, they ensure that every node in subgraph k is whether in A^k or B^k but not both, second, they translate the logical constraints that if node i is in subgraph k then $y^k = 1$. We have moreover:

$$\forall\, k = 1, \ldots, \bar{m}, (i,j) \in E : \quad a_i^k + a_j^k \leq 2 - e_{ij}^k \tag{9}$$

$$\forall\, k = 1, \ldots, \bar{m}, (i,j) \in E : \quad b_i^k + b_j^k \leq 2 - e_{ij}^k \tag{10}$$

$$\forall\, k = 1, \ldots, \bar{m}, i \in V : \quad a_i^k + b_i^k + a_j^k + b_j^k \geq 2e_{ij}^k \tag{11}$$

These constraints ensure that if $e_{ij}^k = 1$, then we have whether $a_i^k = 1, b_i^k = 0, a_j^k = 0, b_j^k = 1$ or $a_i^k = 0, b_i^k = 1, a_j^k = 1, b_j^k = 0$. This eliminates odd-length cycles as these cycles would have an edge $e_{ij}^k = 1$ with $a_i^k = a_j^k = 1$ or $b_i^k = b_j^k = 1$, so subgraph k is bipartite indeed.

2.5 Local connectivity constraints

Define $A = \{(i,j),(j,i) : (i,j) \in E\}$, i.e. edges are transformed in two inversed arcs. The multicommodity flow constraints below ensure that each bipartite subgraph is connected:

$$\forall\, u \in V, v \in V, k \leq \bar{m} : u \neq v \qquad \sum_{j \in V:(i,j) \in A} f_{uj}^{uvk} \geq a_u^k + b_u^k + a_v^k + b_v^k - 1 \tag{12}$$

$$\forall\, u \in V, v \in V, k \leq \bar{m} : u \neq v \qquad \sum_{i \in V:(i,u) \in A} f_{uj}^{uvk} = 0 \tag{13}$$

$$\forall\, u \in V, v \in V, k \leq \bar{m} : u \neq v \qquad \sum_{i \in V:(i,v) \in A} f_{iv}^{uvk} \geq a_u^k + b_u^k + a_v^k + b_v^k - 1 \tag{14}$$

$$\forall\, u \in V, v \in V, k \leq \bar{m} : u \neq v \qquad \sum_{j \in V:(v,j) \in A} f_{vj}^{uvk} = 0 \tag{15}$$

$$\forall\, (u,v) \in V^2, k \leq \bar{m}, j \in V : u \neq v, j \neq u, v \qquad \sum_{i \in V:(i,j) \in A} f_{ij}^{uvk} = \sum_{l \in V:(j,l) \in A} f_{jl}^{uvk} \tag{16}$$

Constraints (12),(13),(14),(15) and (16) ensure that, if both nodes u and v are in subgraph k (in that case in constraints (12) and (14) the right-hand term $a_u^k + b_u^k + a_v^k + b_v^k - 1$ is one), then a single flow unit leaves u and finally (by constraints (16)) arrives at node v, hence defining a path connecting u and v. Constraints (13) and (15) are necessary to ensure the the flow is not composed of two disconnected cycles, one passing through u and the other one through v.

We denote this integer program by (BGC).

Lemma:
(BGC) allows to find the optimal sequence $\{H_k = (A_k, B_k, E_k) \mid k \leq m\}$, that is one of the minimal subset of bipartite graph cover for the edges of G.

Variants of the model introducing new variables are proposed and compared on a set of graph instances.

3 Efficient heuristic

We mean to find a sequence $\{H_k \mid k \leq m\}$ of bipartite subgraphs of G such that the union of all their edges covers the edges E of G. Additionally, we would like to minimize m. We propose the simple but effective heuristic in Alg. 1; it finds a set of bipartite graphs $H_k = (A_k, B_k, E_k)$ with the required properties. Let $\bar{m}$ be an upper bound to m.

Algorithm 1 Fast heuristic for bipartite graph cover.

```
Initialize all H_k to ∅ for k ≤ m̄.
Let k = 1.
while E ≠ ∅ do
  if A_k = B_k = ∅ then
    Let U = Z = V. (1)
  else
    Let U = {v ∈ Z | ∀u ∈ A_k {u, v} ∉ E ∧ δ_E(v) ∩ B_k ≠ ∅}.
  end if
  if U = ∅ then
    Set k ← k + 1. (2)
  else
    Let v ∈ U s.t. |δ_E(v)| is maximum.     (3)
    Set A_k ← A_k ∪ {v},
        B_k ← B_k ∪ δ_E(v),
        Z ← Z ∖ ({v} ∪ δ_E(v)),
        E_k ← E_k ∪ δ̄_E(v),
        E ← E ∖ δ̄_E(v).
  end if
end while
Let m = k.
```

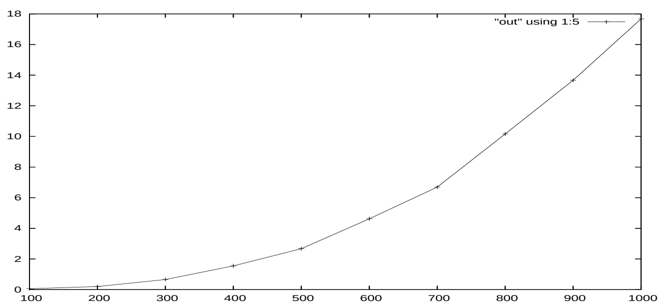
The heuristic works by constructing the k-th bipartite graph (A_k, B_k, E_k) in the cover. It progressively selects the vertex v with highest star degree, disconnected from A_k but whose star intersects B_k; v is added to A_k, its vertex star to B_k and its edge star to E_k. When no further vertices may be added to A_k, k is increased and the procedure is repeated with a smaller edge set $E = E \smallsetminus E_k$. When $E = \emptyset$, the cover is complete. The worst-case complexity of Algorithm 1 in a naive implementation is $O(|V|^3|E|)$.

Lemma 1 *The sequence* $\{H_k = (A_k, B_k, E_k) \mid k \leq m\}$ *found by Algorithm 1 is a bipartite graph cover for the edges of* G.

Proof. By inspection it is easy to see that at termination, the algorithm provides a bipartite graph cover for the edges of G. It remains to be shown that the algorithm terminates, namely that k never increases twice consecutively in Step (2) without $|E|$ decreasing. This is easily shown as follows: at the iteration following the increase in k, we have $A_k = B_k = \emptyset$, whence $U = Z = V$ in Step (1). Since $U \neq \emptyset$ and $E \neq \emptyset$ (since otherwise the algorithm would have already terminated), there is a $v \in U$ such that $|\delta_E(v)| \geq 1$ in Step (3). Thus $|E|$ is decreased. □

The following table and corresponding CPU time vs. $|V|$ plot illustrate the performance of Alg. 1 on a set of 10 randomly generated undirected graphs where each edge has unit cost and is generated with probability 0.5. All experiments were carried out on an Intel Core Duo 1.2GHz with 1.5 GB RAM running the Linux operating system.

$\|V\|$	B	user CPU time (s)
100	20	0.06
200	32	0.20
300	45	0.66
400	56	1.55
500	68	2.67
600	81	4.62
700	91	6.69
800	101	10.16
900	111	13.67
1000	121	17.67



References

[1] G. Alexe, S. Alexe, Y. Crama, S. Foldes, P. Hammer, and B. Simeone. Consensus algorithms for the generation of all maximal bicliques. *Discrete Applied Mathematics*, 145:11–21, 2004.

[2] J. Amilhastre, M.C. Vilarem, and P. Janssen. Complexity of minimum biclique cover and minimum biclique decomposition for bipartite domino-free graphs. *Discrete Applied Mathematics*, 86:125–144, 1998.

[3] H. Müller. On edge perfectness and classes of bipartite graphs. *Discrete Mathematics*, 149:159–187, 1996.

[4] J. Orlin. Contentment in graph theory: Covering graphs with cliques. *Proceedings of the Koninklijke Nederlandse Akademie van Weteschappen, Series A*, 80(5):406–424, 1977.