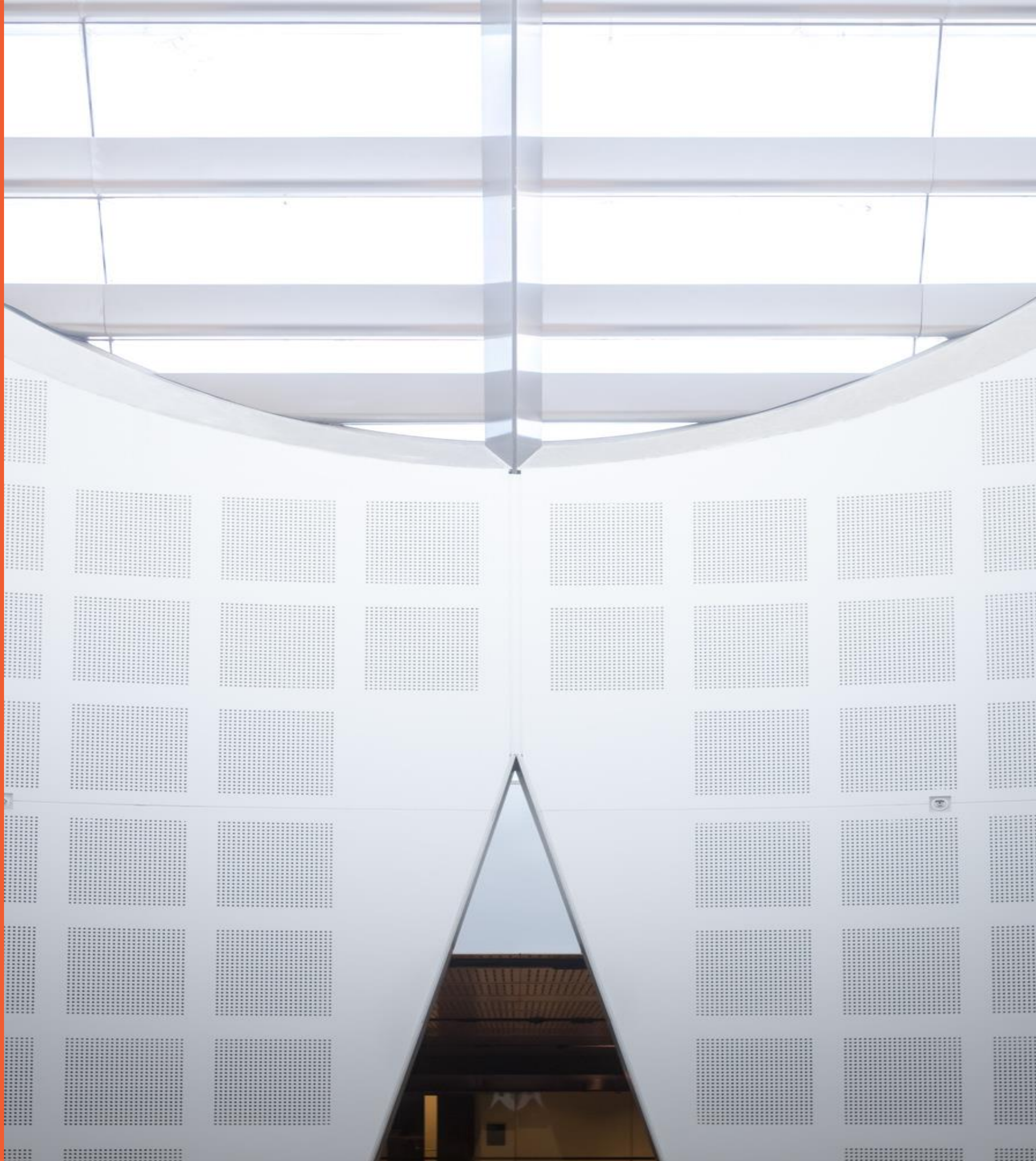
COMPx270: Randomised and Advanced Algorithms
Lecture 10: Linear Programming and Randomised Rounding

Clément Canonne
School of Computer Science

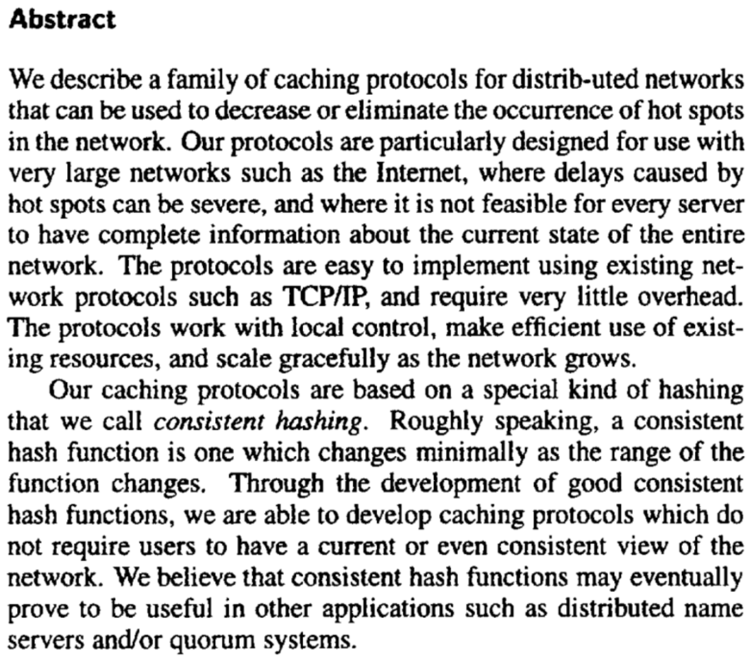
THE UNIVERSITY OF SYDNEY



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

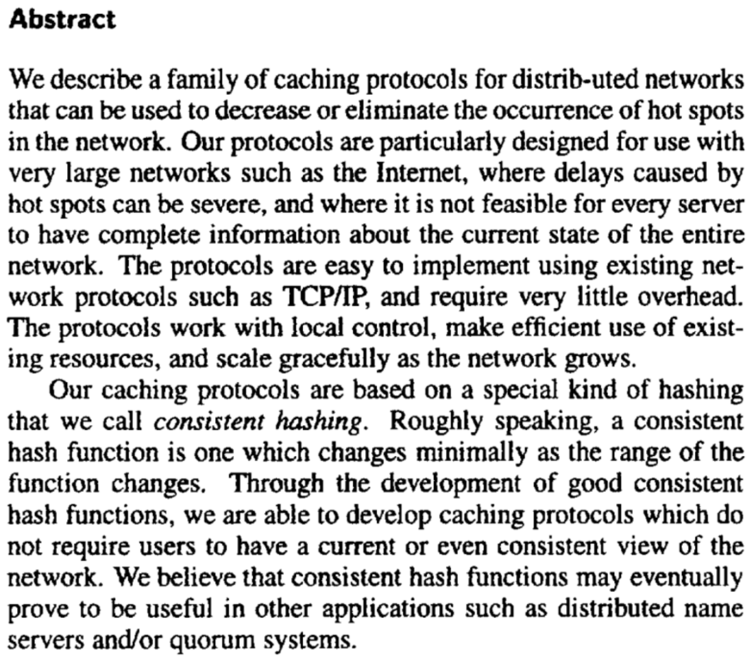
We describe a family of caching protocols for distrib-uted networks that can be used to decrease or eliminate the occurrence of hot spots in the network. Our protocols are particularly designed for use with very large networks such as the Internet, where delays caused by hot spots can be severe, and 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

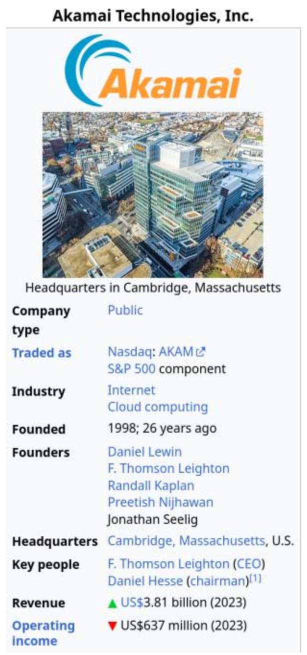
We describe a family of caching protocols for distrib-uted networks that can be used to decrease or eliminate the occurrence of hot spots in the network. Our protocols are particularly designed for use with very large networks such as the Internet, where delays caused by hot spots can be severe, and 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

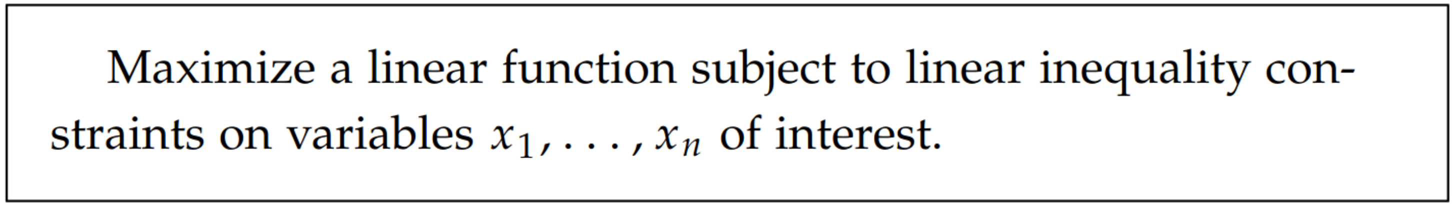


Akamai Technologies, Inc.

Headquarters in Cambridge, Massachusetts

Company type	Public
Traded as	Nasdaq: AKAM S&P 500 component
Industry	Internet Cloud computing
Founded	1998; 26 years ago
Founders	Daniel Lewin F. Thomson Leighton Randall Kaplan Preetish Nijhawan Jonathan Seelig
Headquarters	Cambridge, Massachusetts, U.S.
Key people	F. Thomson Leighton (CEO) Daniel Hesse (chairman)[1]
Revenue	▲ US$3.81 billion (2023)
Operating income	▼ US$637 million (2023)

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 $x_1, \dots, x_n$ of interest.

$$\begin{aligned}
&\text{maximise } \sum_{i=1}^{n} c_i x_i \\
&\text{subject to} \\
&\qquad \sum_{i=1}^{n} A_{ji} x_i \le b_j, \qquad 1 \le j \le m \\
&\qquad\qquad x_i \ge 0, \qquad 1 \le i \le n
\end{aligned}$$

Linear Programming

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

subject to

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

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

Example: Max Flow!

Linear Programming

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

subject to

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

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

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.

$$\alpha \cdot \textsc{opt}(I) \le \textsc{value}(S) \le \textsc{opt}(I)$$

Integer Linear Programming

Integer Linear Programming: st-Min-CUT

Integer Linear Programming: st-Min-CUT

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

LP Relaxation: st-Min-CUT

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

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

LP Relaxation: st-Min-CUT

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

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

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.)

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 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

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

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

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

LP Rounding: st-Min-CUT

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

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

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

~~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.

~~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

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

subject to

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

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

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

ILP+LP Relaxation+Rounding: Max-SAT

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

subject to

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

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

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

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

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

subject to

$$\begin{aligned} \sum_{i:x_i\in C_j} y_i + \sum_{i:\neg x_i\in C_j} (1-y_i) &\geq z_j && \forall 1\leq j\leq m \\ y_i &\in \{0,1\} && \forall 1\leq i\leq n \\ z_j &\in \{0,1\} && \forall 1\leq j\leq m \end{aligned}$$

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

subject to

$$\begin{aligned} \sum_{i:x_i\in C_j} y_i + \sum_{i:\neg x_i\in C_j} (1-y_i) &\geq z_j && \forall 1\leq j\leq m \\ 0\leq y_i &\leq 1 && \forall 1\leq i\leq n \\ 0\leq z_j &\leq 1 && \forall 1\leq j\leq m \end{aligned}$$

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* $(1 - \frac{1}{e})$*-approximation for* Max-SAT.

Detour: the AM-GM inequality 🛠️

Fact (AM–GM). For any $a_1, a_2, \ldots, a_k \geq 0$,

$$\sqrt[k]{a_1 a_2 \ldots a_k} \leq \frac{a_1 + a_2 + \cdots + a_k}{k}$$

ILP+LP Relaxation+Rounding: Max-SAT

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

subject to

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

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

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

ILP+LP Relaxation+Rounding: Max-SAT

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

subject to

$$\begin{aligned}
&\sum_{i:x_i\in C_j} y_i + \sum_{i:\neg x_i\in C_j} (1-y_i) \geq z_j && \forall 1\leq j\leq m\\
&0\leq y_i\leq 1 && \forall 1\leq i\leq n\\
&0\leq z_j\leq 1 && \forall 1\leq j\leq m
\end{aligned}$$

ILP+LP Relaxation+Rounding: Max-SAT

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

subject to

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

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

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

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.

Recap

- LPs are powerful (but not enough)
- We know how to solve them efficiently (in theory and practice)
- ILPs are more powerful
- We don't know how to solve them (though we often can in practice)