Finding Large Subgraphs

Hemanshu Kaul

Illinois Institute of Technology

Graphs and Subgraphs

Graphs model binary relationships.

In a graph $G = (V(G), E(G))$, the objects under study are represented by vertices included in V(G).

If two objects are "related" then their corresponding vertices, say u and v in $V(G)$, are joined by an edge that is represented as uv (technically, $\{u, v\}$) in $E(G)$.

H is a subgraph of G if $V(H) \subseteq V(G)$ and every edge of H belongs to $E(G)$.

From G, remove some edges and some vertices if needed.

H is an induced subgraph of G if $V(H) \subseteq V(G)$ and it has all the edges that appear in G over the same vertex set.

From G, remove some vertices only.

Graphs and Subgraphs

Graphs model binary relationships.

In a graph $G = (V(G), E(G))$, the objects under study are represented by vertices included in V(G).

If two objects are "related" then their corresponding vertices, say u and v in $V(G)$, are joined by an edge that is represented as uv (technically, $\{u, v\}$) in $E(G)$.

H is a subgraph of G if $V(H) \subseteq V(G)$ and every edge of H belongs to $E(G)$.

From G, remove some edges and some vertices if needed.

H is an induced subgraph of G if $V(H) \subseteq V(G)$ and it has all the edges that appear in G over the same vertex set.

From G, remove some vertices only.

Examples of Graph Families

K_n : Complete graph on n vertices. Each of the $\binom{n}{2}$ pairs of vertices is joined by an edge.

Independent set : No edges.

P_n : Path on n vertices.

C_n : Cycle on n vertices.

Forest : Graph with no cycles.

$K_{m,n}$: Complete bipartite graph. $V(G) = A \cup B$ where $|A| = m$ and $|B| = n$ and $E(G)$ has all the mn edges of type ab, $a \in A$ and $b \in B$.

Bipartite graph : A subgraph of $K_{m,n}$. No odd cycles.

Planar graph : A graph that can drawn on the plane with non-crossing edges.

The maximum subgraph problem

The maximum subgraph problem for a Graph property Π asks:

Given a graph G, find a subgraph H of G satisfying property Π that has the maximum number of edges.

The maximum induced subgraph problem

The maximum induced subgraph problem for a Graph property Π asks:

Given a graph G, find an induced subgraph H of G satisfying property Π that has the maximum number of vertices.

In other words, find the minimum number of vertices to remove from G such that the remaining subgraph satisfies the property Π.

Graph Properties

The following Graph properties are commonly considered:

- Forest (cycles are forbidden)
- Bipartite (odd cycles are forbidden)
- Planarity ($\{K_5, K_{3,3}\}$-minors are forbidden)
- Complete

  There is no difference between induced and non-induced versions for this.
- Independent set

  This is meaningful only for the induced version.

All these properties are hereditary, every subgraph of a graph with property Π also has property Π.

Connectedness is an example of a property that is not hereditary.

Graph Properties

The following Graph properties are commonly considered:

- Forest (cycles are forbidden)
- Bipartite (odd cycles are forbidden)
- Planarity ($\{K_5, K_{3,3}\}$-minors are forbidden)
- Complete

  There is no difference between induced and non-induced versions for this.
- Independent set

  This is meaningful only for the induced version.

All these properties are hereditary, every subgraph of a graph with property Π also has property Π.

Connectedness is an example of a property that is not hereditary.

Finding Large Subgraphs

Except for the largest Forest subgraph problem,
all these largest subgraph problems are NP-hard.

In case of the largest induced subgraph problem,
Lewis and Yannakakis (1980) showed that:

The largest induced subgraph problem is NP-hard for every non-trivial hereditary property.

What about approximate solutions?

Approximation Algorithms

Algorithm $\mathcal{A}$ for a maximization problem *MAX* achieves an approximation factor α if

for all inputs G, we have: $\frac{\mathcal{A}(G)}{OPT(G)} \geq \alpha$,
where $\mathcal{A}(G)$ is the value of the output generated by the algorithm $\mathcal{A}$,
and $OPT(G)$ is the optimal value.

A α-approximation algorithm for *MAX* is a polynomial-time algorithm that achieves the approximation factor α.

Approximation Algorithms for Large Subgraphs

For example
for the largest clique subgraph problem:
Feige (2005): $O(n(loglogn)^2/(logn)^3)$-approximation algorithm

Feige et al. (1996): It is hard to approximate MAX-Clique for any constant factor.

Hastad (1999): It is hard to approximate MAX Clique within a factor $O(1/n^{\epsilon})$ for any $\epsilon > 0$

for the largest bipartite subgraph problem:

Goemans and Williamson (1995): 0.878-approximation algorithm

Hastad (1997): If $P \neq NP$ then there is no α-approximation algorithm for any $\alpha > 0.941$.

Approximation Algorithms for Large Subgraphs

For example
for the largest clique subgraph problem:
Feige (2005): $O(n(loglogn)^2/(logn)^3)$-approximation algorithm

Feige et al. (1996): It is hard to approximate MAX-Clique for any constant factor.

Hastad (1999): It is hard to approximate MAX Clique within a factor $O(1/n^{\epsilon})$ for any $\epsilon > 0$

for the largest bipartite subgraph problem:

Goemans and Williamson (1995): 0.878-approximation algorithm

Hastad (1997): If $P \neq NP$ then there is no α-approximation algorithm for any $\alpha > 0.941$.

Finding a Large Bipartite Subgraph

In a given graph G, find a bipartition (cut) (X, Y), with $X \subseteq V(G)$ and $Y = V(G) \setminus X$, that maximizes the number of edges between X and Y.

$b(G)$ be the number of edges in a largest bipartite subgraph of G.

- Extremal results like Edwards-Erdős Inequalities :
  1) $b(G) \geq \frac{1}{2}m + \frac{1}{8}(\sqrt{8m+1} - 1)$, $m = |E(G)|$
  2) $b(G) \geq \frac{1}{2}m + \frac{1}{4}(n - 1)$, $n = |V(G)|$

A local search algorithm

Idea : Starting with an arbitrary vertex partition, switch a vertex from one partite set to the other if doing so increases the number of edges in the cut (the bipartite subgraph induced by the vertex partition).

Given a partition $V(G) = X \cup Y$ of the vertex set of a graph G, a local switch moves a vertex v from X to Y that has more neighbors in X than in Y.

A list of local switches performed successively is a switching sequence.

Size of the bipartite subgraph : How big a bipartite subgraph is guaranteed at the end of a switching sequence?

Length of a switching sequence : How long can a switching sequence be?

A local search algorithm

Idea : Starting with an arbitrary vertex partition, switch a vertex from one partite set to the other if doing so increases the number of edges in the cut (the bipartite subgraph induced by the vertex partition).

Given a partition $V(G) = X \cup Y$ of the vertex set of a graph G, a local switch moves a vertex v from X to Y that has more neighbors in X than in Y.

A list of local switches performed successively is a switching sequence.

Size of the bipartite subgraph : How big a bipartite subgraph is guaranteed at the end of a switching sequence?

Length of a switching sequence : How long can a switching sequence be?

A local search algorithm

Idea : Starting with an arbitrary vertex partition, switch a vertex from one partite set to the other if doing so increases the number of edges in the cut (the bipartite subgraph induced by the vertex partition).

Given a partition $V(G) = X \cup Y$ of the vertex set of a graph G, a local switch moves a vertex v from X to Y that has more neighbors in X than in Y.

A list of local switches performed successively is a switching sequence.

Size of the bipartite subgraph : How big a bipartite subgraph is guaranteed at the end of a switching sequence?

Length of a switching sequence : How long can a switching sequence be?

Size of the Bipartite Subgraph

Erdős: Each local switch increases the number of edges in the cut, so the algorithm has to stop. When the algorithm stops, at least half the edges incident to each vertex are in the cut, so the final bipartite subgraph contains at least half the edges of G.

Bylka + Idzik + Tuza, 1999: A bipartite subgraph of size $\frac{1}{2}m + \frac{1}{4}o(G)$ is guaranteed, where $o(G)$ is the number of odd degree vertices in G.

A slight modification of the local switching rules improves the guarantee to the first Edwards-Erdős Inequality : $b(G) \geq \frac{1}{2}m + \frac{1}{8}(\sqrt{8m+1} - 1)$.

Minimum length of a switching length

Let $s(G)$ denote the minimum length of a maximal flip sequence starting from the trivial vertex partition.

Theorem [Kaul + West, 2008]: If G is an n-vertex loopless multigraph, then $s(G) \leq n/2$.
In fact, there exists a sequence of at most $n/2$ flips that produces a globally optimal partition.

Maximum length of a switching length

Observation: The maximum length of a switching sequence, $l(G)$, is at most $b(G) \leq e(G)$.

This is best possible, as the star $K_{1,n-1}$ achieves equality for both $b(G)$ and $e(G)$.

Theorem [Kaul + West, 2008]: The length of any switching sequence is at most $b(G) - (\frac{3}{8}\delta^2(G) + \delta(G))$.

Maximum length of a switching length

Bounding the length with n : A bipartite graph on n vertices has at most $\frac{n^2}{4}$ edges, so any switching sequence has length at most $\frac{n^2}{4}$.

Can we do faster than $\frac{1}{4}n^2$ switches to reach a local optima?

Cowen + West, 2002: When n is a perfect square, there exists a graph G with n vertices that has a switching sequence of length $e(G) = \frac{1}{2}n^{\frac{3}{2}}$.

This gave hope that $l(G) \leq O(n^{\frac{3}{2}})$.

Maximum length of a switching length

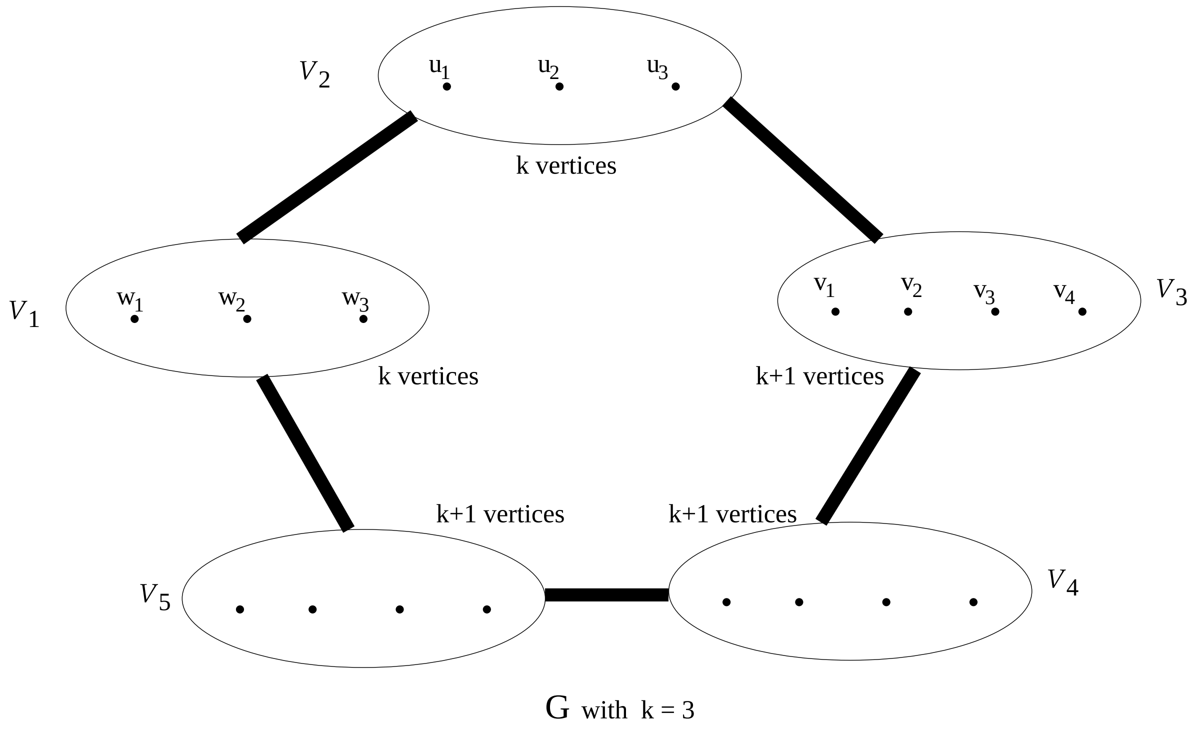
Bounding the length with n : A bipartite graph on n vertices has at most $\frac{n^2}{4}$ edges, so any switching sequence has length at most $\frac{n^2}{4}$.
Can we do faster than $\frac{1}{4}n^2$ switches to reach a local optima?

Cowen + West, 2002: When n is a perfect square, there exists a graph G with n vertices that has a switching sequence of length $e(G) = \frac{1}{2}n^{\frac{3}{2}}$.

This gave hope that $l(G) \leq O(n^{\frac{3}{2}})$.

Maximum length of a switching length

Theorem [Kaul + West, 2008]: For every n, there exists a graph G with n vertices that has a switching sequence of length at least $\frac{2}{25}(n^2 + n - 31)$.



Open Questions I

Problem 1. Determine the exact constant multiple (between $\frac{8}{100}$ and $\frac{25}{100}$) of n^2 for $I(G)$.

Problem 2. New ideas for upper bounds on $I(G)$.

Open Questions I

Modify the switching algorithm by allowing up to $k \geq 1$ vertices to be switched at a time.

How close can we get to the second Edwards-Erdős Inequality : $b(G) \geq \frac{1}{2}m + \frac{1}{4}(n-1)$?

Problem 3. [Tuza, 2001] Given k, determine the largest constant $c = c(k)$ such that the local switching algorithm guarantees a bipartite subgraph of size at least $\frac{1}{2}m + cn - o(n)$.

A construction shows that $c(k) < \frac{1}{4}$, for all k.

What is the smallest k with $c(k) > 0$? Is $c(1) > 0$?

Approximation Algorithms for Large Induced Subgraphs

Lund and Yannakakis (1993): It is hard to approximate the largest induced subgraph problem for any non-trivial hereditary property.

Comparatively, very little research has been done on approximation algorithms for these problems.

For the maximum induced bipartite subgraph problem:
Some results for very special classes of graphs -

Zhu (2009): $5/7$ approximation factor algorithm over triangle-free subcubic graphs.

Addario-Berry (2006): Some results for i-triangulated graphs and clique-separable graphs.

Approximation Algorithms for Large Induced Subgraphs

Lund and Yannakakis (1993): It is hard to approximate the largest induced subgraph problem for any non-trivial hereditary property.

Comparatively, very little research has been done on approximation algorithms for these problems.

For the maximum induced bipartite subgraph problem:
Some results for very special classes of graphs -

Zhu (2009): $5/7$ approximation factor algorithm over triangle-free subcubic graphs.

Addario-Berry (2006): Some results for i-triangulated graphs and clique-separable graphs.

Approximation Algorithms for Large Induced Subgraphs

For the maximum induced Planar subgraph problem:

Calinescu et al. (1998): There exists an $\epsilon > 0$ such that there is no $1 - \epsilon$- approximation algorithm unless $P = NP$.

Edward and Farr (2007): $3/(d+1)$-approximation algorithm on graphs of average degree at most $d \geq 4$,
In fact, this algorithm finds an induced series-parallel subgraph (more about these later).

Approximation Algorithms for Large Subgraphs

For the maximum Planar subgraph problem:

Calinescu et al. (1998): There exists an $\epsilon > 0$ such that there is no $1 - \epsilon$- approximation algorithm unless $P = NP$.

Faria et al. (2004): This is true even if the input is a cubic graph.

Till 1990's a number of algorithms were studied but none gave an approximation ratio better than $1/3$, which can be trivially achieved by the Spanning Tree algorithm.

$ST(G) = n - 1$ and $OPT(G) \leq 3n - 6$

Calinescu et al. (1998): $4/9$-approximation algorithm, which is still the best known.
In fact, this algorithm generates an outerplanar subgraph (which gives a $2/3$-approximation algorithm for the maximum outerplanar graph problem).

Approximation Algorithms for Large Subgraphs

For the maximum Planar subgraph problem:

Calinescu et al. (1998): There exists an $\epsilon > 0$ such that there is no $1 - \epsilon$- approximation algorithm unless $P = NP$.

Faria et al. (2004): This is true even if the input is a cubic graph.

Till 1990's a number of algorithms were studied but none gave an approximation ratio better than $1/3$, which can be trivially achieved by the Spanning Tree algorithm.

$ST(G) = n - 1$ and $OPT(G) \leq 3n - 6$

Calinescu et al. (1998): $4/9$-approximation algorithm, which is still the best known.
In fact, this algorithm generates an outerplanar subgraph (which gives a $2/3$-approximation algorithm for the maximum outerplanar graph problem).

Approximation Algorithms for Large Subgraphs

For the maximum Planar subgraph problem:

Calinescu et al. (1998): There exists an $\epsilon > 0$ such that there is no $1 - \epsilon$- approximation algorithm unless $P = NP$.

Faria et al. (2004): This is true even if the input is a cubic graph.

Till 1990's a number of algorithms were studied but none gave an approximation ratio better than $1/3$, which can be trivially achieved by the Spanning Tree algorithm.

$ST(G) = n - 1$ and $OPT(G) \leq 3n - 6$

Calinescu et al. (1998): $4/9$-approximation algorithm, which is still the best known.
In fact, this algorithm generates an outerplanar subgraph (which gives a $2/3$-approximation algorithm for the maximum outerplanar graph problem).

Large Series-Parallel Subgraphs

Planar graphs are characterized as having no $\{K_5, K_{3,3}\}$ minors or subdivisions.

Outerplanar graphs are characterized as having no $\{K_4, K_{2,3}\}$ minors or subdivisions.

How about subgraphs with no K_4 minors or subdivisions? These will be planar but not outerplanar.

These are Series-Parallel graphs.

H is a minor of G if a graph isomorphic to H can be obtained from G by contracting some edges, deleting some edges, and deleting some isolated vertices.

Large Series-Parallel Subgraphs

Planar graphs are characterized as having no $\{K_5, K_{3,3}\}$ minors or subdivisions.

Outerplanar graphs are characterized as having no $\{K_4, K_{2,3}\}$ minors or subdivisions.

How about subgraphs with no K_4 minors or subdivisions? These will be planar but not outerplanar.

These are Series-Parallel graphs.

H is a minor of G if a graph isomorphic to H can be obtained from G by contracting some edges, deleting some edges, and deleting some isolated vertices.

Large Series-Parallel Subgraphs

Series-Parallel graphs are characterized as:

- No K_4 minor or subdivision.
- Arises from a forest by adding parallel edges, subdividing edges, and at the end removing any parallel edges to keep the graph simple.
- tree width ≤ 2 (subgraph of 2-tree).

Large Series-Parallel Subgraphs

The maximum Series-Parallel subgraph problem is *NP*-hard.

Since, the number of edges of a Series-Parallel graph on n vertices is bounded above by $2n - 3$,
the spanning tree algorithm gives a $1/2$-approximation algorithm.

Can we do better?

Large Series-Parallel Subgraphs

The maximum Series-Parallel subgraph problem is *NP*-hard.

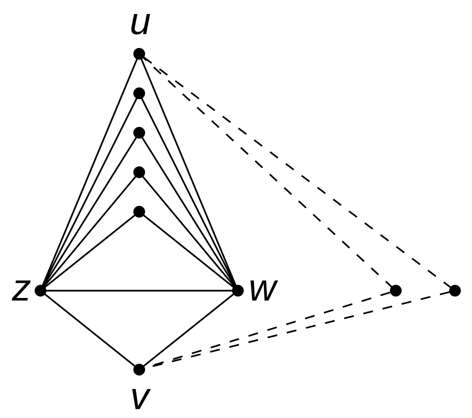
Since, the number of edges of a Series-Parallel graph on n vertices is bounded above by $2n - 3$,
the spanning tree algorithm gives a $1/2$-approximation algorithm.

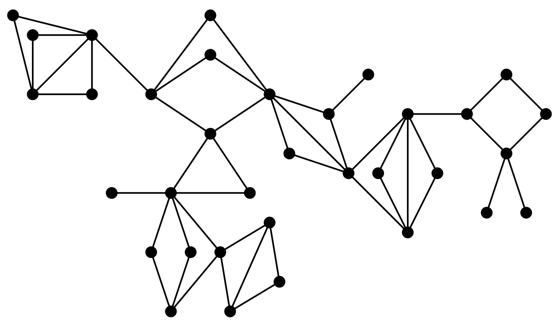
Can we do better?

New Results

Theorem [Calinescu + Fernandes + Kaul, 2009]: $7/12$ approximation algorithm for the maximum Series-Parallel subgraph problem.

The output is a spruce structure: a graph each of whose blocks is either a spruce or an edge.
A spruce consists of two *base* vertices and at least one *tip* vertex, in which each tip vertex is adjacent to exactly the two base vertices.

(a)


(b)


New Results

Theorem [Calinescu + Fernandes + Kaul, 2009]: The maximum spruce structure would give a $2/3$ approximation for the maximum Series-Parallel subgraph.

Theorem [Calinescu + Fernandes + Kaul, 2009]: The maximum spruce structure subgraph problem is *NP*-hard.

New Results

Theorem [Calinescu + Fernandes + Kaul, 2009]: The maximum spruce structure would give a $2/3$ approximation for the maximum Series-Parallel subgraph.

Theorem [Calinescu + Fernandes + Kaul, 2009]: The maximum spruce structure subgraph problem is *NP*-hard.

New Ideas

Comparison with previous algorithms for Planar subgraphs:

- Unlike earlier algorithms, the subgraph we generate is not a tree or an outerplanar graph.
- Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.
- Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.

New Ideas

Comparison with previous algorithms for Planar subgraphs:

- Unlike earlier algorithms, the subgraph we generate is not a tree or an outerplanar graph.
- Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.
- Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.

New Ideas

Comparison with previous algorithms for Planar subgraphs:

- Unlike earlier algorithms, the subgraph we generate is not a tree or an outerplanar graph.
- Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.
- Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.

New Ideas

Comparison with previous algorithms for Planar subgraphs:

- Unlike earlier algorithms, the subgraph we generate is not a tree or an outerplanar graph.
- Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.
- Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.

New Ideas

Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.

If the input graph is a *complete spruce* (spruce with an edge between the base vertices) with $n-2$ tips, then any algorithm that only generates blocks of size at most k would result in an output with a total $n+k-3$ edges.

With large n and fixed k, this is only a $1/2$-approximation.

New Ideas

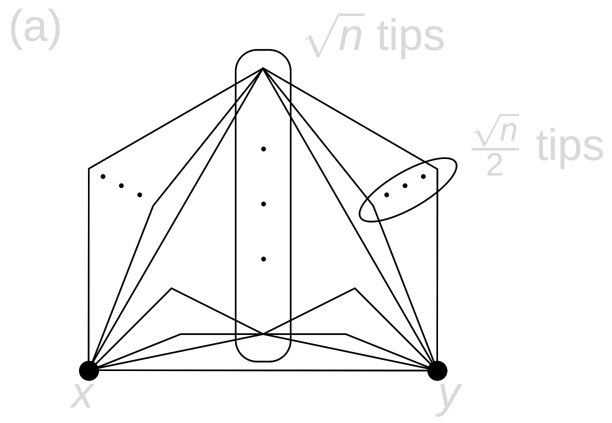
Unlike earlier algorithms, we have to allow blocks of unbounded size in our subgraph.

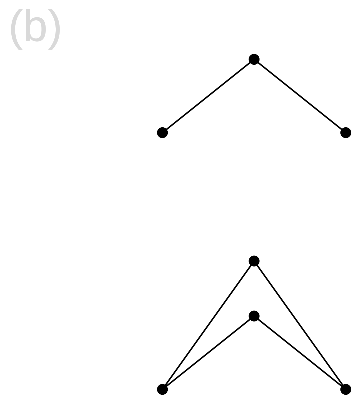
If the input graph is a *complete spruce* (spruce with an edge between the base vertices) with $n-2$ tips, then any algorithm that only generates blocks of size at most k would result in an output with a total $n+k-3$ edges.

With large n and fixed k, this is only a $1/2$-approximation.

New Ideas

Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.





The optimum has n vertices and $2n-3$ edges.

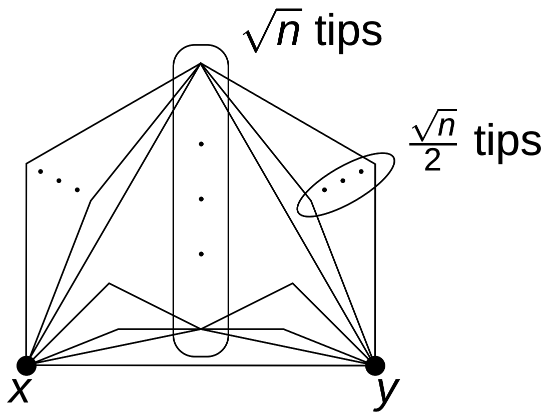
A spruce with base vertices x and y and $\sqrt{n}$ tips. For each of its tips v, there are two complete spruces, one with base vertices x and v, and the other with base vertices v and y, each with $\sqrt{n}/2$ tips.

If an algorithm mistakenly (or greedily) selects the spruce with base vertices x and y, then it cannot add any more spruces and it ends up with about $n+\sqrt{n}$ edges — asymptotically not better than a $1/2$-approximation.

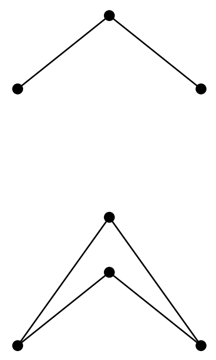
New Ideas

Unlike earlier algorithms, we sometimes have to shrink or throw away previously selected blocks.

(a)



(b)



The optimum has n vertices and $2n-3$ edges.

A spruce with base vertices x and y and $\sqrt{n}$ tips. For each of its tips v, there are two complete spruces, one with base vertices x and v, and the other with base vertices v and y, each with $\sqrt{n}/2$ tips.

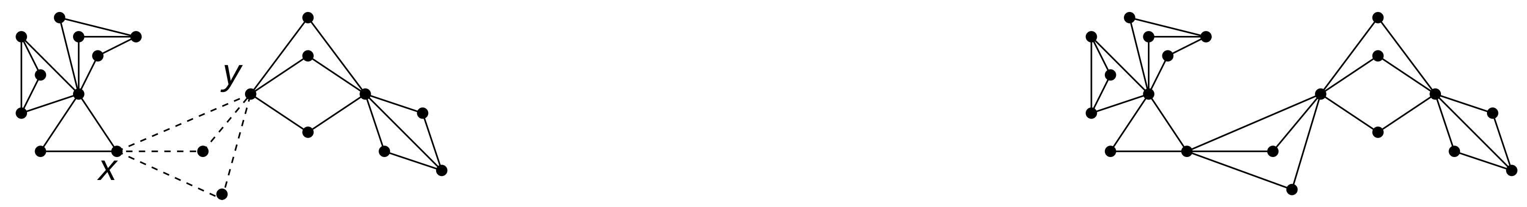
If an algorithm mistakenly (or greedily) selects the spruce with base vertices x and y, then it cannot add any more spruces and it ends up with about $n+\sqrt{n}$ edges — asymptotically not better than a $1/2$-approximation.

The Algorithm

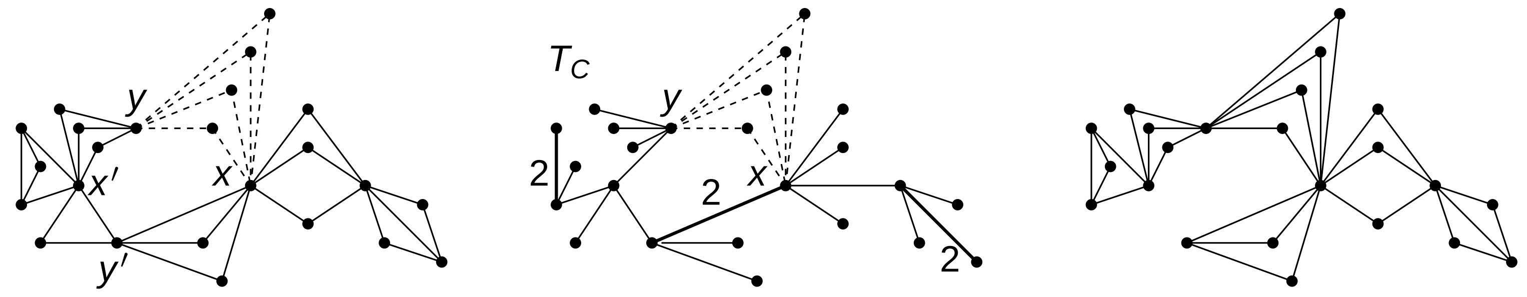
```
CONSTRUCT-SPRUCE-STRUCTURE ($G$)
1   $Q \leftarrow \emptyset$
2   while there are $x$ and $y$ such that $S_Q(x,y)$ is defined
        and $\widehat{gain}(S_Q(x,y)) > w(index_Q(x,y))$ do
3     if $index_Q(x,y)$ is undefined
4       then $Q \leftarrow Q \cup \{S_Q(x,y)\}$
5       else let $x'$ and $y'$ be the endpoints of $index_Q(x,y)$
6            let $S'$ be the spruce in $Q$ containing $x'$ and $y'$
7            $Q \leftarrow Q \setminus \{S'\} \cup \{S_Q(x,y)\}$
8            if $x'$ or $y'$ is a tip of $S'$
9              then let $z$ be between $x', y'$, a tip of $S'$
10                  let $\{e,f\}$ be the edges of $S'$ touching $z$
11                  $S \leftarrow S' - \{e,f\}$
12                  if $S$ is not degenerate nor single edge
13                    then $Q \leftarrow Q \cup \{S\}$
14  add bridges to $Q$ to obtain a connected spanning subgraph of
    $G$
15  return $Q$
```

Local improvement examples

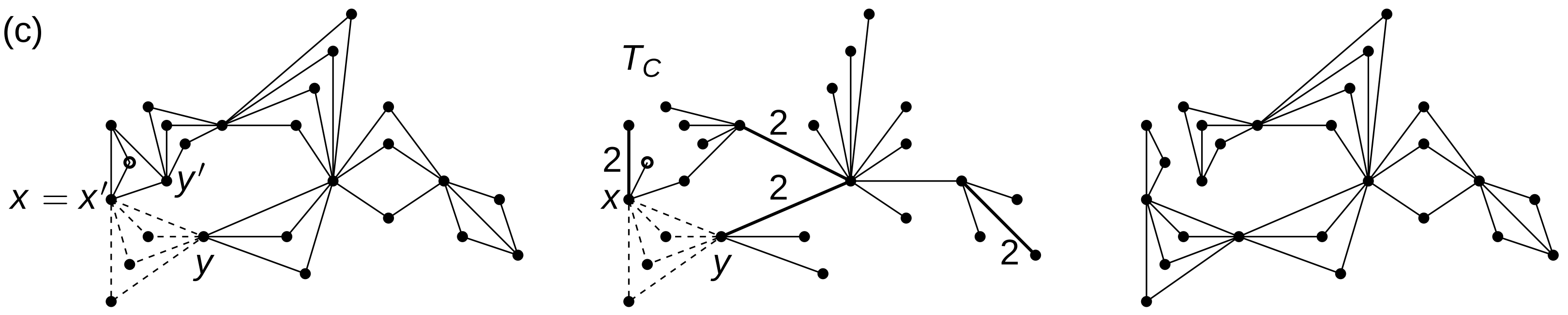
(a)



(b)



(c)



Running Time Analysis

Let $gain(Q)$ = cyclomatic number, the smallest number of edges which must be removed to kill all cycles.

The gain of Q never decreases and, in the iterations in which the gain of Q is same, the number of components increases.

Define $\Phi(Q) = 3\,gain(Q) + c(Q)$, where $c(Q)$ is the number of components of Q when Q is seen as a spanning subgraph of G.

We prove: Every iteration of the algorithm increases the parameter Φ.

$gain(Q) \leq (2n-3) - (n-1) = n-2$,
so $\Phi(Q)$ is bounded by $3(n-2) + n = 4n-6$.

Each iteration can be easily implemented in polynomial time:
$O(n^2)$ pairs x, y for which $S_Q(x, y)$ must be computed and, if possible, used in updating Q.

Running Time Analysis

Let $gain(Q)$ = cyclomatic number, the smallest number of edges which must be removed to kill all cycles.

The gain of Q never decreases and, in the iterations in which the gain of Q is same, the number of components increases.

Define $\Phi(Q) = 3\,gain(Q) + c(Q)$, where $c(Q)$ is the number of components of Q when Q is seen as a spanning subgraph of G.

We prove: Every iteration of the algorithm increases the parameter Φ.

$gain(Q) \leq (2n{-}3) - (n{-}1) = n{-}2$,
so $\Phi(Q)$ is bounded by $3(n{-}2) + n = 4n{-}6$.

Each iteration can be easily implemented in polynomial time:
$O(n^2)$ pairs x, y for which $S_Q(x, y)$ must be computed and, if possible, used in updating Q.

Running Time Analysis

Let $gain(Q)$ = cyclomatic number, the smallest number of edges which must be removed to kill all cycles.

The gain of Q never decreases and, in the iterations in which the gain of Q is same, the number of components increases.

Define $\Phi(Q) = 3\,gain(Q) + c(Q)$, where $c(Q)$ is the number of components of Q when Q is seen as a spanning subgraph of G.

We prove: Every iteration of the algorithm increases the parameter Φ.

$gain(Q) \leq (2n-3) - (n-1) = n-2$,
so $\Phi(Q)$ is bounded by $3(n-2) + n = 4n-6$.

Each iteration can be easily implemented in polynomial time:
$O(n^2)$ pairs x, y for which $S_Q(x, y)$ must be computed and, if possible, used in updating Q.

Approximation ratio ideas - to beat $1/2$

- If significantly many vertices in our structure, we win.
- If OPT has significantly less than $2n$ edges, we win.
- If none of the above, the spruces of OPT have significant *gain*.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, Wagner, \text{two other graphs}\}$.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, Wagner, \text{two other graphs}\}$.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, Wagner, \text{two other graphs}\}$.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, \mathit{Wagner}, \text{two other graphs}\}$.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, \mathit{Wagner}, \text{two other graphs}\}$.

Open Questions II

- Weighted maximum Series-Parallel subgraph problem.
- Maximum induced Series-parallel subgraph problem.
- For fixed r, maximum K_r-minor-free subgraph problem.
- In particular, maximum K_5-minor-free subgraph problem.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, the structural characterization is known - constructed from copies of planar graphs and Wagner's graph by gluing over k-cliques for $k \leq 3$.
- For fixed r, maximum subgraph of tree width $\leq r$.
- In particular, maximum subgraph of tree width ≤ 3.
  Number of edges in such a graph are $\leq 3n - 6$.
  Also, such graphs have no minors from $\{K_5, \mathit{Wagner}, \text{two other graphs}\}$.