COMP36111: Advanced Algorithms I
Part 3: Handling Intractability - a brief introduction

Howard Barringer

Room KB2.20: email: howard.barringer@manchester.ac.uk

December 2010
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
The last part - coping with intractability

• Lectures: Introduction, Steiner Trees and Approximation, Travelling Salesman Problem and Variants

• Exercise/revision session:

• Lectures: On randomised Algorithms

• Possible revision session for whole course
The last part - coping with intractability

• Lectures: Introduction, Steiner Trees and Approximation, Travelling Salesman Problem and Variants
• Exercise/revision session:
• Lectures: On randomised Algorithms
• Possible revision session for whole course
The last part - coping with intractability

- Lectures: Introduction, Steiner Trees and Approximation, Travelling Salesman Problem and Variants
- Exercise/revision session:
- Lectures: On randomised Algorithms
- Possible revision session for whole course
The last part - coping with intractability

- Lectures: Introduction, Steiner Trees and Approximation, Travelling Salesman Problem and Variants
- Exercise/revision session:
- Lectures: On randomised Algorithms
- Possible revision session for whole course
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
Consider you are an engineering machine shop supervisor

- shop floor has 10 working machines
- today you have 50 different milling jobs, each of different, but known, duration, to be completed
- machines are available from 10 am
- you want to be home by 6pm, but can’t leave the machines running
- so, can it be achieved?
Can’t wait for the exact solution

Essentially, the above problem is an instance of the minimum makespan problem, a known NP-hard problem.

Given a set of jobs \( J = \{1, \ldots, n\} \) with duration times \( t_1, \ldots, t_n \) and a positive integer \( m < n \)

Partition \( J \) into \( m \) subsets \( J_1, \ldots, J_m \) such that
\[
\max_{1 \leq i \leq m} \sum_{j \in J_i} t_j
\]
is minimised.

The number of ways for partitioning a set of \( n \) distinct elements into \( m \) non-empty subsets is given by the Stirling number of the second kind.

\[
S(n, m) = S(n - 1, m - 1) + m \times S(n - 1, m)
\]
\[
S(n, n) = 1 \quad S(n, 1) = 1
\]
Can’t wait for the exact solution

Essentially, the above problem is an instance of the minimum makespan problem, a known NP-hard problem.

Given a set of jobs \( J = \{1, \ldots, n\} \) with duration times \( t_1, \ldots, t_n \) and a positive integer \( m < n \)

Partition \( J \) into \( m \) subsets \( J_1, \ldots, J_m \) such that

\[
\max_{1 \leq i \leq m} \sum_{j \in J_i} t_j
\]

is minimised.

The number of ways for partitioning a set of \( n \) distinct elements into \( m \) non-empty subsets is given by the Stirling number of the second kind .

\[
\begin{align*}
S(n, m) &= S(n - 1, m - 1) + m \times S(n - 1, m) \\
S(n, n) &= 1 \\
S(n, 1) &= 1
\end{align*}
\]

\( S(50, 10) = 2615471651586288129201277396577993781727011 \)
And so . . .

- integer programming techniques have long been used to attack NP-hard problems
  - however, there is no guarantee on quality of solution, or running time
  - *Branch and Bound* is a typical integer programming approach
How good is my approximation?

- need to have **measures** on **quality** of approximate solutions
- the longer the computation, the better the quality
- **heuristics** often underly approximation algorithms
- sometimes though such algorithms elude formal analysis
- approximation algorithms are **necessarily polynomial**, and have worst case possible errors, e.g. for a 3/2-approximation algorithm for a minimisation problem $P$
  - for every instance $I$ of $P$, the algorithm gives a solution which is no worse than $3/2$ times the optimum one.
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
Minimum Cost Spanning Tree (MST) - reminder

For an edge-weighted graph $G = (V, E)$ with cost function $c : E \rightarrow \mathbb{R}$

Find a subgraph $G' = (V, E')$ such that

(i) for distinct vertices $u, v \in V$, there’s one path from $u$ to $v$

(ii) there is no other subgraph $G'' = (V, E'')$ satisfying (i) with lower cost, i.e.

$$\sum_{e'' \in E''} c(e'') < \sum_{e' \in E'} c(e')$$

![Minimum cost spanning tree (cost = 15)](image_url)
MST admits polynomial-time algorithms

Fortunately, there are good algorithms for computing minimum spanning trees of graphs.

Famously:

- Prim’s Algorithm — $O(n^2)$, for $n = |V|$
- Kruskal’s Algorithm — $O(e \log(e))$, for $e = |E|$
A variant of MST

Suppose the vertices of the graph are partitioned into two sets, required and others.

The minimal cost spanning tree over just the required vertices may not be the most cost effective.

By including (some) of the other vertices, lower cost trees may be obtained.
The Steiner Tree Problem

Given an edge-weighted graph $G = (V, E)$ with cost function $c : E \rightarrow \mathbb{R}$ and vertices partitioned into sets $R$, the required vertices, and $S$, the Steiner vertices

Find the minimal cost tree in $G$, say $G' = (V', E')$, $R \subseteq V'$ and $(V' - R) \subseteq S$. 

Unfortunately: there are no known polynomial algorithms for solving the Steiner Tree problem. Most versions of the problem are NP-complete problems.
The Steiner Tree Problem

Given an edge-weighted graph \( G = (V, E) \) with cost function \( c : E \to \mathbb{R} \) and vertices partitioned into sets

\( R \), the required vertices, and
\( S \), the Steiner vertices

Find the minimal cost tree in \( G \), say \( G' = (V', E') \), \( R \subseteq V' \) and \( (V' - R) \subseteq S \).

Unfortunately: there are no known polynomial algorithms for solving the Steiner Tree problem. Most versions of the problem are NP-complete problems.
Is this a real problem?

Solutions to (large) Steiner Tree problems required for

- circuit layout in pcb and VLSI design
- electricity distribution in power grid networks
- phylogenetic tree construction in biological systematics and linguistics
- oil and water pipeline network construction
- ...

...to name just a few...
Approximation theory - 1

Assume $\Pi$ is an NP-optimization problem.

An instance $I$ of $\Pi$ has a nonempty set of feasible solutions.

Each solution is given an objective function value ($\in \mathbb{Q}^+$).

There will be polynomial-time algorithms for

- checking validity of an instance $I$
- checking feasibility of a solution
- computing the objective function value of a solution

An *optimal* solution is one that achieves an optimal objective function value — we will use $OPT_\Pi(I)$ to denote this value.
A factor $f$ approximation algorithm $\mathcal{A}$ for $\Pi$ will generate a feasible solution that has an objective function value within factor $f$ of the optimal solution value, $OPT$. 
Example: metric Steiner trees

A *metric* Steiner tree is a Steiner tree graph $G = (V, E)$ which is complete with edges satisfying $c(u, w) \leq c(u, v) + c(v, w)$, for vertices $u, v$ and $w$.
Example: metric Steiner trees

A *metric* Steiner tree is a Steiner tree graph \( G = (V, E) \) which is complete with edges satisfying \( c(u, w) \leq c(u, v) + c(v, w) \), for vertices \( u, v \) and \( w \).

Note that the optimal cost Steiner tree is \( \frac{3}{4} \) the cost of the minimum spanning tree on required vertices.
Approximation algorithm for metric Steiner tree problem

The minimum-cost spanning tree algorithm (MST) applied to the required vertices of a metric Steiner tree is a factor 2 approximation algorithm for the metric Steiner tree problem (mSTP), and thus yields a solution with cost $\leq 2 \times OPT$. 
Is this a metric Steiner tree?

Is the graph complete?

Does it have the triangulation property?
Is this a metric Steiner tree?

Fails on triangulation: $3 \leq 1 + 1$
Proving the factor 2 approximation

1. Assume an optimal metric Steiner tree of cost $OPT$.
2. Double the edges of this tree, and construct an Euler tour, of cost $2 \times OPT$.
3. Short circuit the paths through the Steiner vertices, and previously visited vertices, to give a Hamiltonian cycle with cost $\leq 2 \times OPT$.
4. Remove an edge from the cycle to create a spanning tree with cost $\leq 2 \times OPT$ over just the required vertices.
5. Thus, the MST over the required vertices costs $\leq 2 \times OPT$. 
Importantly...

There is an approximation factor preserving (polynomial time) reduction from the Steiner tree problem to the metric Steiner tree problem.

Hence, an approximation factor obtained for the mSTP will carry over to the full STP.
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
A little history: Fermat points ...

In the 17th century, Fermat studied how to find a point that has the shortest total distance to three points in the Euclidean plane.

![Diagram of Fermat point]

The generalisation to $n$ points, rather than 3, is still referred to as the Fermat problem.

Note: this isn’t quite the Steiner tree problem — why?
and: Schumacher’s paradox for Gauss

In the 19th century, Schumacher wrote to Gauss about the following paradox.

For a convex quadrilateral, the intersection of the two diagonals gives the Fermat point. Move the end points of an edge together. The intersection point moves towards end points. In the limit, we have a triangle. But the Fermat point of a triangle is not a vertex.
Euclidean Steiner trees

For Steiner trees, more than one “Steiner” point can be introduced to graph to get minimal tree.

Assume a unit square

\[ AF + BF + CF + DF = 2\sqrt{2} \]

\[ AF + BF + FG + CG + DG = 1 + \sqrt{3} \]
Key Properties

A (minimal) Euclidean Steiner tree satisfies three key properties:

1. it is a tree;
2. Any two edges of the tree meet at an angle $\geq 120^\circ$;
3. A Steiner point has degree of at least 3.

Thus we have:

Steiner points have exactly degree 3 (from (2) and (3) above);
For a graph with $n$ nodes, there can be at most $n-2$ Steiner points introduced;
A Steiner tree is full if there are $n-2$ Steiner points, and then each non-Steiner node has degree 1.
Example Euclidean Steiner tree

But there can be many possibilities ....
Two obvious topologies for a convex quadrilateral

A graph can have many different Steiner topologies (given by the connections), here are two for a convex quadrilateral.

Each topology has a unique relatively minimal tree.

The minimal Euclidean Steiner tree will be the smallest of all the relatively minimal trees.
Outline of an exact algorithm for EST

Given a set of points $A$ (the required points of the Steiner tree),

Enumerate all the different Steiner topologies for $A$

Compute the “cost” of each topology’s relatively minimal tree

Choose the relatively minimal tree of the Steiner topology that is smallest.

The difficulty of this approach is the huge number of topologies that may exist.

For $n$ required points (i.e. $n = |A|$), there are: $\frac{(2n - 4)!}{2^{n-2}(n - 2)!}$ full Steiner topologies...
Outline of an exact algorithm for EST

Given a set of points $A$ (the required points of the Steiner tree),

Enumerate all the different Steiner topologies for $A$

Compute the “cost” of each topology’s relatively minimal tree

Choose the relatively minimal tree of the Steiner topology that is smallest.

The difficulty of this approach is the huge number of topologies that may exist.

For $n$ required points (i.e. $n = |A|$), there are: $\frac{(2n-4)!}{2^{n-2}(n-2)!}$ full Steiner topologies...

Wow, for $n = 10$, this is 2,027,025 and for $n = 12$ it is 654,729,075
Outline of an exact algorithm for EST

Given a set of points $A$ (the required points of the Steiner tree),

Enumerate all the different Steiner topologies for $A$
Compute the “cost” of each topology’s relatively minimal tree
Choose the relatively minimal tree of the Steiner topology that is smallest.

The difficulty of this approach is the huge number of topologies that may exist.

For $n$ required points (i.e. $n = |A|$), there are: $\frac{(2n-4)!}{2^{n-2}(n-2)!}$ full Steiner topologies...

Wow, for $n = 10$, this is 2,027,025 and for $n = 12$ it is 654,729,075

When the restriction to full is removed the numbers are much larger, e.g. for $n = 10$, there are 2,382,538,725 topologies!
Approximation algorithms for EST

Clearly, the minimum spanning tree over the required points will be an approximation. This is computable in $n \log n$ time.

This can be used to gauge performance of approximation algorithms.

The Steiner Ratio $= \min_A \frac{\text{length of SMT on } A}{\text{length of MST on } A}$ was conjectured to be $\frac{\sqrt{3}}{2}$ by Gilbert and Pollack in 1968.

Following a history of intermediate results for fixed sizes of $n$, the conjecture was proved in 1990 (Du and Hwang).
A simple Steiner Insertion Algorithm

See Dreyer and Overton’s paper (in refs at end of slides).

Assume \( A \) is set of required/fixed/terminal points.

1. Compute an MST over \( A \)
2. For each edge connecting fixed points \((x, y)\)
   2.1 For an edge \((y, z)\) that meets \((x, y)\) at the smallest angle, with \(z\) either a Steiner or fixed point
   2.2 If the angle is less than 120\(^\circ\) then
      2.2.1 put a new Steiner point \( s \) on top of \( y \)
      2.2.2 remove the edges \((x, y), (y, z)\)
      2.2.3 add edges \((x, s), (y, s)\) and \((z, s)\)
3. Apply a local optimisation algorithm over the new tree, to optimise placement of the additional Steiner points
Comments on simple SIA

Ignoring the final step of SIA, this is an $O(n^3)$ algorithm. The quality of the result, however, is largely bound up in the local optimisation algorithm.

Therefore the cost depends largely on how much time is committed to local optimisation.

Experimentation has shown, see Dreyer and OVerton, that between 0.7 and 3.4% performance gain over MST.

Much prior work on approximation algorithms for the Euclidean Steiner Minimal Tree exists.
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
TSP: Review
TSP — by brute force

Each town in directly connected to every other town, with known distance.

Must find the shortest (circular) route from a given town that visits all the towns.

Can enumerate all possible routes to find shortest.

Given \( n \) towns, how many routes must be examined?
TSP — by brute force

Each town in directly connected to every other town, with known distance.

Must find the shortest (circular) route from a given town that visits all the towns.

Can enumerate all possible routes to find shortest.

Given \( n \) towns, how many routes must be examined?

\[
\frac{(n - 1)!}{2}
\]
TSP — by brute force

Each town in directly connected to every other town, with known distance.

Must find the shortest (circular) route from a given town that visits all the towns.

Can enumerate all possible routes to find shortest.

Given $n$ towns, how many routes must be examined?

$$\frac{(n - 1)!}{2}$$

For 15 cities, there are 43,589,145,600 possible tours
TSP — perhaps not such a brute!

Amazingly, TSP still attracts considerable research interest.

There are some (much) faster techniques, over the brute force approach, for exact solutions.
TSP — perhaps not such a brute!

Amazingly, TSP still attracts considerable research interest.

There are some (much) faster techniques, over the brute force approach, for exact solutions.

Using linear programming, in 2001, exact solution to 15,112 German towns found.

It required the equivalent of 22.6 years using a single 0.5GHz processor.
A tour of Sweden

from: http://www.tsp.gatech.edu/sweden/index.html

In May 2004, the traveling salesman problem of visiting all 24,978 cities in Sweden was solved: a tour of length 855,597 TSPLIB units (approximately 72,500 kilometers) was found and it was proven that no shorter tour exists. At the time of the computation, this was the largest solved TSP instance, surpassing the previous record of 15,112 cities through Germany set in April 2001.
TSP: Concorde Solver

A quick demonstration... see: http://www.tsp.gatech.edu
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
Approximation is problematic for full TSP

Unless \( P = NP \), TSP cannot be approximated within a factor of any polynomial time computable function \( f(n) \).

If we assume the existence of a factor \( f(n) \) approximation algorithm \( A \).

We can show Hamiltonian cycles can be found in polynomial time.

But Hamiltonian cycle problem is known to be \( NP \)-hard.
Counter argument ... in pictures

Hamiltonian Cycle

TSP tour length = 5

No Hamiltonian Cycle

TSP tour length = 3 + 2.f(5.5)

Hamiltonian Cycle TSP tour length = 5
No Hamiltonian Cycle

f(5.5)
Metric TSP — factor 2 approximation

The above proof relied on assigning edge costs that didn’t satisfy the triangulation property. If we restrict to Metric TSP, poly factor approximations can be found.

1. Find an MST of graph $G$.
2. Double each edge of the MST to give an Eulerian graph.
3. Find an Euler tour $T_E$ on the expanded graph.
4. Now take a tour $T_{TSP}$ that visits nodes of $G$ in the order they appear in $T_E$.

Note: $\text{cost}(\text{MST}) \leq \text{OPT}$. The Euler tour has $\text{cost}(T_E) = 2 \times \text{cost}(\text{MST})$. And $\text{cost}(T_{TSP}) \leq \text{cost}(T_E)$. Hence the factor 2 approximation.
A factor $3/2$ approximation for metric TSP

1. Find an MST of graph $G$.
2. Compute a minimum cost perfect matching, $M$, on the set of odd-degree vertices of the MST.
3. Add $M$ to the MST and find its Euler tour $T_E$.
4. Now take a tour $T_{TSP}$ that visits nodes of $G$ in the order they appear in $T_E$.

Note: the size of the set of odd-degree vertices of a minimum spanning tree is even. This then gives that the $\text{cost}(M) \leq \frac{\text{OPT}}{2}$. Hence we get that
\[
\text{cost}(T_{TSP}) \leq \text{cost}(T_E) + \text{cost}(M) \leq \text{OPT} + \frac{\text{OPT}}{2}.
\]
Outline

Introduction
  Course Structure
  Motivation

Steiner Trees and Approximation
  General Problem
  Euclidean Steiner Trees

Travelling Salesman
  General Problem
  Towards approximations

Literature
  Further reading
Further Reading - General

General books:


An excellent book with extensive coverage of the topic and major algorithms and analysis
Further Reading - Steiner trees 1

M. Herring: The Euclidean Steiner tree problem.
http://www.denison.edu/academics/departments/mathcs/herring.pdf

A short note introducing the problem and a couple of approaches to its solution.


A short readable account of the problem and two approaches.


Seminal early theoretical work investigating these trees


An extensive survey paper (up to 1989) on Steiner tree problems, covering Euclidean, rectilinear, graphic, phylogenetic and other generalisations. Includes some 310 references!
Further Reading - Steiner trees 2


Another seminal result. Difficult!


A collection of advanced papers on Steiner trees and associated algorithms


A collection of advanced topic papers relating to applications of Steiner tree problems in industry. Not an easy read but gives the state of the art.
Further Reading - TSP

The TSP website. http://www.tsp.gatech.edu/

A major site for historical and current information about the TSP problem. The site is also the home for the Concorde TSP solver — available as a download.


A major resource book on the computational methods underlying the Concorde TSP solver.


A collection of advanced papers on algorithms and analysis techniques (theoretical as well as experimental) for the TSP problem and variants.