Some Open Problems in Approximation Algorithms

David P. Williamson

School of Operations Research and Information Engineering
Cornell University

April 25, 2018
Outline

- About writing the book
- Our ten open problems (Chapter 17)
- Some thoughts about the field
FAQ #1: How long did it take to write the book?

Answer: 13-14 years, depending on how you count.

Fax from July 16, 1997 with book outline.

Spring 1998: Taught course from outline at Columbia University (IEOR).

Fall 1998: Taught course from outline at Cornell University (ORIE).

Spring 2000: Taught course from outline at MIT (EECS).

FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
Fax from July 16, 1997 with book outline.
FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
Fax from July 16, 1997 with book outline.

Spring 1998: Taught course from outline at Columbia University (IEOR).
FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
Fax from July 16, 1997 with book outline.

Spring 1998: Taught course from outline at Columbia University (IEOR).
Fall 1998: Taught course from outline at Cornell University (ORIE).
FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
Fax from July 16, 1997 with book outline.

Spring 1998: Taught course from outline at Columbia University (IEOR).
Fall 1998: Taught course from outline at Cornell University (ORIE).
Spring 2000: Taught course from outline at MIT (EECS).
FAQ #1: How long did it take to write the book?
Answer: 13-14 years, depending on how you count.
Fax from July 16, 1997 with book outline.

Spring 1998: Taught course from outline at Columbia University (IEOR).
Fall 1998: Taught course from outline at Cornell University (ORIE).
Spring 2000: Taught course from outline at MIT (EECS).
1. Introduction
   A. What are approximation algorithms? Why approximation algorithms?
   B. An introduction to approximation algorithms: 2-approximation algorithms for vertex cover
      1. Unweighted via matchings
      2. Linear programming and deterministic rounding (Hochbaum; Nemhauser-Trotter?)
      3. Constructing a dual (Bar-Yehuda and Even)
      4. Randomization (Pitrd)
   C. Set cover
      1. Extending arguments: Hochbaum, Bar-Yehuda & Even, Pits
      2. A Better Algorithm for Unweighted (Johnson, Lovasz)
      3. A Better Algorithm for Weighted (Chvatal)

2. Approximation Algorithms for Classic Problems
   A. Traveling Salesman Problem
      1. Doubling a min-cost spanning tree
      2. Christofides' Algorithm
   B. Steiner Trees and a simple 2-approximation algorithm
   C. Knapsack and Dynamic Programming (Ibarra-Kim)

3. Bin-Packing
   A. Simple proof of a Johnson algorithm?
   B. de la Vega and Lueker
   C. Kannkar and Karp

4. Clique, Independent Sets, and Colorings
   A. Johnson's algorithm
   B. Wolgamot's O(n^3) algorithm for 3-colorable graphs
   C. Boppana-Halldorsen (how hard is this?)
   D. Edge-coloring (Vizing)

5. Scheduling
   A. Graham for P[ij, max]
   B. Dynamic programming: PTAS for P[ij, max] (Hochbaum & Shmoys)

III. The Power of Randomization
   A. Flipping Coins and Easy Randomization
      1. MAX CUT (Sahni-Gonzalez) and MAX SAT (Johnson)
      2. Flipping Best Coins: MAX SAT (Liebeherr and Specker)
      3. Derandomization: the Method of Conditional Expectations
   B. Randomized Rounding
      1. MAX SAT (Goemans-Williamson)
      2. Multicommodity Flows (Raghavan-Thompson)
      3. Prize-collecting TSP revisited (Goemans)
      4. Set cover and oversampling
      5. MAX CUT in dense graphs (Arora-Karger-Karpinski)
   C. Rounding Semidefinite Programs
The first outline (2)

1. MAX CUT (Goemans-Williamson)
2. Coloring (Karger-Motwani-Sudan)

IV. Finding cuts in graphs & metric methods & applications
A. A warm-up: multway cuts (Dahlhaus et al.)
B. The metric method: multicut (Garg-Vazirani-Yannakakis)
C. Balanced separators (Even-Naor-Rao-Schieber)
D. Applications of balanced separators (linear arrangement, etc.)

V. Network Design Problems, filtering, and the primal-dual method
A. Facility location and filtering
1. Early results (Dyer-Frieze, Hochbaum-Shmoys)
2. Filtering (Shmoys-Tardos-Aardal)
B. The primal-dual method
1. 0-1 proper functions (Goemans-Williamson)
2. Survivable Network Design with multiple edge copies (Agrawal-Klein-Ravi, Goemans-Williamson)
3. Prize-collecting TSP revisited again (Goemans-Williamson)?
4. k-MST (Garg)?
5. A non-network design application: feedback vertex sets (Bafna-Berman-Fujito as interpreted by Chudak-Goemans-Hochbaum-Williamson)

C. More network design
1. Steiner trees (Zeikovsky)
2. Low-degree trees (Furer-Raghavan)

VI. Dynamic Programming
A. Baker’s PTAS for planar graphs
B. Arora’s PTAS for Euclidean problems

VII. More Scheduling

VIII. Hardness results and implications
A. Statement of ALMSS and proof of hardness of MAX SAT
B. The label-cover theorem and implications (?)
C. Statement of hardness of clique, coloring, Hasted’s result for equations over GF(2)
I. Introduction - the book in miniature
   A. The what/why of approx. alg.
   B. An introduction to approx. alg.: Set cover
      1. Unweighted greedy
      2. Deterministic rounding (Kernighan)
      3. Introducing a dual: J. Horakova's dual rounding
      4. Primal-dual (Bar-Yehuda & Even)
      5. Unweighted greedy (Dvoretzky)

II. TSP & deterministic rounding??
   A. TSP
      1. Doubling on MST & nearest addition
      2. Christofides
      3. The PCST
         a. Ellipsoid method
         b. Deterministic rounding
            (but need parameterized property)
   B. k-median - easy det. rounding from CGST??

III. Dynamic Programming & Bin packing
   A. Knapsack
   B. PII Conv
      1. List scheduling
      2. PTAS (Vaz HS)
   C. Bin packing
      1. Exact GT
      2. Dileep & Lueker's (HS) exercise
      3. Kemper & Karp

IV. Randomization
   A. Max Sat: a case study
      1. Randomized Johnson
      2. Determinization: conditional expectations
      3. Almost Bernoulli coins: Lieberherr & Spencer
      4. Randomized rounding: GW
         5. Non-uniform and additive GW
   B. More non-uniform round: PCST (Goemans)
A later outline (2)

C. Max Cut: a case study
1. Easy \( 1/2 \)- (SG)
2. PTAS in dense graphs (AKK)
   Intro to Cheeger bounds

D. SDP
1. Max Cut
2. Quadratic programming (Kolountzakis)
3. Coloring
   a. Wägner's 0.01
   b. MDS alg. 1
   c. MDS alg. 2
4. Belief? Bisection?

I. Cuts & Metrics
A. Multicut cuts
1. Easy 2 (Dinitz et al.)
2. Improved LP + by metric view (CKR)

B. Multicuts (GKY)
C. Balanced separators (ENRS)
   1. applications thereof
D. Sparsest cut & LLR low distortion embeddings (Khale as ex.)

II. The primal-dual method
A. Intro to Generalized Steiner trees
B. Facility Location (GV)
C. PCST??
D. Lagrangian relaxation
   1. K-Median
   2. K-MST (5)
E. FVS?
A later outline (3)

II. Local search
   A. Facility location (Guha & Charikar)
   B. Low degree spanning tree (Furer & Razgon)

III. Advanced deterministic rounding: Jain

IV. Advanced dynamic programming: Asoro/Mitchell

V. Advanced Scheduling
   A. Simple rules and simple analyses
      1. EDD for \( \text{Min} \) \( L_{\text{max}} \)
      2. EPT for \( \text{Min} \) \( L_{\text{max}} \)
   B. LP based rules
      1. \( \text{Gi} \) for \( \text{Min} \) \( L_{\text{max}} \)
      2. Generalized assignment
   C. Randomization
      1. \( e^{-1} \) for \( \text{Min} \) \( L_{\text{max}} \)
      2. \( 2 \) for \( \text{Min} \) \( L_{\text{max}} \)

VI. Hardness bounds
   A. Statement of ALMPS \( \Rightarrow \) no PTAS for MAX SAT
   B. Label cover
   C. Hardness of clique, coloring, etc. over GF(2)
Some principles

Guiding principles

- Presentation of simple, elegant algorithms and analysis
- Techniques widely applicable (over exhaustive, but hodge-podge coverage of particular problems & results)
- Illustrate the power of new techniques, formulations by revisiting core problems many times: PCST, k-median/fac loc, FVS, Max Sat, Max Cut, Multiway cut
Problems or techniques?

A pedagogical issue: teach problems or techniques?
Problems or techniques?

A pedagogical issue: teach problems or techniques?

Hard because historically the two are intertwined; for example:

- Deterministic rounding/primal-dual and set cover/vertex cover (Hochbaum, Bar-Yehuda and Even)
- Randomized rounding and integer multicommodity flow (Raghavan and Thompson)
- SDP and max cut (Goemans and W)
- Region-growing and multicut (Garg, Vazirani, Yannakakis)
Problems or techniques?

A pedagogical issue: teach problems or techniques?

Hard because historically the two are intertwined; for example:

- Deterministic rounding/primal-dual and set cover/vertex cover (Hochbaum, Bar-Yehuda and Even)
- Randomized rounding and integer multicommodity flow (Raghavan and Thompson)
- SDP and max cut (Goemans and W)
- Region-growing and multicut (Garg, Vazirani, Yannakakis)

If techniques, then some algorithms are hard to categorize; e.g. what is Christofides’ algorithm?
Problems or techniques?

A pedagogical issue: teach problems or techniques?

Hard because historically the two are interwined; for example:

- Deterministic rounding/primal-dual and set cover/vertex cover (Hochbaum, Bar-Yehuda and Even)
- Randomized rounding and integer multicommodity flow (Raghavan and Thompson)
- SDP and max cut (Goemans and W)
- Region-growing and multicut (Garg, Vazirani, Yannakakis)

If techniques, then some algorithms are hard to categorize; e.g. what is Christofides’ algorithm?

If problems, then what is the main takeaway of the course?
Next: ten open problems from the book.
Problem 10: A primal-dual algorithm for the maximum cut problem

Maximum Cut Problem

Input: An undirected graph $G = (V, E)$ with nonnegative edge weights $w_{ij} \geq 0$ for all $i, j \in V$.

Goal: Find a set of vertices $S \subseteq V$ that maximizes $\sum_{i \in S, j \notin S} w_{ij}$.
Problem 10: A primal-dual algorithm for the maximum cut problem

What’s known?

- an \((\alpha - \epsilon)\)-approximation algorithm using semidefinite programming (Goemans, W 1995) for

\[
\alpha = \min_{-1 \leq x \leq 1} \frac{1}{\pi} \arccos(x) \approx \frac{1}{2(1 - x)} \approx .87856,
\]

and any \(\epsilon > 0\).
Problem 10: A primal-dual algorithm for the maximum cut problem

What’s known?

- an \((\alpha - \epsilon)\)-approximation algorithm using semidefinite programming (Goemans, W 1995) for

\[
\alpha = \min_{-1 \leq x \leq 1} \frac{1}{\pi} \arccos(x) \cdot \frac{1}{\frac{1}{2}(1 - x)} \approx 0.87856,
\]

and any \(\epsilon > 0\).

- Assuming the unique games conjecture, no \((\alpha + \epsilon)\)-approximation algorithm is possible unless \(P = NP\) (Khot, Kindler, Mossel, O’Donnell 2007; Mossel, O’Donnell, Oleszkiewicz 2010)

- Assuming the unique games conjecture, no \((\alpha + \epsilon)\)-approximation algorithm is possible unless \(P = NP\) (Khot, Kindler, Mossel, O’Donnell 2007; Mossel, O’Donnell, Oleszkiewicz 2010)
Problem 10: A primal-dual algorithm for the maximum cut problem

What’s known?

- An \((\alpha - \epsilon)\)-approximation algorithm using semidefinite programming (Goemans, W 1995) for

\[
\alpha = \min_{-1 \leq x \leq 1} \frac{1}{\pi} \arccos(x) \frac{1}{2(1 - x)} \approx 0.87856,
\]

and any \(\epsilon > 0\).

- Assuming the unique games conjecture, no \((\alpha + \epsilon)\)-approximation algorithm is possible unless \(P = NP\) (Khot, Kindler, Mossel, O’Donnell 2007; Mossel, O’Donnell, Oleszkiewicz 2010)

- No \(\beta\)-approximation algorithm possible for constant \(\beta > \frac{16}{17} \approx 0.941\) unless \(P = NP\) (Håstad 1997).
Problem 10: A primal-dual algorithm for the maximum cut problem

The problem:
Solving the semidefinite program is computationally expensive. Can one obtain an $(\alpha - \epsilon)$-approximation algorithm for the problem via computationally easier means? E.g. a primal-dual algorithm?
Problem 10: A primal-dual algorithm for the maximum cut problem

The problem:
Solving the semidefinite program is computationally expensive. Can one obtain an $(\alpha - \epsilon)$-approximation algorithm for the problem via computationally easier means? E.g. a primal-dual algorithm?

A potential start:

Open question: Give a tight analysis for Trevisan’s algorithm.
Some work

There’s a bunch of papers extending multiplicative weight update/exponential penalty algorithms to semidefinite programming, but this isn’t exactly what we had in mind. E.g. Klein, Lu (STOC 1996) Arora, Kale (STOC 2007, JACM 2016)

Maybe closer: approximating max cut via random walk (Kale, Seshadri ICS 2011).
Lightweight approximation algorithms

*Lightweight* approximation: can we replace more expensive computational primitives with cheaper ones and still get the same guarantees?

SDP $\rightarrow$ SOCP $\rightarrow$ LP $\rightarrow$ Network flow/primal-dual $\rightarrow$ greedy

Ellipsoid $\rightarrow$ polysized LP $\rightarrow$ ...
Lightweight approximation algorithms

*Lightweight* approximation: can we replace more expensive computational primitives with cheaper ones and still get the same guarantees?

SDP $\rightarrow$ SOCP $\rightarrow$ LP $\rightarrow$ Network flow/primal-dual $\rightarrow$ greedy
Ellipsoid $\rightarrow$ polysized LP $\rightarrow$ \ldots

Lots of work already done in this direction (e.g. Poloczek and Schnitger (SODA 2010), randomized $\frac{3}{4}$-approximation algorithm for MAX SAT without solving LP or network flow), but let’s do more.
Problem 9: Coloring 3-colorable graphs

Input: An undirected, 3-colorable graph $G = (V, E)$.

Goal: Find a $k$-coloring of the graph with $k$ as small as possible.
Problem 9: Coloring 3-colorable graphs

What’s known?

- A poly-time algorithm using semidefinite programming that uses at most $\tilde{O}(n^{0.211})$ colors (Arora, Chlamtac, Charikar 2006)
Problem 9: Coloring 3-colorable graphs

What’s known?

- A poly-time algorithm using semidefinite programming that uses at most $\tilde{O}(n^{0.211})$ colors (Arora, Chlamtac, Charikar 2006)
- It is NP-hard to decide if a graph needs only 3 colors or at least 5 colors (Khanna, Linial, Safra 2000)
Problem 9: Coloring 3-colorable graphs

What’s known?

- A poly-time algorithm using semidefinite programming that uses at most $\tilde{O}(n^{0.211})$ colors (Arora, Chlamtac, Charikar 2006)
- It is NP-hard to decide if a graph needs only 3 colors or at least 5 colors (Khanna, Linial, Safra 2000)
- Assuming a variant of the unique games conjecture, for any constant $k > 3$, it is NP-hard to decide if a graph needs only 3 colors or at least $k$ colors (Dinur, Mossel, Regev 2009)
Problem 9: Coloring 3-colorable graphs

What’s known?

- A poly-time algorithm using semidefinite programming that uses at most $\tilde{O}(n^{0.211})$ colors (Arora, Chlamtac, Charikar 2006)
- It is NP-hard to decide if a graph needs only 3 colors or at least 5 colors (Khanna, Linial, Safra 2000)
- Assuming a variant of the unique games conjecture, for any constant $k > 3$, it is NP-hard to decide if a graph needs only 3 colors or at least $k$ colors (Dinur, Mossel, Regev 2009)

The problem:
Give an algorithm that uses $O(\log n)$ colors for 3-colorable graphs (or show this is not possible modulo some complexity theoretic condition).
What’s known now?

- A poly-time algorithm using semidefinite programming that uses at most $\tilde{O}(n^{0.19996})$ colors (Kawarabayashi, Thorup 2017)
Problem 8: Scheduling related machines with precedence constraints \((Q|\text{prec}|C_{\text{max}})\)

Scheduling related machines with precedence constraints

Input:
- \(n\) jobs with processing requirements \(p_1, \ldots, p_n \geq 0\).
- \(m\) machines with speeds \(s_1 \geq s_2 \geq \cdots \geq s_m > 0\).
- A precedence relation \(\prec\) on jobs.

Goal: Find a schedule of minimum length in which all jobs are completely scheduled and if \(j \prec j'\), then job \(j\) completes before job \(j'\) starts. Job \(j\) on machine \(i\) takes \(p_j/s_i\) units of time.
Problem 8: Scheduling related machines with precedence constraints

What’s known?

- If machines are identical \((s_1 = s_2 = \cdots = s_m)\) then there is a 2-approximation algorithm (Graham 1966).
Problem 8: Scheduling related machines with precedence constraints

What’s known?

- If machines are identical ($s_1 = s_2 = \cdots = s_m$) then there is a 2-approximation algorithm (Graham 1966).
- For general case, an $O(\log m)$-approximation algorithm is known (Chudak and Shmoys 1999; Chekuri and Bender 2001).
Problem 8: Scheduling related machines with precedence constraints

What’s known?

- If machines are identical ($s_1 = s_2 = \cdots = s_m$) then there is a 2-approximation algorithm (Graham 1966).
- For general case, an $O(\log m)$-approximation algorithm is known (Chudak and Shmoys 1999; Chekuri and Bender 2001).
- If machines are identical, and given a variant of the unique games conjecture, then no $\alpha$-approximation algorithm is possible for $\alpha < 2$ unless $P = NP$. (Svensson STOC 2010).
Problem 8: Scheduling related machines with precedence constraints

What’s known?

- If machines are identical \((s_1 = s_2 = \cdots = s_m)\) then there is a 2-approximation algorithm (Graham 1966).
- For general case, an \(O(\log m)\)-approximation algorithm is known (Chudak and Shmoys 1999; Chekuri and Bender 2001).
- If machines are identical, and given a variant of the unique games conjecture, then no \(\alpha\)-approximation algorithm is possible for \(\alpha < 2\) unless \(P = NP\). (Svensson STOC 2010).

The problem:

Give an \(\alpha\)-approximation algorithm for some constant \(\alpha\), or show that \(O(\log m)\) is the best possible modulo the unique games conjecture.
What’s known now?

- No progress.
Problem 7: Scheduling unrelated machines $(R||C_{\text{max}})$

Scheduling unrelated machines

Input:
- $m$ machines.
- $n$ jobs with processing requirements $p_{ij}$ for scheduling job $j$ on machine $i$.

Goal: Find a schedule of minimum length.
Problem 7: Scheduling unrelated machines

What’s known?

- A 2-approximation algorithm via LP rounding (Lenstra, Shmoys, Tardos 1990)
Problem 7: Scheduling unrelated machines

What’s known?

- A 2-approximation algorithm via LP rounding (Lenstra, Shmoys, Tardos 1990)
- A local search algorithm with bound 1.94 if $p_{ij} \in \{p_j, \infty\}$ for all $i, j$, but not poly time. (Svensson STOC 2011).
Problem 7: Scheduling unrelated machines

What’s known?

- A 2-approximation algorithm via LP rounding (Lenstra, Shmoys, Tardos 1990)
- A local search algorithm with bound 1.94 if $p_{ij} \in \{p_j, \infty\}$ for all $i, j$, but not poly time. (Svensson STOC 2011).
- No $\alpha$-approximation algorithm with $\alpha < 3/2$ is possible unless $P = NP$ (Lenstra, Shmoys, Tardos 1990).
Problem 7: Scheduling unrelated machines

What’s known?

- A 2-approximation algorithm via LP rounding (Lenstra, Shmoys, Tardos 1990)
- A local search algorithm with bound 1.94 if \( p_{ij} \in \{p_j, \infty\} \) for all \( i, j \), but not poly time. (Svensson STOC 2011).
- No \( \alpha \)-approximation algorithm with \( \alpha < 3/2 \) is possible unless \( P = NP \) (Lenstra, Shmoys, Tardos 1990).

The problem:
Give an \( \alpha \)-approximation algorithm for \( 3/2 \leq \alpha < 2 \), or show that this is not possible.
What’s known now?

- No progress, except for restricted cases.
Problem 6: Generalized Steiner tree

Generalized Steiner tree (aka Steiner forest)

Input:
- Undirected graph $G = (V, E)$.
- Nonnegative edge costs $c_e \geq 0$ for all $e \in E$.
- $k$ source-sink pairs $s_1-t_1$, $s_2-t_2$, \ldots, $s_k-t_k$.

Goal: Find edges $F$ of minimum cost so that for each $i$, $s_i$ and $t_i$ are connected in $(V, F)$.
Problem 6: Generalized Steiner tree

What’s known?


No $\alpha$-approximation algorithm possible for Steiner tree for $\alpha < \frac{96}{95} \approx 1.01$ unless P = NP (Chlebík, Chlebíková 2008).

Find an $\alpha$-approximation algorithm for the generalized Steiner tree problem for constant $\alpha < 2$. 
Problem 6: Generalized Steiner tree

What’s known?


- If $s_i = s$ for all $i$, have the Steiner tree problem; then a 1.39-approximation algorithm known using LP rounding (Byrka, Grandoni, Rothvoß, Sanità STOC 2010).

No $\alpha$-approximation algorithm possible for Steiner tree for $\alpha < 96.01$ unless $P=NP$ (Chlebík, Chlebíková 2008).
Problem 6: Generalized Steiner tree

What’s known?

- If $s_i = s$ for all $i$, have the Steiner tree problem; then a 1.39-approximation algorithm known using LP rounding (Byrka, Grandoni, Rothvoß, Sanità STOC 2010).
- No $\alpha$-approximation algorithm possible for Steiner tree for $\alpha < \frac{96}{95} \approx 1.01$ unless P = NP (Chlebík, Chlebíková 2008)
Problem 6: Generalized Steiner tree

What’s known?


- If $s_i = s$ for all $i$, have the Steiner tree problem; then a 1.39-approximation algorithm known using LP rounding (Byrka, Grandoni, Rothvoß, Sanità STOC 2010).

- No $\alpha$-approximation algorithm possible for Steiner tree for $\alpha < \frac{96}{95} \approx 1.01$ unless $P = NP$ (Chlebík, Chlebíková 2008).

The problem

Find an $\alpha$-approximation algorithm for the generalized Steiner tree problem for constant $\alpha < 2$. 
What’s known now?

- No progress.
A belief about approximation algorithms

A proof of approximation guarantee $\alpha$ for algorithm $A$ is always a proof about a polytime-computable relaxation $R$:

$$R \leq OPT \leq A \leq \alpha R.$$
A belief about approximation algorithms

A proof of approximation guarantee $\alpha$ for algorithm $A$ is always a proof about a polytime-computable relaxation $R$:

$$R \leq \text{OPT} \leq A \leq \alpha R.$$  

The aim of this paper is to look for one or two guiding principles [in analyzing heuristics], and in particular principles relating the analysis of heuristics to such traditional preoccupations of operations researchers as linear programming and branch and bound... We assume problem can be formulated as a linear integer program, and the essential step is to relate the heuristic solution to a dual feasible solution of the given integer problem.

Wolsey, Heuristic analysis, linear programming and branch and bound (1980)
Problem 5: Capacitated facility location

Capacitated facility location

Input:
- A set $F$ of facilities; each $i \in F$ has facility cost $f_i \geq 0$.
- A set $D$ of clients.
- A metric $c_{ij}$ on locations $i, j \in F \cup D$.
- A capacity $U$ on each facility.

Goal: Find $S \subset F$ and assignment $\sigma : D \to S$ such that $|\sigma^{-1}(i)| \leq U$ for all $i \in S$ that minimizes $\sum_{i \in S} f_i + \sum_{j \in D} c_{\sigma(j), j}$. 

David P. Williamson (Cornell University)
Problem 5: Capacitated facility location

What’s known?
A local search algorithm: Let $S$ be a set of currently open facilities. As long as it improves the overall cost,

- **Add**: $S \leftarrow S \cup \{i\}$ for $i \notin S$;
- **Drop**: $S \leftarrow S - \{i\}$ for $i \in S$; or
- **Swap**: $S \leftarrow S \cup \{i\} - \{j\}$ for $i \notin S$, $j \in S$.

Can show this gives an $(\alpha + \epsilon)$-approximation algorithm for
- $\alpha = 8$ (Koropolu, Plaxton, Rajaraman 2000)
- $\alpha = 6$ (Chudak, W 2005)
- $\alpha = 3$ (Aggarwal et al. 2010)
Problem 5: Capacitated facility location

What’s known?
A local search algorithm: Let $S$ be a set of currently open facilities. As long as it improves the overall cost,

- **Add**: $S \leftarrow S \cup \{i\}$ for $i \notin S$;
- **Drop**: $S \leftarrow S - \{i\}$ for $i \in S$; or
- **Swap**: $S \leftarrow S \cup \{i\} - \{j\}$ for $i \notin S$, $j \in S$.

Can show this gives an $(\alpha + \epsilon)$-approximation algorithm for

- $\alpha = 8$ (Koropolu, Plaxton, Rajaraman 2000)
- $\alpha = 6$ (Chudak, W 2005)
- $\alpha = 3$ (Aggarwal et al. 2010)
Problem 5: Capacitated facility location

The problem:
Is there a polytime-computable relaxation $R$ of the problem within a constant factor of the optimal?
Problem 5: Capacitated facility location

The problem:
Is there a polytime-computable relaxation \( R \) of the problem within a constant factor of the optimal?

Or, what’s the approximate min-max relaxation?

\[
R \leq \text{OPT} \leq A \leq \alpha R.
\]
What’s known now?

Problem 4: Survivable network design

Survivable network design

Input:
- An undirected graph $G = (V, E)$
- Costs $c_e \geq 0$ for all $e \in E$
- Integer connectivity requirements $r_{ij}$ for all $i, j \in V$

Goal: Find a minimum-cost set of edges $F$ so that for all $i, j \in V$, there are at least $r_{ij}$ edge-disjoint paths between $i$ and $j$ in $(V, F)$. 
Problem 4: Survivable network design

Survivable network design

Input:
- An undirected graph $G = (V, E)$
- Costs $c_e \geq 0$ for all $e \in E$
- Integer connectivity requirements $r_{ij}$ for all $i, j \in V$

Goal: Find a minimum-cost set of edges $F$ so that for all $i, j \in V$, there are at least $r_{ij}$ edge-disjoint paths between $i$ and $j$ in $(V, F)$. 
Problem 4: Survivable network design

What’s known?

- A primal-dual $2H_R$-approximation algorithm (Goemans, Goldberg, Plotkin, Shmoys, Tardos, W ’94), where $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$ and $R = \max_{i,j} r_{ij}$.
- An LP rounding 2-approximation algorithm (Jain 2001)
Problem 4: Survivable network design

What’s known?

- A primal-dual $2H_R$-approximation algorithm (Goemans, Goldberg, Plotkin, Shmoys, Tardos, W ’94), where $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$ and $R = \max_{i,j} r_{ij}$.
- An LP rounding 2-approximation algorithm (Jain 2001)

\[
\begin{align*}
\text{minimize} & \quad \sum_{e \in E} c_e x_e \\
\text{subject to} & \quad \sum_{e \in \delta(S)} x_e \geq \max_{i \in S, j \notin S} r_{ij}, \quad \forall S \subset V, \\
& \quad 0 \leq x_e \leq 1, \quad \forall e \in E.
\end{align*}
\]

Theorem (Jain 2001)

For any basic feasible solution $x^*$ of the LP relaxation, there exists some edge $e \in E$ such that $x^*_e > 1/2$. 
Problem 4: Survivable network design

The problem:
Is there a lightweight 2-approximation algorithm? E.g. a primal-dual algorithm?
What’s known now?

- (Feldmann, Könemann, Pashkovich, Sanità ISAAC 2016) A multiplicative-weights update based $(2 + \epsilon)$-approximation algorithm.
- Still not quite what we were looking for.
Problem 3: Bin packing

Input: $b_i$ pieces of size $s_i$, $0 < s_i < 1$, for $i = 1, \ldots, m$

Goal: Find a packing of pieces into bins of size 1 that minimizes the total number of bins used
Problem 3: Bin packing

What’s known?
An LP-rounding algorithm that uses $OPT + O(log^2 OPT)$ bins (Karmarkar, Karp 1982)
Problem 3: Bin packing

What’s known?
An LP-rounding algorithm that uses $\text{OPT} + O(\log^2 \text{OPT})$ bins (Karmarkar, Karp 1982)

Enumerate all $N$ possible ways of packing a bin. $j$th configuration uses $a_{ij}$ pieces of size $i$.

\[
\begin{align*}
\text{minimize} & \quad \sum_{j=1}^{N} x_j \\
\text{subject to} & \quad \sum_{j=1}^{N} a_{ij} x_j \geq b_i, \quad i = 1, \ldots, m, \\
& \quad x_j \text{ integer}, \quad j = 1, \ldots, N.
\end{align*}
\]
Problem 3: Bin packing

The problem:
Find a polytime algorithm that uses at most $\text{OPT} + c$ bins for some constant $c$. 

Note that there are instances known for which $\text{OPT} > \text{LP} + 1$, but currently no known instances for which $\text{OPT} > \text{LP} + 2$. Possibly $\text{OPT} \leq \lceil \text{LP} \rceil + 1$. 

David P. Williamson (Cornell University)
Problem 3: Bin packing

The problem:
Find a polytime algorithm that uses at most \( \text{OPT} + c \) bins for some constant \( c \).

Note that there are instances known for which

\[
\text{OPT} > \text{LP} + 1,
\]

but currently no known instances for which

\[
\text{OPT} > \text{LP} + 2.
\]

Possibly

\[
\text{OPT} \leq \lceil \text{LP} \rceil + 1.
\]
What’s known now?

- (Rothvoss FOCS 2013) \( \text{OPT} + O(\log \text{OPT} \log \log \text{OPT}) \) bins.
- (Hoberg, Rothvoss SODA 2017) \( \text{OPT} + O(\log \text{OPT}) \) bins.
Problems 1 and 2: the traveling salesman problem

Traveling salesman problem

Input:
- Set of cities $V$
- Travel costs $c_{ij}$ such that $c_{ij} \leq c_{ik} + c_{kj}$ for all $i, j, k \in V$

Goal: Find a minimum-cost tour of all the cities
Problem 2: the asymmetric case ($c_{ij} \neq c_{ji}$)

**What’s known?**

- An $O(\log n)$-approximation algorithm (Frieze, Galbiati, Maffioli 1982)
- An LP rounding $O(\log n / \log \log n)$-approximation algorithm (Asadpour, Goemans, Madry, Oveis Gharan, Saberi 2010)
- Can’t approximate better than $\frac{117}{116} \approx 1.008$ unless P = NP (Papadimitriou, Vempala 2006)
Problems 1 and 2: the traveling salesman problem

\[
\begin{align*}
\text{minimize} & \quad \sum_{i,j \in V} c_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} \quad i \in V, \\
        & \quad \sum_{i \in S, j \notin S} x_{ij} \geq 1 \quad \forall S \subset V \\
        & \quad x_{ij} \geq 0 \quad \forall i, j \in V.
\end{align*}
\]

No instance known for which the integrality gap is worse than 2 (Charikar, Goemans, Karloff 2006)
Problems 1 and 2: the traveling salesman problem

minimize \[ \sum_{i,j \in V} c_{ij} x_{ij} \]
subject to \[ \sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji} \quad i \in V, \]
\[ \sum_{i \in S, j \notin S} x_{ij} \geq 1 \quad \forall S \subset V \]
\[ x_{ij} \geq 0 \quad \forall i, j \in V. \]

No instance known for which the integrality gap is worse than 2 (Charikar, Goemans, Karloff 2006)

The problem:
Find an \( \alpha \)-approximation algorithm for \( \alpha \) constant for the asymmetric case.
Problems 1 and 2: the traveling salesman problem

Problem 1: the symmetric case $c_{ij} = c_{ji}$ for all $i, j \in V$

What’s known?

- A $\frac{3}{2}$-approximation algorithm (Christofides 1976)
- Can’t approximate better than $\frac{220}{219} \approx 1.004$ unless $P = NP$ (Papadimitriou, Vempala 2006)
Problems 1 and 2: the traveling salesman problem

Problem 1: the symmetric case $c_{ij} = c_{ji}$ for all $i, j \in V$

What’s known?

- A $\frac{3}{2}$-approximation algorithm (Christofides 1976)
- Can’t approximate better than $\frac{220}{219} \approx 1.004$ unless $P = NP$ (Papadimitriou, Vempala 2006)

Graphical case: given graph $G = (V, E)$, $c_{ij}$ is shortest-length path between $i$ and $j$ in $G$

- Oveis Gharan, Saberi, Singh (December 2010): $\frac{3}{2} - 10^{-12}$
- Mömke, Svensson (April 2011): $\frac{14(\sqrt{2}-1)}{12\sqrt{2}-13} \approx 1.461$
- Mucha (August 2011): $\frac{35}{24} \approx 1.458$
- Sebő, Vygen (2012): 1.4
Problems 1 and 2: the traveling salesman problem

minimize \[ \sum_{i,j \in V: i < j} c_{ij} x_{ij} \]

subject to \[ \sum_{j \in V: i < j} x_{ij} + \sum_{j \in V: i > j} x_{ji} = 2 \quad i \in V \]
\[ \sum_{i \in S, j \notin S \text{ or } i \notin S, j \in S} x_{ij} \geq 2 \quad \forall S \subset V \]
\[ x_{ij} \geq 0 \quad \forall i, j \in V, i < j.\]

Integrality gap at most \( \frac{3}{2} \) (Wolsey 1980). No instance known with gap worse than \( \frac{4}{3} \).
Problems 1 and 2: traveling salesman

$k$
Problems 1 and 2: the traveling salesman problem

The problem:
Find an $\alpha$-approximation algorithm for constant $\alpha < \frac{3}{2}$. 
A hard, simple case

Suppose LP solution is a fractional 2-matching (all $x_{ij} \in \{0, 1/2, 1\}$). Can we do better than $3/2$ whenever this is the case?
A hard, simple case

Suppose LP solution is a fractional 2-matching (all $x_{ij} \in \{0, 1/2, 1\}$). Can we do better than 3/2 whenever this is the case?

Conjecture (Schalekamp, W, van Zuylen 2011):
Such instances give the worst-case integrality gap.
What’s known now?

- Symmetric TSP: No progress
- Graph TSP: No progress
Problems that didn’t make the cut:

- Directed Steiner tree
- LP-based Steiner tree (then Byrka et al. came out)
- Feedback arc set in directed graphs (improve $O(\log n \log \log n)$)
- $P|prec|C_{\text{max}}$ (then Svensson came out)
- Edge coloring multigraphs (+1 result)
- Flow shop, job shop scheduling
- Minimum-cost $k$-connected subgraph
- Subset feedback vertex set (better than 8)
No open problem of the form “this problem has an $\alpha$-approximation algorithm for constant $\alpha$, find a PTAS.”
One reason I like the field

Once I start thinking “maybe all the really interesting stuff has been done,” someone proves me wrong.
One reason I like the field

Once I start thinking “maybe all the really interesting stuff has been done,” someone proves me wrong.

Just in the last year or so

- The new asymmetric TSP result
- $\frac{3}{2} + \frac{1}{34}$ s-t path TSP result
- Hoberg-Rothvoss $+O(\log \text{OPT})$ bin-packing
One reason I like the field

Once I start thinking “maybe all the really interesting stuff has been done,” someone proves me wrong.

Just in the last year or so

- The new asymmetric TSP result
- $\frac{3}{2} + \frac{1}{34}$ s-t path TSP result
- Hoberg-Rothvoss $+ O(\log \text{OPT})$ bin-packing

And perhaps your work will be next!