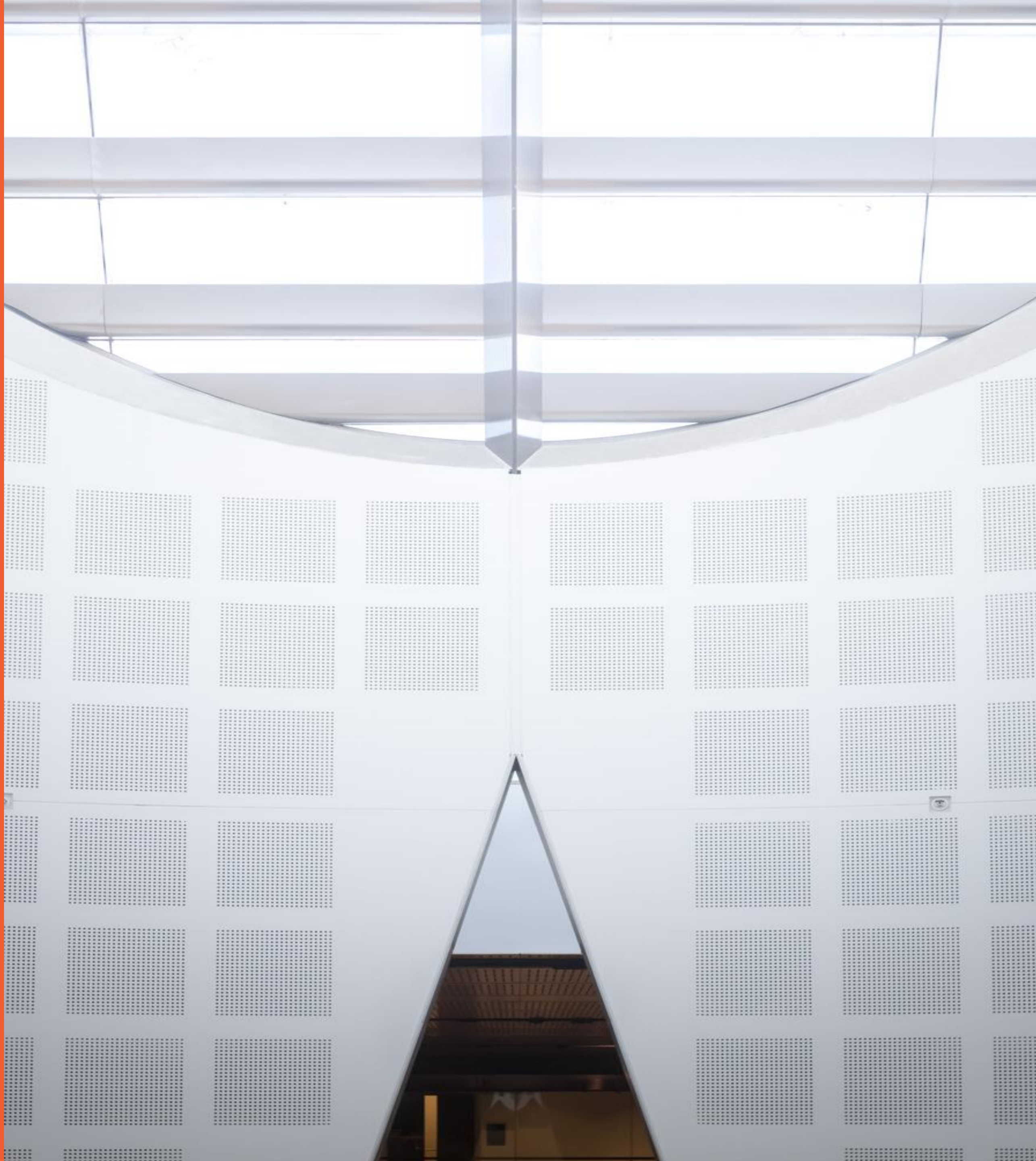
COMPx270: Randomised and Advanced Algorithms
Lecture 10: Linear Programming and Randomised Rounding

Clément Canonne

School of Computer Science

THE UNIVERSITY OF SYDNEY



Some housekeeping

- A2 being marked, solutions online
- A3 (after Simple Extension) due next Wednesday
- Don't forget the "participation" assignment (Oct 18)
- Sample exam is out
- Feedback welcome: https://forms.office.com/r/DymMcfn47n
- Final exam on Tues, Nov 12 (9am)

Assignment 2: what was this about?

Consistent Hashing:

David R. Karger, Eric Lehman, Frank Thomson Leighton, Rina Panigrahy, Matthew S. Levine, Daniel Lewin. *Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web.* STOC 1997: 654-663

Abstract

We describe a family of caching protocols for distrib-uted networks that can be used to decrease or eliminate the occurrence of hot spots in the network. Our protocols are particularly designed for use with very large networks such as the Internet, where delays caused by hot spots can be severe, and where it is not feasible for every server to have complete information about the current state of the entire network. The protocols are easy to implement using existing network protocols such as TCP/IP, and require very little overhead. The protocols work with local control, make efficient use of existing resources, and scale gracefully as the network grows.

Our caching protocols are based on a special kind of hashing that we call *consistent hashing*. Roughly speaking, a consistent hash function is one which changes minimally as the range of the function changes. Through the development of good consistent hash functions, we are able to develop caching protocols which do not require users to have a current or even consistent view of the network. We believe that consistent hash functions may eventually prove to be useful in other applications such as distributed name servers and/or quorum systems.

Assignment 2: what was this about?

Consistent Hashing:

David R. Karger, Eric Lehman, Frank Thomson Leighton, Rina Panigrahy, Matthew S. Levine, Daniel Lewin. *Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web.* STOC 1997: 654-663

Abstract

We describe a family of caching protocols for distrib-uted networks that can be used to decrease or eliminate the occurrence of hot spots in the network. Our protocols are particularly designed for use with very large networks such as the Internet, where delays caused by hot spots can be severe, and where it is not feasible for every server to have complete information about the current state of the entire network. The protocols are easy to implement using existing network protocols such as TCP/IP, and require very little overhead. The protocols work with local control, make efficient use of existing resources, and scale gracefully as the network grows.

Our caching protocols are based on a special kind of hashing that we call *consistent hashing*. Roughly speaking, a consistent hash function is one which changes minimally as the range of the function changes. Through the development of good consistent hash functions, we are able to develop caching protocols which do not require users to have a current or even consistent view of the network. We believe that consistent hash functions may eventually prove to be useful in other applications such as distributed name servers and/or quorum systems.

Assignment 2: what was this about?

Consistent Hashing:

David R. Karger, Eric Lehman, Frank Thomson Leighton, Rina Panigrahy, Matthew S. Levine, Daniel Lewin. *Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web.* STOC 1997: 654-663

This week: Linear Programming, and Randomised Rounding

> Maximize a linear function subject to linear inequality constraints on variables $x_1, \ldots, x_n$ of interest.

Linear Programming

Maximize a linear function subject to linear inequality constraints on variables $x_1, \ldots, x_n$ of interest.

or minimise

$$\text{maximise} \sum_{i=1}^{n} c_i x_i$$

subject to

$$\sum_{i=1}^{n} A_{ji} x_i \le b_j, \qquad 1 \le j \le m$$

$$x_i \ge 0, \qquad 1 \le i \le n$$

Linear Programming

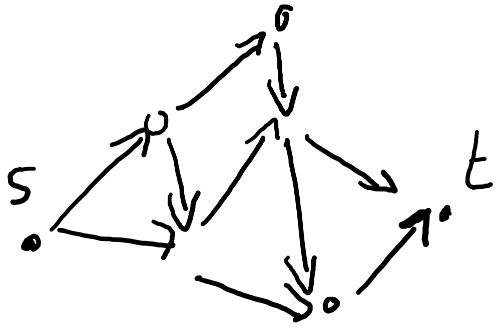
$$\text{maximise} \sum_{i=1}^{n} c_i x_i$$

subject to

$$\sum_{i=1}^{n} A_{ji} x_i \le b_j, \qquad 1 \le j \le m$$

$$x_i \ge 0, \qquad 1 \le i \le n$$

Example: Max Flow!



$G = (V, E)$ directed

$s, t \in V$

capacities $\{c_e\}_{e \in E}$

$$\max \sum_{u : (s,u) \in E} f_{su}$$

s.t. $\quad 0 \le f_e \le c_e \qquad \forall e \in E$

$$\sum_{v : (u,v) \in E} f_{uv} = \sum_{v : (v,u) \in E} f_{vu} \qquad \forall u \in V \setminus \{s,t\}$$

Linear Programming

$$\text{maximise} \sum_{i=1}^{n} c_i x_i$$

subject to

$$\sum_{i=1}^{n} A_{ji} x_i \leq b_j, \qquad 1 \leq j \leq m$$

$$x_i \geq 0, \qquad 1 \leq i \leq n$$

① Everything (in P) can be an LP

② LPs can be solved in poly-time (in size of the LP)

③ LPs ______ efficiently in practice

④ We're not going to see that.

Linear Programming

$$\text{maximise } \sum_{i=1}^{n} c_i x_i$$

subject to

$$\sum_{i=1}^{n} A_{ji} x_i \le b_j, \qquad 1 \le j \le m$$

$$x_i \ge 0, \qquad 1 \le i \le n$$

Use them to solve problems either exactly or approximately.

Task T

Instance I for T } get solution S

(valid)

$$\alpha \cdot \textsc{opt}(I) \le \underbrace{\textsc{value}(S)}_{\text{my value}} \le \textsc{opt}(I)$$

$\alpha \in (0, 1]$

optimal value for I

$\mathbb{E}[\textsc{value}(S)]$

Integer Linear Programming

maximize $\sum_{i=1}^{n} c_i x_i$

s.t. $Ax \leq b$

$x_i \in \{0, 1, \ldots, k\} \quad \forall i$

(typically $x_i \in \{0, 1\}$)

More power!

Don't know how to solve them (efficiently)

LP:

$x_i \geq 0$

$x_i = 1$

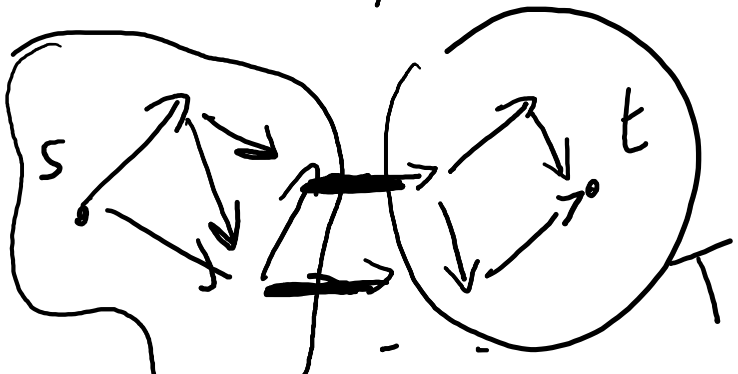
ILP:

$x_i^2 \stackrel{?}{=} x_i$

Integer Linear Programming: st-Min-CUT

Directed $G=(V,E)$, costs $\{c_e\}_e$

$s,t \in V$



S

Minimize value of the st-cut

$$\sum_{\substack{e=(u,v)\\ u\in S\\ v\in T=S^c}} c_e$$

maximize $-\sum_{e\in E} c_e x_e$

variables for edges

s.t.

$$y_s = 0$$

$$y_t = 1$$

$$y_v \le y_u + x_{uv} \quad \forall (u,v)\in E$$

$$x_e, y_v \in \{0,1\} \quad \forall e\in E,\ \forall v\in V.$$

Integer Linear Programming: st-Min-CUT

$$
\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \leq y_u + x_e, \qquad \forall e = (u,v) \in E \\
& x_e, y_v \in \{0,1\} \qquad \forall e \in E, v \in V
\end{aligned}
$$

LP Relaxation: st-Min-CUT

$$\begin{aligned}
\text{maximise} \quad & -\sum_{e\in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\
& x_e, y_v \in \{0,1\} \qquad \forall e \in E, v \in V
\end{aligned}$$

$$\begin{aligned}
\text{maximise} \quad & -\sum_{e\in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\
& x_e, y_v \in [0,1] \qquad \forall e \in E, v \in V
\end{aligned}$$

LP Relaxation: st-Min-CUT

$$
\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, && \forall e = (u,v) \in E \\
& x_e, y_v \in \{0,1\} && \forall e \in E, v \in V
\end{aligned}
$$

$$
\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, && \forall e = (u,v) \in E \\
& x_e, y_v \in [0,1] && \forall e \in E, v \in V
\end{aligned}
$$

Fact 45.1. *Let* OPT_{ILP} *be the optimal value of a solution to an ILP (maximisation problem), and* OPT_{LP} *be the optimal value of a solution to its LP relaxation. Then*

$$\text{OPT}_{\text{ILP}} \le \text{OPT}_{\text{LP}}\,.$$

(For a minimisation problem, the inequality is reversed.)

Problem $\xrightarrow{(\Leftarrow)}$ ILP $\xrightarrow{\text{relaxation}}$ LP $\xrightarrow{\text{"rounding" of solution}}$

use LP constraints to prove solution "not too bad" compared to OPT_{LP}

LP Relaxation: st-Min-CUT

- Solve LP relaxation optimally
  $\to$ get x^*, y^* st $\text{VALUE}(y^*) = OPT_{LP}$
- Round?
  If $y_u^* > 0.5$, $y_u \leftarrow 1$

$$
\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\
& x_e, y_v \in \{0,1\} \qquad \forall e \in E, v \in V
\end{aligned}
$$

$$
\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\
& x_e, y_v \in [0,1] \qquad \forall e \in E, v \in V
\end{aligned}
$$

$y_v^* - y_u^* \le x_e^* \quad \forall e$

LP Rounding!

$$\begin{aligned} \text{maximise} \quad & -\sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \\ & y_s = 0 \\ & y_t = 1 \\ & y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\ & x_e, y_v \in \{0,1\} \qquad \forall e \in E, v \in V \end{aligned}$$

$$\begin{aligned} \text{maximise} \quad & -\sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \\ & y_s = 0 \\ & y_t = 1 \\ & y_v \le y_u + x_e, \qquad \forall e = (u,v) \in E \\ & x_e, y_v \in [0,1] \qquad \forall e \in E, v \in V \end{aligned}$$

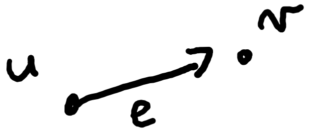
LP Rounding: st-Min-CUT

$$\begin{aligned}\text{maximise} \quad & -\sum_{e\in E} c_e x_e \\ \text{subject to} \quad & \\ & y_s = 0 \\ & y_t = 1 \\ & y_v \le y_u + x_e, \qquad \forall e=(u,v)\in E \\ & x_e, y_v \in \{0,1\} \qquad \forall e\in E, v\in V\end{aligned}$$

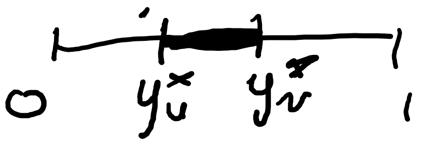
$$\begin{aligned}\text{maximise} \quad & -\sum_{e\in E} c_e x_e \\ \text{subject to} \quad & \\ & y_s = 0 \\ & y_t = 1 \\ & y_v \le y_u + x_e, \qquad \forall e=(u,v)\in E \\ & x_e, y_v \in [0,1] \qquad \forall e\in E, v\in V\end{aligned}$$

Pick τ in $(0,1)$ uniformly at random.
for all $v \in V$ **do**
  Set $y_v = 1$ if $y_v^* > \tau$, 0 otherwise
return y.

$e = (u,v)$ is cut iff $y_u^* = 0$, $y_v^* = 1$



$\iff y_u^* \le \tau < y_v^* \rightarrow \Pr[\tau \in [y_u^*, y_v^*)] = y_v^* - y_u^*$
$\le x_e^*$



LP Rounding: st-Min-CUT

$$\mathbb{E}[\text{value}(S)] = \sum_{e \in E} c_e \, \mathbb{E}[\mathbb{1}_{e \text{ is cut}}]$$

$$= \sum_{e \in E} c_e \Pr[e \text{ is cut}]$$

$$= \sum_{(u,v) \in E} c_{uv} \Pr_{\tau}[y^*_u \leq \tau < y^*_v]$$

$$\underset{\text{Previous slide}}{\leq} \sum_{(u,v) \in E} c_{uv} \, x^*_{uv}.$$

$$= -\text{OPT}_{LP}$$

$$\leq -\text{OPT}_{ILP} = \text{value min cut}.$$

$$\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \leq y_u + x_e, \quad \forall e = (u,v) \in E \\
& x_e, y_v \in \{0,1\} \quad \forall e \in E, v \in V
\end{aligned}$$

$$\begin{aligned}
\text{maximise} \quad & -\sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \\
& y_s = 0 \\
& y_t = 1 \\
& y_v \leq y_u + x_e, \quad \forall e = (u,v) \in E \\
& x_e, y_v \in [0,1] \quad \forall e \in E, v \in V
\end{aligned}$$

LPand ILP in practice

- https://au.mathworks.com/help/optim/ug/linprog.html
- https://au.mathworks.com/help/optim/ug/intlinprog.html
- https://reference.wolfram.com/language/ref/LinearProgramming.html
- https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linprog.html
- https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.milp.html
- [...]

ILP+LP Relaxation+Rounding: Max-SAT

$x_1, \dots, x_n \in \{\text{true}, \text{false}\}$

$\text{true} \to 1$, $\text{false} \to 0$

$\phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$ where

$C_j = x_{i_1} \vee \overline{x_{i_2}} \vee \dots \vee x_{i_{\ell_j}}$ $\ell_j = |C_j|$

* **SAT**: is there a way to set $x_1, \dots, x_n$ st $\phi \equiv$ true?
  (SAT is NP-Complete)

* **Max-SAT**: how many C_j's can I make true?
  (NP-Hard)
  ↳ want to approximate

Note: don't know how to do better than .878-approx

~~ILP+LP Relaxation+Rounding:~~ Max-SAT

Theorem 47. *The "obvious" randomised algorithm which sets each variable x_i independently and uniformly at random gives, in expectation, a $\frac{1}{2}$-approximation for* MAX-SAT.

~~ILP+LP Relaxation+Rounding:~~ Max-SAT

Theorem 47. *The "obvious" randomised algorithm which sets each variable x_i independently and uniformly at random gives, in expectation, a $\frac{1}{2}$-approximation for* MAX-SAT.

Goal: $\max \underbrace{\sum_{j=1}^{m} \mathbb{1}_{C_j \text{ satisfied}}}_{\text{value}}$

Proof Fix $1 \leq j \leq m$, consider C_j.

$$C_j = x_{i_1} \vee x_{i_2} \vee \bar{x}_{i_3} \vee \_ \vee \bar{x}_{i_{25}} \vee x_{i_{21}}$$

$$\ell_j = |C_j|$$

$$\Pr[C_j \text{ not sat.}] = \frac{1}{2^{\ell_j}}$$

$$\mathbb{E}[\text{value}] = \sum_{j=1}^{m} \Pr[C_j \text{ sat.}] = \sum_{j=1}^{m} \left(1 - \frac{1}{2^{\ell_j}}\right) \geq \frac{m}{2}$$

$\ell_j \geq 1$

$\square$

~~ILP+LP Relaxation+Rounding:~~ Max-SAT

Theorem 47. *The "obvious" randomised algorithm which sets each variable x_i independently and uniformly at random gives, in expectation, a $\frac{1}{2}$-approximation for* Max-SAT.

Pathetic, <u>but</u> good if $\ell_j \gg 1$ for all j.

even ≥ 2 is not bad.

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

is x_i = true ?

$$y_i \in \{0,1\} \qquad \forall 1 \leq i \leq n$$

$$z_j \in \{0,1\} \qquad \forall 1 \leq j \leq m$$

is C_j satisfied ?

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i: x_i \in C_j} y_i + \sum_{i: \neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$y_i \in \{0,1\} \qquad \forall 1 \leq i \leq n$$

$$z_j \in \{0,1\} \qquad \forall 1 \leq j \leq m$$

- $\Rightarrow$ Given an assignment to MAX-SAT $x_1, \ldots, x_n$
  get assign^t — $y_1, \ldots, y_n$, $z_1, \ldots, z_m$ to ILP
  w/ same value

- $\Leftarrow$ (the other way around)
  given optimal solution to the ILP,
  get sol. to MAX-SAT
  w/ same value

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$y_i \in \{0,1\} \qquad \forall 1 \leq i \leq n$$

$$z_j \in \{0,1\} \qquad \forall 1 \leq j \leq m$$

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

ILP+LP Relaxation+Rounding: Max-SAT

How to round? Given optimal solution y^*, z^* to LP,

y_i^* is <u>not</u> in $\{0,1\}$

but in $[0,1]$ $\to$ can be seen as a proba!

 Set $x_i := \begin{cases} 0 & \text{w/p } 1 - y_i^* \\ 1 & \text{w/p } y_i^* \end{cases}$

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i: x_i \in C_j} y_i + \sum_{i: \neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$y_i \in \{0,1\} \qquad \forall 1 \leq i \leq n$$

$$z_j \in \{0,1\} \qquad \forall 1 \leq j \leq m$$

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i: x_i \in C_j} y_i + \sum_{i: \neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise } \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$y_i \in \{0,1\} \qquad \forall 1 \leq i \leq n$$

$$z_j \in \{0,1\} \qquad \forall 1 \leq j \leq m$$

$$\text{maximise } \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

Input: Instance $\phi = (C_1, \ldots, C_m)$ of MAX-SAT on n variables

1: Solve the LP relaxation (Fig. 16), getting solution (y^*, z^*).

2: **for all** $1 \leq i \leq n$ **do**

3: Set $x_i = 1$ with probability y_i^*, independently of others.

4: **return** x.

ILP+LP Relaxation+Rounding: Max-SAT

Theorem 48. *The randomised rounding given in Algorithm 20 gives, in expectation, a* $\underbrace{(1 - \frac{1}{e})}_{\approx 0.632}$*-approximation for* MAX-SAT.

ILP+LP Relaxation+Rounding: Max-SAT

maximise $\sum_{j=1}^{m} z_j$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1-y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

$$\mathbb{E}[\text{value}_\phi(x)] = \sum_{j=1}^{m} \Pr[C_j \text{ is satisfied}]$$

$$= \sum_{j=1}^{m} (1 - \underbrace{\Pr[C_j \text{ not sat.}]}_{?})$$

Fix j.

$$C_j = \bigvee_{i \in S} x_i \quad \bigvee_{i \in T} \bar{x}_i$$

$$\Pr[C_j \text{ not sat.}] = \prod_{i \in S} (1 - y_i^*) \cdot \prod_{i \in T} y_i^*$$

$$\leq \left(\frac{1}{\ell_j} \left(\sum_{i \in S} (1-y_i^*) + \sum_{i \in T} y_i^* \right) \right)^{\ell_j} \qquad (\text{AM-GM})$$

$$= \left(\frac{1}{\ell_j} \left(\ell_j - \underbrace{\left(\sum_{i \in S} y_i^* + \sum_{i \in T} (1-y_i^*) \right)}_{\geq z_j} \right) \right)^{\ell_j} \leq \left(1 - \frac{z_j}{\ell_j} \right)^{\ell_j}$$

AM-GM.

$$\sqrt{ab} \leq \frac{a+b}{2}$$

$$\sqrt[k]{a_1 \cdots a_k} \leq \frac{a_1 + \cdots + a_k}{k}$$

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1-y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

$$\forall j \quad \Pr[C_j \text{ not sat}] \leq \left(1 - \frac{z_j^*}{\ell_j}\right)^{\ell_j}$$

$$\mathbb{E}[\text{value}_\phi(x)] \geq \sum_{j=1}^{m} \left(1 - \left(1 - \frac{z_j^*}{\ell_j}\right)^{\ell_j}\right) \ldots$$

Want $\sum_{j=1}^{m} z_j^*$

$$\overset{(A)}{\geq} \sum_{j=1}^{m} \left(1 - \left(1 - \frac{1}{\ell_j}\right)^{\ell_j}\right) z_j^*$$

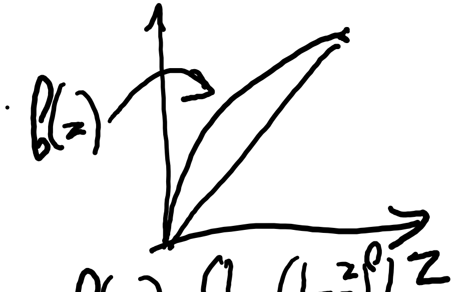
$$\geq \inf_{\ell \geq 1} \left(1 - \left(1 - \frac{1}{\ell}\right)^{\ell}\right) \cdot \sum_{j=1}^{m} z_j^*$$

$$= \left(1 - \frac{1}{e}\right) \sum_{j=1}^{m} z_j^* = \left(1 - \frac{1}{e}\right) \mathrm{OPT}_{LP}$$

$$\geq \left(1 - \frac{1}{e}\right) \mathrm{OPT}_{ILP} = \left(1 - \frac{1}{e}\right) \mathrm{OPT}_\phi \qquad \square$$

(A) "Concavity"

for any $\ell \geq 1$



$$f(z) = \left(1 - \left(1 - \frac{z}{\ell}\right)^{\ell}\right)$$

ILP+LP Relaxation+Rounding: Max-SAT

$$\text{maximise} \sum_{j=1}^{m} z_j$$

subject to

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\neg x_i \in C_j} (1 - y_i) \geq z_j \qquad \forall 1 \leq j \leq m$$

$$0 \leq y_i \leq 1 \qquad \forall 1 \leq i \leq n$$

$$0 \leq z_j \leq 1 \qquad \forall 1 \leq j \leq m$$

Good for small clauses!

Max-SAT: Can we do better?

Max-SAT: Can we do better?

Theorem. The “best-of-two” approach which runs both the naïve randomised algorithm and the randomised rounding gives, in expectation, a 3/4-approximation for Max-SAT.

Max-SAT: Can we do better?

Theorem. The "best-of-two" approach which runs both the naïve randomised algorithm and the randomised rounding gives, in expectation, a 3/4-approximation for Max-SAT.

$$\mathbb{E}[\max(val_\phi(x), val_\phi(x'))] \geq \frac{1}{2}\mathbb{E}[val_\phi(x) + val_\phi(x')]$$

$$\geq \frac{1}{2}\sum_{j=1}^{m}\left[\left(1-\frac{1}{2^{\ell_j}}\right) + \left(1-\left(1-\frac{1}{\ell_j}\right)^{\ell_j}\right)z_j^*\right]$$

$$\geq \frac{1}{2}\sum_{j=1}^{m}\underbrace{\left[\left(1-\frac{1}{2^{\ell_j}}\right) + \left(1-\left(1-\frac{1}{\ell_j}\right)^{\ell_j}\right)\right]}_{g(\ell_j)} z_j^*$$

$$\geq \frac{3}{4}\sum_{j=1}^{m} z_j^*$$

$$\geq \frac{3}{4} OPT_{ILP}$$

$\square$

$\frac{1}{2}(a+b) \leq \max(a,b)$ $(\leq a+b)$

Claim.

$g(\ell) \geq \frac{3}{2}$

$\forall \ell \in \{1\} \cup [2,\infty)$

Recap

LPs are powerful (but not enough)
We know how to solve them.

ILPs are more powerful.
We don't know how to solve them.

$$\text{Problem} \Longleftrightarrow \text{ILP} \xrightarrow[\text{relax}]{} \text{LP} \xrightarrow[\text{solve}]{} x^* \xrightarrow[\text{round}]{} x \xrightarrow[\text{relate to } OPT_{LP}]{} \text{value}(x) \text{ good}$$

(using what the LP does)