Finding bipartite subgraphs efficiently

Dhruv Mubayi[*] and György Turán[†]

Abstract

Polynomial algorithms are given for the following two problems:

- given a graph with n vertices and m edges, find a complete balanced bipartite subgraph $K_{q,q}$ with $q = \lfloor \frac{\ln n}{\ln(2en^2/m)} \rfloor$,
- given a graph with n vertices, find a decomposition of its edges into complete balanced bipartite graphs having altogether $O(n^2/\ln n)$ vertices.

The first algorithm can be modified to have running time linear in m and find a $K_{q',q'}$ with $q' = \lfloor q/5 \rfloor$. Previous proofs of the existence of such objects, due to Kővári-Sós-Turán [10], Chung-Erdős-Spencer [5], Bublitz [4] and Tuza [13] were non-constructive.

1 Introduction

Determining the minimal number of edges in a bipartite graph which guarantees the existence of a complete balanced bipartite subgraph $K_{q,q}$ is known as the Zarankiewicz problem (see, e.g., Bollobás [3]). It was shown by Kővári, Sós and Turán [10] that every bipartite graph with n vertices in both sides and $c_q n^{2-1/q}$ edges contains a $K_{q,q}$. The same bound (with different constant c_q) holds for general n-vertex graphs. The argument from [10] also shows that n-vertex graphs of constant density, i.e., graphs with ϵn^2 edges,

[*] Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, IL 60607. Email: mubayi@math.uic.edu. Research supported in part by NSF grant DMS 0653946.

[†] Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, IL 60607 and Research Group on Artificial Intelligence, Hungarian Academy of Sciences, University of Szeged. Email: gyt@uic.edu. Research supported in part by NSF grant CCF 0916708.

contain a complete bipartite graph with parts of size at least $c_\epsilon \ln n$. The proofs of all these results are based on counting, and thus are *non-constructive*. Nevertheless, as a referee pointed out to us, the counting argument easily yields a randomized polynomial time algorithm that finds a copy of $K_{q,q}$. The algorithm is very simple: namely choose a random set of q vertices and check if they have a common neighborhood of size q. The counting argument shows that the expected number of common neighbors of a random q-set of vertices is at least q. Since the number of common neighbors is certainly at most n, we conclude that the probability that a random q-set has at least q neighbors is at least $1/n$ and therefore the algorithm succeeds in finding a $K_{q,q}$ with probability at least $3/4$ if we repeat this procedure $O(n)$ times.

We consider the question whether such subgraphs can be found by *deterministic* polynomial time algorithms. This question has been considered recently by Kirchner [9], who gave a deterministic polynomial time algorithm to find a complete balanced bipartite subgraph with parts of size $\Omega(\sqrt{\ln n})$ in graphs of constant density.

We improve this result by giving a deterministic polynomial time algorithm which finds a complete balanced bipartite subgraph with parts of size $\Omega(\ln n)$, i.e. of the optimal order of magnitude, in graphs of constant density. Our algorithm gives subgraphs of similar size as the counting argument in other ranges as well [1]. The algorithm works by restricting the search to a search space of polynomial size; the correctness proof uses the original counting argument. There is a trade-off between efficiency and the size of the subgraphs: the algorithm can be modified to run in *linear time* and produce subgraphs that are smaller by a constant factor. We emphasize that obtaining a deterministic algorithm from a randomized one in such settings is not necessarily an easy task, indeed, already the suboptimal result from [9] had a quite complicated proof.

Finding a largest balanced complete bipartite subgraph is an important optimization problem, which is known to be NP-hard, and even hard to approximate (see, e.g., Feige and Kogan [6]). We would like to emphasize that we are *not* trying to give an approximation algorithm for this problem. Our objective is to give an efficient algorithm which finds a balanced complete bipartite subgraph of size close to the largest size that is guaranteed to exist knowing *only* the number of edges in the graph. Thus, even in a dense graph, we are finding a subgraph of logarithmic size only. Results of this type could perhaps be referred to as algorithmic extremal graph theory, and are given, for example, in Alon *et*

[1] Note that the problem becomes meaningless in the sense studied here for fewer than $n^{3/2}$ edges, as such graphs do not always contain even $K_{2,2}$ subgraphs.

al. [1].

The counting argument of [10] has several applications to other combinatorial problems. It seems to be an interesting question whether the algorithmic version of the counting argument leads to further algorithmic results in these applications. As a case in point, we consider the question of decomposing, or partitioning, the edge set of a graph into complete bipartite graphs. The motivation to look for such algorithms comes from an application in approximation algorithms [2].

Every n-vertex graph can be decomposed into at most $n-1$ stars, and Graham and Pollak [7] showed that $n-1$ complete bipartite graphs are necessary for the n-vertex complete graph. Instead of minimizing the number of complete bipartite graphs in a decomposition, one can also try to minimize the complexity of decompositions, measured by *the sum of the number of vertices of the complete bipartite graphs used in the decomposition.* This measure of complexity was suggested by Tarján [12] in the context of circuit complexity. For recent connections to circuit complexity see Jukna [8].

It was shown by Chung, Erdős and Spencer [5], and by Bublitz [4], that there is always a decomposition of complexity $O(n^2/\ln n)$, and this order of magnitude is best possible. Similar results were obtained by Tuza [13] for decomposing bipartite graphs. These results are obtained by repeatedly applying the counting argument to show the existence of a large complete bipartite graph and removing its edges. Thus the decomposition results obtained in [4, 5, 13] are also non-constructive. As a direct application of our algorithm for finding bipartite subgraphs, we obtain efficient algorithms to find decompositions of complexity $O(n^2/\ln n)$.

2 Complete balanced bipartite subgraphs

Searching for a $K_{q,q}$ by checking all subgraphs of that size would give an algorithm with superpolynomial running time if q is, say, logarithmic in the number of vertices. A polynomial algorithm could be given by restricting the search space to a polynomial size set of candidate subgraphs. One possibility for that would be to find a bipartite subgraph (R, S) with the following properties:

- it is dense enough for the known results to guarantee the *existence* of a $K_{q,q}$, and
- the number of q-element subsets of R is only *polynomial.*

If such an (R, S) can be found efficiently then a required $K_{q,q}$ is obtained by checking the common neighborhood of all q-element subsets of R. It turns out that this approach indeed works if one chooses R to be the right number of vertices with maximal degree and S to be the remaining vertices. Thus, we consider the following procedure.

The inputs are a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, and parameters s and t.

```
Algorithm FIND-BIPARTITE $(G, s, t)$

if $0 < m \leq 8n^{3/2}$ then return any $(\{u\}, \{v\})$ with $(u, v) \in E$
else
    $R :=$ $s$ vertices having highest degree
    for all subsets $C \subseteq R$ with $|C| = t$ do
        $D := \bigcap\{N(v) - R \,:\, v \in C\}$
        if $|D| \geq t$ then $D' :=$ any set of $t$ elements of $D$,   return $(C, D')$
```

The algorithm can be implemented to run in time

$$O\left(m + \binom{s}{t} nt\right). \tag{1}$$

We assume that graphs are represented by adjacency lists. The claim about the running time is trivial if $m \leq 8n^{3/2}$. Otherwise, a size n array A containing the vertex degrees can be computed in time $O(m)$ by traversing the adjacency lists. The entries of A can be sorted in time $O(n \log n)$, which is $o(m)$. This provides the set R required by the algorithm. For the implementation of the **for** loop note that all t-subsets of R can be listed in $O(\binom{s}{t})$ steps (see, e.g. [11]). For a given t-subset T of R, the common neighbors of T outside of T can be found in $O(nt)$ steps, for example, by counting in a separate array B of size n the number of times each vertex of G occurs in the adjacency lists of vertices in T. This can be done by initializing B to 0, traversing the adjacency lists of vertices in T, and increasing the occurrence count in B each time a new edge is encountered.

Theorem 1 *Let*

$$q := \left\lfloor \frac{\ln(n/2)}{\ln(2en^2/m)} \right\rfloor, \quad r := \left\lfloor \frac{qn^2}{m} \right\rfloor.$$

If n is sufficiently large and $m \geq 8n^{3/2}$ then Algorithm **FIND-BIPARTITE** (G, r, q) *returns a $K_{q,q}$ with $q \geq 2$. The running time of the algorithm is polynomial in n.*

Remark. Note that our algorithm finds a $K_{q,q}$ in an n-vertex graph with $m = c_q n^{2-1/q}$ edges as long as c_q is large. This is optimal for $q = 2, 3$ as there exist n-vertex graphs with $c'_q n^{2-1/q}$ edges and no $K_{q,q}$, and if certain conjectures in extremal graph theory are true (see [3]), then it is also optimal for fixed $q > 3$.

Proof. After selecting $i < r$ vertices, the number of edges incident to these vertices is less than rn. Hence in the subgraph induced by the remaining vertices there is a vertex of degree at least $2(m - rn)/n$. Thus if R is the set of r highest degree vertices in G then

$$\sum_{v \in R} deg_G(v) \geq \frac{2r(m - rn)}{n}.$$

Hence the bipartite graph H with parts $R, V - R$ and edge set comprising those edges of G with one endpoint in R and the other in $V - R$ has at least $2rm/n - 3r^2$ edges.

We will now argue that $rm/n \geq 3r^2$. Indeed, $rm/n \geq 3r^2$ is equivalent to $r \leq m/3n$. Now $r \leq qn^2/m$ so it is enough to show that $qn^2/m \leq m/3n$ or equivalently, that $3qn^3 \leq m^2$. Using the definition of q, we see that $3qn^3 \leq m^2$ follows from

$$m^2 \ln(2en^2/m) \geq 3n^3 \ln(n/2).$$

Suppose first that $8n^{3/2} \leq m \leq 3n^{3/2}\sqrt{\ln n}$. Since n is sufficiently large

$$m^2 \ln\left(\frac{2en^2}{m}\right) \geq 64n^3 \ln\left(\frac{2en^2}{3n^{3/2}\sqrt{\ln n}}\right) > 64n^3 \ln\left(\sqrt{\frac{n}{\ln n}}\right) > 4n^3 \ln n > 3n^3 \ln(n/2).$$

On the other hand, if $m \geq 3n^{3/2}\sqrt{\ln n}$, then using $m < n^2/2$ we have

$$m^2 \ln(2en^2/m) \geq 9n^3 \ln n \ln(2en^2/m) > 9n^3 \ln n \ln(4e) > 3n^3 \ln(n/2).$$

We conclude that H has at least $2rm/n - 3r^2 \geq rm/n$ edges.

For the correctness of the algorithm it is sufficient to show that H contains a copy of $K_{q,q}$. This follows by the counting argument referred to in the introduction which we now describe in detail. Let b denote the number of stars with centers in $V-R$ and q leaves. Then

$$b = \sum_{v \in V-R} \binom{deg_H(v)}{q} \ \geq \ |V-R| \binom{\frac{\sum_{v \in H} deg_H(v)}{n}}{q} \ \geq \ \frac{n}{2} \binom{rm/n^2}{q}.$$

Explanation: the first inequality uses the convexity of the function which is $\binom{x}{q}$ if $x \geq q-1$ and 0 otherwise, and the second inequality uses the lower bound for the number of edges in H, and the inequality $r \leq n/2$ which follows by the lower bound on m.

If the latter quantity is greater than $(q-1)\binom{r}{q}$ then there is a q-subset of R which is the leaf set for at least q distinct stars, and this gives a copy of $K_{q,q}$. Observe that the definition of q implies that $n/2 \geq (2en^2/m)^q$ and this is equivalent to

$$\frac{n}{2} \left(\frac{rm}{n^2 q} \right)^q \geq \left(\frac{2er}{q} \right)^q .$$

Now the inequality above and standard estimates of the binomial coefficients give

$$\frac{n}{2} \binom{rm/n^2}{q} > \frac{n}{2} \left(\frac{rm}{n^2 q} \right)^q \geq \left(\frac{2er}{q} \right)^q \geq q \left(\frac{re}{q} \right)^q > (q-1) \binom{r}{q},$$

Thus H indeed contains a $K_{q,q}$.

In order to estimate the running time bound given in (1) note that

$$\binom{r}{q} \leq \left(\frac{re}{q} \right)^q \leq e^q \left(\frac{n^2}{m} \right)^q = e^q e^{q \ln(n^2/m)}.$$

Now $m < n^2/2$ implies that

$$e^q \leq e^{\ln n / \ln 4e} = n^{1/\ln 4e} < n^{0.4195}, \tag{2}$$

and $q < \ln n / \ln(n^2/m)$ implies that

$$e^{q \ln(n^2/m)} < e^{\ln n} = n. \tag{3}$$

Combining these bounds with the other terms in (1) it follows that the running time of the algorithm is $O(n^{2.42})$. □

As noted above, the size of the bipartite graphs found in Theorem 1 is optimal in a certain range of values of m if certain conjectures in extremal graph theory hold. We now show that bipartite graphs that are smaller by a constant factor can be found in linear time.

Theorem 2 *Let q, r be as in Theorem 1 and $q' = \lfloor q/5 \rfloor$. If n is sufficiently large and $m \geq 8n^{3/2}$ then Algorithm* **FIND-BIPARTITE** (G, r, q') *returns a $K_{q',q'}$. The running time of the algorithm is $O(m)$.*

Proof. As r is the same as in Theorem 1 and $q' < q$, the proof of Theorem 1 implies that the algorithm finds a copy of $K_{q',q'}$. Thus it is sufficient to show that the running time bound (1) becomes $O(m)$. Let us repeat the computation for bounding $\binom{r}{q}$, with q replaced by q'. Using $q/5 < r/2$, we have

$$\binom{r}{q'} \leq \binom{r}{q/5} \leq \left(\frac{5re}{q}\right)^{\frac{q}{5}} \leq 5^{\frac{q}{5}} e^{\frac{q}{5}} \left(\frac{n^2}{m}\right)^{\frac{q}{5}} = 5^{\frac{q}{5}} e^{\frac{q}{5}} e^{\frac{q}{5}\ln(n^2/m)} = 5^{\frac{q}{5}} \left(e^q e^{q\ln(n^2/m)}\right)^{\frac{1}{5}}.$$

Now by (2)

$$5^{\frac{q}{5}} = e^{\frac{q}{5}\ln 5} = (e^q)^{\frac{\ln 5}{5}} \leq (n^{0.42})^{\frac{\ln 5}{5}},$$

and applying (3) we obtain

$$\binom{r}{q'} = O\left(n^{\frac{0.42(1+\ln 5)+1}{5}}\right) = O\left(n^{0.4192}\right).$$

So the last term in (1) is $o(n^{3/2}) = o(m)$, as $m \geq 3n^{3/2}$. Thus, apart from finding the r largest degree vertices in the beginning, the running time of the algorithm is actually sublinear in m. □

3 Decomposition into balanced complete bipartite subgraphs

Given a graph $G = (V, E)$, we consider complete bipartite subgraphs $G_i = (A_i, B_i, E_i), i = 1, \ldots, t$ such that the edges sets E_i form a partition of E. The *complexity* of such a decomposition is measured by the total number of vertices, i.e., by

$$\sum_{i=1}^{t} |A_i| + |B_i|.$$

We find a decomposition of complexity $O(n^2/\ln n)$. The decomposition contains *balanced* bipartite graphs, thus $|A_i| = |B_i|$ holds as well. The algorithm uses Algorithm **FIND-BIPARTITE** in a straightforward manner. As stated, Algorithm **FIND-BIPARTITE** is guaranteed to work only if $n \geq n_0$ for some n_0. As we are only interested in proving an asymptotic result, let us assume that graphs on fewer vertices are handled by some brute-force method.

Algorithm **FIND-DECOMPOSITION** (G)

> Given an n-vertex input graph $G = (V, E)$, if $n < n_0$, use a brute-force method to find an optimal decomposition of G. Else, use Algorithm **FIND-BIPARTITE** (with parameters r and q as in Theorem 1) repeatedly to find a complete balanced bipartite subgraph and delete it from the current graph, as long as there are more than $n^2/\ln n$ edges. After that, form a separate bipartite graph from each remaining edge.

Theorem 3 *For every n-vertex graph G, Algorithm* **FIND-DECOMPOSITION** (G) *finds a decomposition of G into balanced complete bipartite graphs, having complexity*

$$O\left(\frac{n^2}{\ln n}\right).$$

The running time of the algorithm is polynomial in n.

Proof. As the size of the subgraphs produced by Algorithm **FIND-BIPARTITE** is of the same order of magnitude as guaranteed by the existence theorems, the theorem follows as in [4, 5, 13]. For completeness, we give the argument, following [13].

Let the subgraphs produced by the calls of Algorithm **FIND-BIPARTITE** be $G_i = (A_i, B_i)$ with $|A_i| = |B_i| = q_i$, where $i = 1, \ldots, t$ for some t . We need to show that

$$\sum_i q_i = O\left(\frac{n^2}{\ln n}\right). \tag{4}$$

Let us divide the iterations of the algorithm into *phases*. The ℓ'th phase consists of those iterations where the number of edges in the input graph of Algorithm **FIND-BIPARTITE** is more than $n^2/(\ell + 1)$ and at most n^2/ℓ. Dividing up the term q_i in (4) between the q_i^2 edges of G_i, each edge gets a weight of $1/q_i$. We have to upper bound the sum of the weights assigned to the edges.

It follows from the definition of q_i in Theorem 1 that graphs formed in the ℓ'th phase have $q_i = \Theta(\ln n / \ln \ell)$. Thus edges, which get their weight in the ℓ'th phase, get a weight of $\Theta(\ln \ell / \ln n)$. The number of edges getting their weight in the ℓ'th phase is $\Theta((\frac{1}{\ell} - \frac{1}{\ell+1})n^2) = \Theta(n^2/\ell^2)$. Hence the total weight assigned to the edges is at most of the order of magnitude

$$\sum_{\ell=1}^{\infty} \frac{\ln \ell}{\ln n} \cdot \frac{n^2}{\ell^2} = \Theta\left(\frac{n^2}{\ln n}\right),$$

as $\sum \frac{\ln \ell}{\ell^2}$ is convergent. The polynomiality of the running time follows directly from the polynomial running time of Algorithm **FIND-BIPARTITE**. □

Acknowledgment We thank Stefan Kirchner for sending us his Ph.D. dissertation and the referees for very helpful comments that improved the presentation.

References

[1] N. Alon, R. A. Duke, H. Lefmann, V. Rödl, R. Yuster: The algorithmic aspects of the regularity lemma, *J. of Algorithms* **16** (1994), 80-109.

[2] A. Bhattacharya, B. DasGupta, D. Mubayi, Gy. Turán: On approximate Horn minimization, In preparation.

[3] B. Bollobás: *Extremal Graph Theory.* Academic Press, 1978.

[4] S. Bublitz: Decomposition of graphs and monotone formula size of homogeneous functions, *Acta Informatica* **23** (1986), 689-696.

[5] F. R. K. Chung, P. Erdős, J. Spencer: On the decomposition of graphs into complete bipartite graphs, in: *Studies in Pure Mathematics, To the Memory of Paul Turán*, 95-101. Akadémiai Kiadó, 1983.

[6] U. Feige, S. Kogan: Hardness of approximation of the balanced complete bipartite subgraph problem, *Tech. Rep. MCS04-04, Dept. of Comp. Sci. and Appl. Math., The Weizmann Inst. of Science*, 2004.

[7] R. L. Graham, H. O. Pollak: On the addressing problem for loop switching, *Bell Syst. Techn. J.* **50** (1971), 2495-2519.

[8] S. Jukna: Disproving the single level conjecture, *SIAM J. Comp.* **36** (2006), 83-98.

[9] S. Kirchner: Lower bounds for Steiner tree algorithms and the construction of bicliques in dense graphs. Ph.D. Dissertation, Humboldt-Universität zu Berlin, 2008. (In German.)

[10] T. Kővári, V. T. Sós, P. Turán: On a problem of K. Zarankiewicz, *Colloq. Math.* **3** (1954), 50-57.

[11] E. M. Reingold, J. Nievergelt, N. Deo: *Combinatorial Algorithms.* Prentice Hall, 1977.

[12] T. Tarján: Complexity of lattice-configurations, *Studia Sci. Math. Hung.* **10** (1975), 203-211.

[13] Zs. Tuza: Covering of graphs by complete bipartite subgraphs; complexity of 0-1 matrices, *Combinatorica* **4** (1984), 111-116.