CS 599 B1: Math for TCS

Lecture 24: CSPs, Proofs, and LP Hierarchies

Mark Bun

Resources

- Ryan O’Donnell’s lectures on CSPs, LP Hierarchies, and Proof Systems
- Monique Laurent’s A Comparison of the Sherali-Adams, Lovasz-Schrijver, and Lasserre Relaxations for 0-1 Programming
- Fleming, Kothari, and Pitassi, Semialgebraic Proofs and Efficient Algorithm Design

What are we doing?

This unit so far:

- Linear and semidefinite programming
- Using LP / SDP relaxations to approximately solve combinatorial optimization problems
- Paradigm: Exact OPT $\leq$ LP OPT
  Round LP solution back to an integral solution

Where we're going:

- What general class(es) of problems can we solve like this?
- To what extent are approximation algorithms based on LP/SDP relaxations automatable?
- Can we certify the *non-existence* of good solutions to combinatorial optimization problems?

A Class of Problems: CSPs

CSP = Constraint Satisfaction Problem

Domain D e.g., $D = \{true, false\}$, $\{-1, +1\}$, $\{1, 2, \dots, q\}$

Variables $V = x_1, \dots, x_n$

Constraint Set $\Psi = \{\psi \mid \psi : D^* \to \{0,1\}\}$

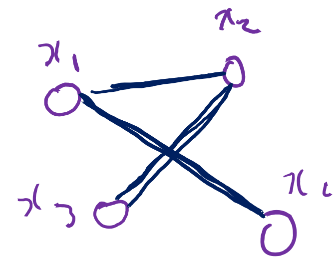
$\psi : D^3 \to \{0,1\}$

$\psi(\underbrace{x_7, x_9, x_2}_{V_i})$

CSP Instance: A list of constraints, each of the form $C_i = (\psi_i, V_i)$ where $\psi_i \in \Psi$ and $V_i \subseteq V$.

Goal: Assign variables to maximize number of satisfied $\psi_i(V_i)$

(Not-equal, (x_1, x_2)), (NE, (x_2, x_3)), (NE, (x_1, x_4))



Ex: MAX-CUT

$D = \{0,1\}$ $\Psi = \{ \text{Not-equal} : \{0,1\}^2 \to \{0,1\} \}$

A Class of Problems: CSPs

CSP = Constraint Satisfaction Problem

Domain D e.g., $D = \{true, false\}$, $\{-1, +1\}$, $\{1, 2, \dots, q\}$

Variables $V = x_1, \dots, x_n$

Constraint Set $\Psi = \{\psi \mid \psi: D^* \to \{0, 1\}\}$

CSP Instance: A list of constraints, each of the form $C_i = (\psi_i, V_i)$
where $\psi_i \in \Psi$ and $V_i \subseteq V$.

Goal: Assign variables to maximize number of satisfied $\psi_i(V_i)$

Ex: MAX-3-COL

$D = \{red, green, blue\}$ $\Psi = \{NE : \{red, green, blue\}^2 \to \{0,1\}\}$

A Class of Problems: CSPs

CSP = Constraint Satisfaction Problem

Domain D e.g., $D = \{true, false\}$, $\{-1, +1\}$, $\{1, 2, \dots, q\}$

Variables $V = x_1, \dots, x_n$

Constraint Set $\Psi = \{\psi \mid \psi: D^* \to \{0, 1\}\}$

CSP Instance: A list of constraints, each of the form $C_i = (\psi_i, V_i)$ where $\psi_i \in \Psi$ and $V_i \subseteq V$.

Goal: Assign variables to maximize number of satisfied $\psi_i(V_i)$

$f(x_1, \dots, x_n) = \underbrace{(x_1 \vee \bar{x}_2 \vee x_7)}_{3} \wedge (x_3 \vee x_4 \vee \bar{x}_5)$ $\wedge \ldots$

Ex: MAX-EXACT-3-SAT

$D = \{0, 1\}$ $\Psi = \{ OR(\cdot, \cdot, \cdot), OR(\bar{\cdot}, \cdot, \cdot), OR(\cdot, \bar{\cdot}, \cdot), \dots, OR(\bar{\cdot}, \bar{\cdot}, \bar{\cdot}) \}$

A Class of Problems: CSPs

CSP = Constraint Satisfaction Problem

Domain D e.g., $D = \{true, false\}$, $\{-1, +1\}$, $\{1, 2, \dots, q\}$

Variables $V = x_1, \dots, x_n$

Constraint Set $\Psi = \{\psi \mid \psi : D^* \to \{0, 1\}\}$

CSP Instance: A list of constraints, each of the form $C_i = (\psi_i, V_i)$ where $\psi_i \in \Psi$ and $V_i \subseteq V$.

Goal: Assign variables to maximize number of satisfied $\psi_i(V_i)$

Ex: MAX-3-SAT

$D = \{0, 1\}$

$$\Psi = \left\{ \begin{array}{l} OR(\cdot, \cdot, \cdot),\ OR(\bar{\cdot}, \cdot, \cdot),\ \dots \\ OR(\cdot, \cdot),\ OR(\bar{\cdot}, \cdot),\ \dots \\ OR(\cdot),\ OR(\bar{\cdot}) \end{array} \right\}$$

CSP Algorithmic Problems

$\in [0,1]$

For a CSP $L = (C_1, \ldots, C_m)$, define $OPT(L)$ = max fraction of satisfiable constraints

(ψ_i, V_i)

Val_L (assignment to V) = # satisfied constraints in L

<u>Satisfiability:</u> Given L, are *all* constraints satisfiable? ($OPT(L) = 1$?)

E.g. decision version of SAT, 3-COL, etc.

<u>Optimization:</u> Given L, find an assignment that approximately maximizes the number of satisfied constraints (⇒ certify that $OPT(L) \geq \beta$)

<u>Certification:</u> Given L, provide a "proof" that $OPT(L) \leq \beta$

CSP Satisfiability

Sometimes it's easy:

2-SAT, Horn-SAT, LIN-EQ-MOD2, bipartiteness testing $\in$ P

decision version of MAX-CUT

Sometimes it's hard:

3-SAT, 3-COL, … are NP-complete

Schaefer '78: When $D = \{0, 1\}$, every CSP is either in P or NP-complete

Dichotomy Conjecture [Fejer-Vardi '93]: Every CSP is either in P or NP-complete

Proved (independently) by Bulatov and Zhuk in 2017

CSP Optimization & Certification

Optimization (α, β)-approximation algorithm: Given L with $OPT(L) \geq \beta$, find an assignment with value α. $\alpha \leq \beta$

Exercise 10.2 gave a $\left(\frac{3}{4}\beta, \beta\right)$-approximation to MAX-3SAT for every β

Goemans-Williamson is a $(0.878\beta, \beta)$-approximation to MAX-CUT for every β

Certification (α, β)-certifier: Given L with $OPT(L) \leq \alpha$, output a proof that $OPT(L) \leq \beta$. $\beta \geq \alpha$

Exercise 10.2 gave a $\left(\frac{3}{4}\beta, \beta\right)$-certifier for every β IP $\rightsquigarrow$ LP

Goemans-Williamson is *also* a $(0.878\beta, \beta)$-certifier

In general, an (α, β)-approximation is also an (α, β)-certifier

What are we doing?

This unit so far:

- Linear and semidefinite programming
- Using LP / SDP relaxations to approximately solve combinatorial optimization problems
- Paradigm: Exact OPT $\leq$ LP OPT

  Round LP solution back to an integral solution

Where we're going:

- What general class(es) of problems can we solve like this? CSPs
- To what extent are approximation algorithms based on LP/SDP relaxations automatable?
- Can we certify the *non-existence* of good solutions to combinatorial optimization problems?

Linear Programming as a Proof System

Recall: Bipartite matching LP

$$\max \textstyle\sum_{(\ell,r)\in E} x_{\ell,r}$$

$$\text{s.t.} \quad \textstyle\sum_{r\sim\ell} x_{\ell,r} \le 1 \quad \forall \ell \in L$$

$$\textstyle\sum_{\ell\sim r} x_{\ell,r} \le 1 \quad \forall r \in R$$

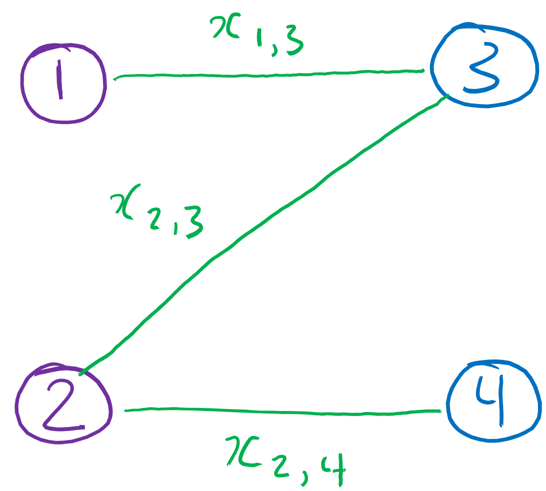
$$x_{\ell,r} \ge 0 \quad \forall (\ell,r) \in E$$

$x_{\ell,r} \in \{0,1\}$ IP

Linear Programming as a Proof System

Recall: Bipartite matching LP

$$\begin{aligned} \max \quad & x_{1,3} + x_{2,3} + x_{2,4} \\ \text{s.t.} \quad & x_{1,3} \le 1 \\ & x_{2,3} + x_{2,4} \le 1 \\ & x_{1,3} + x_{2,3} \le 1 \\ & x_{2,4} \le 1 \\ & x_{1,3}, x_{2,3}, x_{2,4} \ge 0 \end{aligned}$$



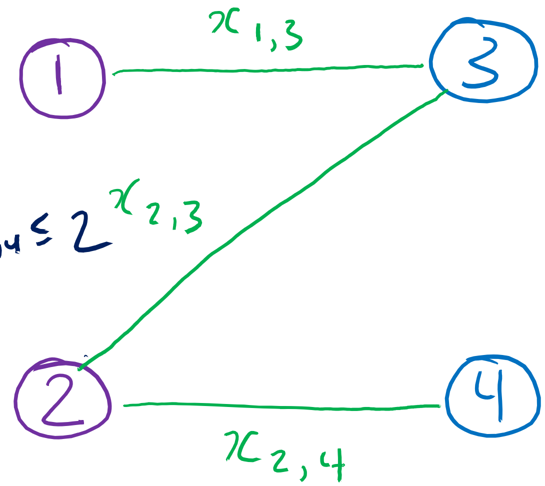
(β, β)-approximation algorithm: Solve the LP and produce an integral solution, e.g., $x_{1,3} = 1, x_{2,3} = 0, x_{2,4} = 1$

Linear Programming as a Proof System

Recall: Bipartite matching LP

$$
\begin{array}{lll}
\max & x_{1,3} + x_{2,3} + x_{2,4} & \\
\text{s.t.} & x_{1,3} \le 1 & y_1 = 1 \\
& x_{2,3} + x_{2,4} \le 1 & y_2 = 1 \\
& x_{1,3} + x_{2,3} \le 1 & y_3 = 0 \\
& x_{2,4} \le 1 & y_4 = 0 \\
& x_{1,3}, x_{2,3}, x_{2,4} \ge 0 &
\end{array}
$$

$\Rightarrow x_{1,3} + x_{2,3} + x_{2,4} \le 2$



(β, β)-certifier: Find a combination of constraints that certifies an upper bound on $x_{1,3} + x_{2,3} + x_{2,4}$ a.k.a. solve the dual LP a.k.a. find a min vertex cover

Linear Programming as a Proof System

More abstractly:

indeterminates

$$
\begin{aligned}
\max\quad & x_{1,3} + x_{2,3} + x_{2,4} \\
\text{s.t.}\quad & x_{1,3} \le 1 & \qquad y_1 = 1 \\
& x_{2,3} + x_{2,4} \le 1 & \qquad y_2 = 1 \\
& x_{1,3} + x_{2,3} \le 1 & \qquad y_3 = 0 \\
& x_{2,4} \le 1 & \qquad y_4 = 0 \\
& x_{1,3}, x_{2,3}, x_{2,4} \ge 0 &
\end{aligned}
$$

axioms

Derivation / inference rules

Linear combinations of previous "lines of proof"

$x_{1,3} \le 1$, $x_{2,3} + x_{2,4} \le 1$

$\vdash x_{1,3} + x_{2,3} + x_{2,4} \le 2$

<u>Goal:</u> Prove that $x_{1,3} + x_{2,3} + x_{2,4} \le 2$

Linear Programming as a Proof System

Indeterminates: $x_1, \ldots, x_n$

Axioms: $\langle a, x \rangle \leq b$

Proof lines: Linear inequalities

Inference rules: Can derive non-negative linear combinations of previous proof lines

Goal: Prove that $\boxed{\langle c, x \rangle \leq \beta}$

Properties:

- Soundness: Any statement proved is true
- Completeness: Any true statement can be proved (LP duality theorem)
- Automatizable: Can efficiently find a proof of any provable statement

LPs as Proof Systems for Integer Programs

Indeterminates: $x_1, \ldots, x_n$

Axioms: $\langle a, x \rangle \leq b$ that are implied by IP axioms

Proof lines: Linear inequalities

Inference rules: Can derive non-negative linear combinations of previous proof lines

Goal: Prove that $\langle c, x \rangle \leq \beta$

OPT ≤ LinOPT

Properties:

- Soundness: Any statement proved is true
- ~~Completeness: Any true statement can be proved (LP duality theorem)~~
- Automatable: Can efficiently find a proof of any provable statement

Toward Completeness

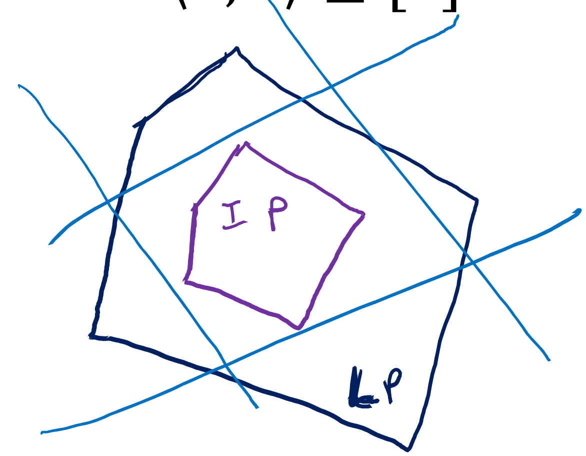
Idea: Add extra (linear) axioms and derivation rules that are consistent with integer solutions

<u>Cutting Planes</u>

New derivation rule: If a is integral, then $\langle a, x\rangle \le b \;\; \rightarrow \;\; \langle a, x\rangle \le \lfloor b \rfloor$

$$x_1 + 2x_2 + x_3 \le 7.5$$

$$\vdash \; x_1 + 2x_2 + x_3 \le 7$$



Toward Completeness

Idea: Add extra (linear) axioms and derivation rules that are consistent with integer solutions

<u>Lovasz-Schrijver and Sherali-Adams</u>

New axioms: (True) inequalities involving low-degree polynomials of indeterminates

1) Extend: Add polynomial constraints implied by integrality
2) Linearize: Replace monomials with placeholder variables to get an LP
3) Project: Round solution over placeholder variables

Example: MAX-SAT

OPT(f) ≤ $\frac{3}{4}$

$$f(x) = (\cancel{x_1} \vee \cancel{x_2} \vee \overline{x_3}) \wedge (\cancel{x_1} \vee x_3) \wedge (\cancel{x_1} \vee \overline{x_2}) \wedge \overline{x_1}$$

$x_2 = 0$ $\quad x_1 = 0$

$$\begin{aligned}
\max \quad & z_1 + z_2 + z_3 + z_4 \\
\text{s.t.} \quad & x_1 + x_2 + (1 - x_3) \geq z_1 \\
& x_1 + x_3 \geq z_2 \\
& x_1 + (1 - x_2) \geq z_3 \\
& (1 - x_1) \geq z_4 \\
& \cancel{x_i, z_i \in \{0,1\}} \quad 0 \leq x_i, z_i \leq 1
\end{aligned}$$

$x_1 = \frac{1}{2}$ $\quad z_1 = \frac{3}{2}$

$x_2 = \frac{1}{2}$ $\quad z_2 = 1$

$x_3 = \frac{1}{2}$ $\quad z_3 = 1$

$z_4 = \frac{1}{2}$

Level 2 Sherali-Adams

$$f(x) = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge \overline{x_1}$$

$$\begin{aligned} \max \quad & x_4 + x_5 + x_6 + x_7 \\ \text{s.t.} \quad & x_1 + x_2 + (1 - x_3) \geq x_4 \\ & x_1 + x_3 \geq x_5 \\ & x_1 + (1 - x_2) \geq x_6 \\ & (1 - x_1) \geq x_7 \\ & 0 \leq x_i \leq 1 \end{aligned}$$

<u>Extend via new degree-2 constraints:</u>

$x_1 x_2 \geq 0, \ldots$
$x_1(1 - x_2) \geq 0, \ldots$
$(1 - x_1)(1 - x_2) \geq 0, \ldots$

$x_1\big(x_1 + x_2 + (1 - x_3)\big) \geq x_1 x_1, \ldots$
$(1 - x_1)\big(x_1 + x_2 + (1 - x_3)\big) \geq (1 - x_1)x_1, \ldots$

Level 2 Sherali-Adams

$$f(x) = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee x_3) \wedge (x_1 \vee \overline{x_2}) \wedge \overline{x_1}$$

$$\begin{aligned} \max \quad & x_4 + x_5 + x_6 + x_7 \\ \text{s.t.} \quad & x_1 + x_2 + (1 - x_3) \geq x_4 \\ & x_1 + x_3 \geq x_5 \\ & x_1 + (1 - x_2) \geq x_6 \\ & (1 - x_1) \geq x_7 \\ & 0 \leq x_i \leq 1 \end{aligned}$$

<u>Linearize by replacing</u> $x_i x_j$ with $y_{\{i,j\}}$, x_i with $y_{\{i\}}$:

$x_1 x_2 \geq 0, \ldots$ $\rightarrow$ $y_{\{1,2\}} \geq 0$

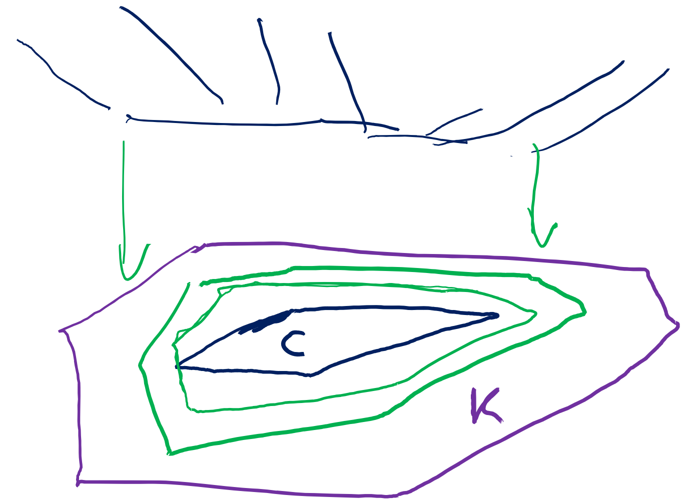
$x_1(1 - x_2) \geq 0, \ldots$ $\rightarrow$ $y_{\{1\}} - y_{\{1,2\}} \geq 0$

$(1 - x_1)(1 - x_2) \geq 0, \ldots$ $\rightarrow$ $1 - y_{\{1\}} - y_{\{2\}} + y_{\{1,2\}} \geq 0$

$x_1\big(x_1 + x_2 + (1 - x_3)\big) \geq x_1 x_1, \ldots$

$(1 - x_1)\big(x_1 + x_2 + (1 - x_3)\big) \geq (1 - x_1)x_1, \ldots$

Level d Sherali-Adams



Given: $K = \{\langle a_1, x\rangle \geq 0, \dots, \langle a_m, x\rangle \geq 0\}$

1. Extend: Include every constraint of the form

$$\underbrace{\langle a_i, x\rangle \cdot \prod_{j\in S} x_j \prod_{k\in T}(1 - x_k)}_{\text{degree} \leq d} \geq 0$$

2. Linearize:

 a. Replace every appearance of x_j^c with x_j

 b. Replace every appearance of $\prod_{y\in S} x_j$ with y_S

The resulting relaxation is called $SA_d(K)$

Facts about Sherali-Adams

- Each SA_d is a tightening of SA_{d-1} : It preserves all integral solutions, while removing some fractional ones

  Sound proof system

- SA_{n+1} recovers the original integral feasible set

  SA_{n+1} is complete

- Each SA_d involves roughly $m \cdot n^d$ constraints and can be optimized over in $poly(m \cdot n^d)$ time.

What’s it good for?

- Can get a
- For a given size LP relaxation, Sherali-Adams is essentially optimal [Chan-Lee-Raghavendra-Steurer13]