Report:

Application of A Planar Separator Theorem

Student: Li Yan
School of Computer Science
Carleton University

Abstract. Through the paper *Application of A Planar Separator Theorem* by RICHARD J. LIPTON and ROBERT TARJAN, we get that any n-vertex planar graph can be divided into components of roughly equal size by removing only $O(\sqrt{n})$ vertices. This separator theorem with a divide-and-conquer strategy can help us resolve many new complexity planar graph problems. This report briefly describes six application of this theorem in planar graph case

.

Introduction. "Divide-and-conquer" is a solution that divide original problem into two or more smaller problems, and then apply for the method recursively in order to get solution of original problem. This is an efficient way to resolve problem. Let's see the planar separator theorem firstly, which is called theorem 1 here.

Theorem 1 [1]. *Let G be any n-vertex planar graph with nonnegative vertex costs summing to no more than one. Then the vertices of G can be partitioned into three sets A, B, C, such that no edge joins a vertex in A with a vertex in B, neither A nor B has total vertex cost exceeding 2/3, and C contains no more than* $2\sqrt{2}\sqrt{n}$ *vertices. Furthermore A, B, C can be found in O(n) time.*

We can get corollary 1 of the above theorem in the special case of equal-cost vertices.

Corollary 1. *Let G be any n-vertex planar graph. The vertices of G can be partitioned into three sets A, B, C, such that no edge joins a vertex in A with a vertex in B, neither A nor B contains more than 2n/3 vertices, and C contains no more than* $2\sqrt{2}\sqrt{n}$ *vertices.*

The following sections will describe six different applications of theorem 1 in planar graph by removing a small number of vertices on the planar graph. The solution to resolve NP-complete problem using theorem 1 will be given detailedly in this report.

1. Approximation algorithms for NP-complete problems.

Using Theorem 1 combined with divide-and–conquer to find a good approximate solution to NP-complete problems on planar graph. Maximum independent set problems on planar graph will be resolved by theorem 1, as an example.

Theorem 3. *Let G be an n-vertex planar graph with nonnegative vertex costs summing to no more than one and let* $0 \le \varepsilon \le 1$. *Then there is some set C of* $O(\sqrt{n/\varepsilon})$ *vertices whose removal leaves G with no connected component of cost exceeding* ε. *Furthermore the set C can be found in O(n log n) time.*

Proof as following:

If $\varepsilon \le 1/\sqrt{n}$, let C contain all the vertices of G. Then we can get it. Otherwise, use the following algorithm:

1) Define method level (x): any two components on the same level are vertex-disjoint.
2) Initialization. Let $C = \emptyset$.
3) If cost (G – C) $\leqslant \varepsilon$
   Return level(i) = 0

4) While exists connected component K= {G-C} and cost (level(i)_k) > ε Do

Apply Theorem 1 split K

Cost (level(i+1)_A1) ⩽ 2/3 cost (level(i)_K)

Cost (level(i+1)_B1) ⩽ 2/3 cost (level(i)_K)

$N(C1) \leqslant 2\sqrt{2}\, n_k$

C = C ∪ { C1}

i = i + 1

Iterate do step 4 until find the maximum independent set C.

Analyze:

Assume at level i , in which we got this kind of set C, then cost(G-C) ⩽ ε

i = 1 Cost(level(1)) ⩾ ε

i = 2 Cost(level(2)) ⩾ 3/2 ε

i = 3 Cost(level(3)) ⩾$(3/2)^2$ ε

… …

Thus, at each level i, the cost of this level satisfies

Cost (level(i)) ⩾ $(3/2)^{i-1}$ ε

Since,the total cost ⩽ 1

Vertices n at each level i satisfy the following inequation:

$\mathbf{N_i} \leqslant 1/(3/2)^{i-1}\,\varepsilon = (2/3)^{i-1}/\varepsilon$

With condition 0 ⩽ ε ⩽ 1, induce the next inequation:

$1 \leqslant (2/3)^{k-1}/\varepsilon \leqslant (2/3)^{k-1}\sqrt{n}$

Maximum level k satisfies:

$K \leqslant (\log_{3/2} n)/2 + 1$

Because the time to split a component is linear in its number of vertices, and any two components on the same level are vertex-disjoint, then we get the total running time of the algorithm is *O(n log n)*.

How to calculate the size of set *C* produced by the algorithm?

Let $K_1, K_2, \ldots, K_l$, of sizes $n_1, n_2, \ldots, n_l$, respectively at some level $i \geq 1$.

The number of vertices added to C is

$2\sqrt{2}\sum_{j=1}^{l}\sqrt{n_j}$

We already got from above proof:

$l \leqslant (2/3)^{i-1}/\varepsilon$

$\sum_{j=1}^{l} n_j \leq n$

$\sum_{j=1}^{l}\sqrt{n_j}$

is maximized by setting $n_j = n/l$ ($1 \leq j \leq l$);

Thus, $2\sqrt{2}\sum_{j=1}^{l}\sqrt{n_j} \leq 2\sqrt{2}\sqrt{nl} \leq 2\sqrt{2}\sqrt{n/\varepsilon(2/3)^{(i-1)/2}}$

We can get $|C| \leq \sum_{i=1}^{\infty} 2\sqrt{2}\sqrt{n/\varepsilon(2/3)^{(i-1)/2}} = O(\sqrt{n/\varepsilon})$.

The following algorithm is used to find an approximately maximum independent set *I* in a planar graph *G = (V, E).*

Step 1, Let $\varepsilon = k(n)/n$, and each vertex v_i *with* cost$(v_i) = 1/n$

Apply theorem 3 to G,

we can get this kind of set C with size $O(\sqrt{n/\varepsilon})$

And cost(G-C) ⩽ ε

Since each vertex of cost$(v_i) = 1/n$

Thus, we can get the total vertices of G-C ⩽ ε / (1/n) = (k(n)/n)/(1/n)

Total vertices of G-C ⩽ k(n)

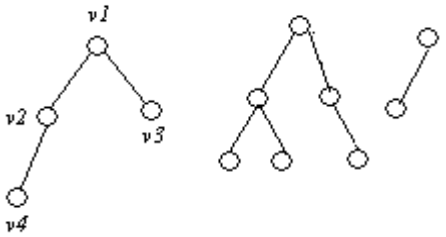
Time: *O(n log n)*

Step 2,

- check each subset of vertices for independence in each connected component of G-C.
- Form a set *I* as union of maximum independent sets, one by one from each components

For instance, we get G-C as following,

For the first connected component, we can get it's subset of 2^4 like that



$\{Ø\}$, $\{v_1\}$, $\{v_2\}$, $\{v_3\}$, $\{v_4\}$,
$\{v_1, v_2\}$, $\{v_1, v_3\}$, $\{v_1, v_4\}$, $\{v_2, v_3\}$,
$\{v_2, v_4\}$, $\{v_3, v_4\}$, $\{v_1, v_2, v_3\}$,
$\{v_1, v_2, v_4\}$, $\{v_2, v_3, v_4\}$, $\{v_1, v_3, v_4\}$,
$\{v_1, v_2, v_3, v_4\}$

Check each subset to see whether there exists independent set, if exists, then add it into maximum set I. So we can get the time for check each component with n_i vertices is

$O(n_i\, 2^{n_j})$.

The total time required in step 2 is :

$$O(\max\{\sum_{i=1}^{n} n_i 2^{n_j} \mid \sum_{i=1}^{n} n_i = n \text{ and } O \le n_i \le k(n)\}) = O(\frac{n}{k(n)}\, k(n)\, 2^{k(n)}) = O(n\, 2^{k(n)})$$

Hence the entire algorithm requires $O(n \bullet \max\{\log n, 2^{k(n)}\})$ time.

Analyze the relative error:

Assume, I^* be a maximum independent set of G. And restriction of I^* to one of the connected components be no larger than the restriction of I to the same component when C is removed from G. Then $|I^*| - |I| = O(n / \sqrt{k(n)})$.

Since G is planar, G is four-colorable, and $|I^*| \ge n/4$.

So $(|I^*| - |I|) / |I^*| = O(1 / \sqrt{k(n)})$, when n is increasing the relative error in the size of I tends to zero.

If we let $k(n) = \log n$, then can get an $O(n^2)$ time algorithm with $O(1 / \sqrt{\log n})$ relative error.

If we let $k(n) = \log\log n$, then we can get an $O(n \log n)$ algorithm with $O(1 / \sqrt{\log\log n})$ relative error.

2. Nonserial dynamic programming.

We can define many *NP*-complete problems, such as the maximum independent set problem, the graph coloring problem, and so on, as nonserial dynamic programming.[2] An additional concept need to descripted, the restriction of an objective function $f = \sum_{k=1}^{m} f_k$ to a set of variables $x_{i_1}, \ldots, x_{i_j}$ is the objective function $f' = \sum \{f_k \mid f_k$ depends only upon $x_{i_1}, \ldots, x_{i_j}\}$. We will use algorithm to solve the problem: maximize f subject to the constraints on the variables in S.

Step 1. If $n < 100$ we can treat it by assignments to the unconstrained variables. Otherwise, see the step 2.

Step 2. We can use Corollary 1 to the interaction graph G of f. Partition G into three sets A, B, C, and let f_1 be the restriction of f to $A \cup C$ and let f2 be the restriction of f to

$B \cup C$. For each assignment of values to the variables in $C - S$, perform the following steps:

(a) Maximize f1 with values for variables in $S \cup C$ by applying the method recursively;

(b) Maximize f2 with values for variables in $S \cup C$ by applying the method recursively;

(c) Plus (a) and (b) to obtain a maximum value of f with the given value for variables in $S \cup C$.

Through this algorithm we can get that if n ≥ 100, there is at most $2^{O(\sqrt{n})}$ subproblems in Step 2, since C is of $O(\sqrt{n})$ size. And each variables have at most 2n/3 + $2\sqrt{2}\sqrt{n}$ ≤ 29n/30 variables. So we have the running time of the algorithm $t(n) \le O(n) + 2^{O(\sqrt{n})} \bullet t(29n/30)$; if $n < 100$, and $t(n) = O(1)$. In brifly, the result is $t(n) \le 2^{O(\sqrt{n})}$

3. Pebbling. Let $G = (V, E)$ be a directed acyclic graph with maximum in-degree k. And define (u, w) is an edge of G, u is a predecessor of w. We give a restriction that a vertex may be pebbled only if all its predecessors have pebbles. This satisfies the topological order that an order such that if v is a predecessor of w, v is pebbled before w.

Theorem 4. *Any n-vertex planar acyclic directed graph with maximum in-degree k can be pebbled using* $O(\sqrt{n} + k \log_2 n)$ *pebbles*.

Proof. Assume there is a graph $G = (V, E)$, $a = 2\sqrt{2}$ and $\beta = 2/3$. If $n = 1$, pebble the single vertex of G. If $n > 1$, find a vertex partition A, B, C satisfying Corollary 1. And pebble the vertices of G in topological order. After pebble a vertex v, delete all pebbles except those on C. Using this method recursively. If $p(n)$ is the maximum number of pebbles required by this method on any n-vertex graph, then we can get:

$p(1) = 1$,

$p(n) \le a\sqrt{n} + k + p(\ 2n/3\ )$ if $n > 1$

It's easy to get that $p(n)$ is $O(\sqrt{n} + k \log_2 n)$.

We can reduce the number of pebbling steps by a polynomial bound as following theorem shows.

Theorem 5. *Any n-vertex planar acyclic directed graph with maximum in-degree k can be pebbled using* $O(n^{2/3} + k)$ *pebbles in* $O(n^{5/3})$ *time.*

Proof. We can define C be a set of $O(n^{2/3})$ vertices whose removal leaves G with no weakly connected component containing no more than $n^{2/3}$ vertices. There exists such a set C according to Theorem 2. Recursively perform pebbling a predecessor u by deleting all pebbles from G except those on vertices in C or on predecessors of u, and find the weakly connected component in G minus C containing u. We know that the number of pebbles in pebble vertices in C is $O(n^{2/3})$, and of pebble each weakly connected component is $n^{2/3}$, and of pebble predecessors of vertex v is k. Bound the number of pebbling steps, we can get the total pebbling time is $n + \sum_{v \in V} d_1(v)\, n^{2/3} \le n + (3n - 3)\, n^{2/3} = O(n^{5/3})$

4. Lower bounds on Boolean circuit size.

Here the size the means the number of vertices. A Boolean circuit is an acyclic directed graph in which each vertex has in-degree zero or two, and the predecessors of each vertex are ordered, and corresponding to each vertex v of in-degree two is a binary Boolean operation b_v. There is a Boolean function which is associcated with each vertex of the circuit, which the vertex computes. This function is defined like this, with each k vertices v_i of in-degree zero (inputs), we associate a variable x_i and an function $f_{v_i}(x_i) = x_i$. With each vertex w in degree two having predecessors u, v, and

we associate the function $f_w = b_w (f_u, f_v)$. The circuit computes the set of functions which associate with its vertices of out-degree zero(outputs). We will use algorithm to get lower bounds on the size of this Boolean circuits which compute certain functions. If we assume that the circuits are planar, it's easy to obtain such lower bounds using Theorem 1. We can find a way to convert circuit graph into a planar circuit as following description. The first step is that we can embed the circuit in a plane, and some edges may cross. And then we can replace each pair of crossing edges by the crossover circuit which is illustrated in following figure. As we know that any lower bound on the size of planar circuits is also a lower bound on the total number of vertices and edge crossings in any planar which represent a corresponding nonplanar circuit. But this lower bounds imply that it may be expensive to realize certain commonly used functions in hardware in a technology.

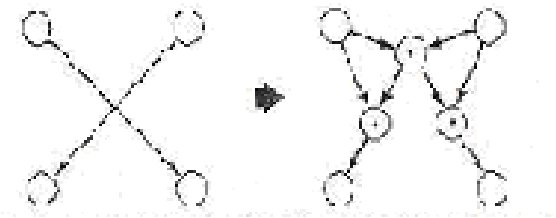


Figure Elimination of a crossover by use of three "exclusive or" gate. Reference [11]

Based on Theorem 1 we can get next Theorem.

Theorem 6. *Any m-input, m-output planar superconcentrator contains at least $m^2/72$ vertices.*

Proof. Give a definition that a superconcentrator is an acyclic directed graph with m inputs and m outputs, and any set of k inputs and any set of k outputs are joined by k vertex-disjoint paths, for all k in the range $1 \le k \le m$. Let G be an m-input, and m-output planar superconcentrator. We assign each input and output G a cost of $1/(2m)$, and every other vertex a cost zero. Applying Theorem 1, we can get three partitions *A, B, C* on *G.* (ignoring edge directions). Suppose C have p inputs and outputs, then A contains at least $m/3 - p/2$ inputs and at most $m - p/2$ inputs and outputs; and B contains at least $m - p - (m/2 - p/4) = m/2 - 3p/4$ outputs. Define $k = \min\{ m/3 - p/2 , \lceil m/2 - 3p/4 \} $. Since *G* is a superconcentrator, any set of k inputs in *A* and any set of *k* outputs in *B* are joined by *k* vertex-disjoint paths. And each path must contain a vertex in C which is neither an input nor an output. Thus $2\sqrt{2}\sqrt{n} - p \ge \min\{m/3 - p/2, m/2 - 3p/4\}$, and $n \ge m^2/72$.

Because the property of being a superconcentrator is too strong to be useful in deriving lower bounds on the complexity of functions. We use shifting property of G, which is that in $G = (V, E)$, there exists any k in the range $1 \le k \le m$, any l in the range $0 \le l \le m - k$, and any subset of k sources $\{v_{i_1}, ..., v_{i_k}\}$, in which $i_1, i_2, ..., i_k \le m - l$, there are k vertex-disjoint paths joining the set of inputs $\{v_{i_1}, ..., v_{i_k}\}$ with the set of outputs $\{w_{i_1+l}, ..., w_{i_k+l}\}$. So we lead to Theorem 7.

Theorem 7. *Let G be a planar acyclic directed graph with the shifting property. Then C contains at lease $m/2^2/162$ vertices.*

There are several Corollary are induced here, such that

Corollary 2. *Any planar shifting circuit has at least $\lfloor m/2^2/162$ vertices.*

Corollary 3. *Any planar circuit for computing Boolean convolution has at least* $\lfloor m/2^{2} / 162$ *vertices.*

Corollary 4. *any planar circuit for computing the product of two m bit binary integers has at least* $\lfloor m/2^{2} / 162$ *vertices*.

And one Theorem 8 . *Any planar circuit G for multiplying two* $m \times m$ *Boolean matrices contains at least* cm^4 *vertices, for some positive constant c*.

The proof of these are shown on [3][4]

5. **Embedding of data structures**

References

[1] R.J.LIPTON AND R.E. TARJAN, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp.177-189

[2] U. BERTELE AND F. BRIOSCHI, *Nonserial Dynamic Programming,* Academic Press, New York, 1972

[3] L.G.VALIANT, *On non-linear lower bounds in computational complexity*, Proc. Seventh Annual ACM Symp. on Theory of Computing (1975), pp.45-53

[4] L.G.VALIANT, *Graph-theoretic arguments in low-level complexity,* Computer Science Dept., University of Edinburgh, 1977.