Aalto University
School of Science

Department of Computer Science

Combinatorics of Efficient Computations

Approximation Algorithms

Lecture 11: Maximum Satisfiability

Joachim Spoerhase

2019

Maximum Satisfiability (Max Sat)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Maximum Satisfiability (MAX SAT)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Find: An assignment of the variables $x_1, \ldots, x_n$ such that the total weight of *satisfied* clauses is maximized.

Maximum Satisfiability (Max Sat)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Find: An assignment of the variables $x_1, \ldots, x_n$ such that the total weight of *satisfied* clauses is maximized.

- Literal: variable or negation of a variable, e.g,. x_1, $\bar{x}_1$

Maximum Satisfiability (Max Sat)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Find: An assignment of the variables $x_1, \ldots, x_n$ such that the total weight of *satisfied* clauses is maximized.

- Literal: variable or negation of a variable, e.g,. x_1, $\bar{x}_1$
- Clause: disjuntion of *literals* – e.g., $x_1 \vee \bar{x}_2 \vee x_3$

Maximum Satisfiability (Max Sat)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Find: An assignment of the variables $x_1, \ldots, x_n$ such that the total weight of *satisfied* clauses is maximized.

- Literal: variable or negation of a variable, e.g,. x_1, $\bar{x}_1$
- Clause: disjuntion of *literals* – e.g., $x_1 \vee \bar{x}_2 \vee x_3$
- Clause Length: number of literals

Maximum Satisfiability ($\textsc{Max Sat}$)

Given: n boolean variables $x_1, \ldots, x_n$, and m clauses $C_1, \ldots, C_m$, where each clause C_j has a weight w_j.

Find: An assignment of the variables $x_1, \ldots, x_n$ such that the total weight of *satisfied* clauses is maximized.

- Literal: variable or negation of a variable, e.g,. x_1, $\bar{x}_1$
- Clause: disjuntion of *literals* – e.g., $x_1 \vee \bar{x}_2 \vee x_3$
- Clause Length: number of literals
- Note: $\textsc{Satisfiability}$ ($\textsc{Sat}$) is NP-complete where one is to decide whether a given propositional formula (in conjunctive normal form) has a satisfying assignment. E.g., $(x_1 \vee \bar{x}_2 \vee x_3) \wedge (x_2 \vee \bar{x}_3 \vee x_4) \wedge (x_1 \vee \bar{x}_4)$

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

Proof.

- Let $Y_j \in \{0, 1\}$ and W be random variables where Y_j is the truth value of C_j and W is the weight of satisfied clauses.

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

Proof.

- Let $Y_j \in \{0, 1\}$ and W be random variables where Y_j is the truth value of C_j and W is the weight of satisfied clauses.

$$E[W] = E\left[\sum_{j=1}^{m} w_j Y_j\right] = \sum_{j=1}^{m} w_j E[Y_j] = \sum_{j=1}^{m} w_j \Pr[C_j \text{ sat.}]$$

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

Proof.

- Let $Y_j \in \{0, 1\}$ and W be random variables where Y_j is the truth value of C_j and W is the weight of satisfied clauses.

$$E[W] = E\left[\sum_{j=1}^{m} w_j Y_j\right] = \sum_{j=1}^{m} w_j E[Y_j] = \sum_{j=1}^{m} w_j \Pr[C_j \text{ sat.}]$$

- Let $l_j :=$ length of C_j. $\Pr[C_j \text{ satisfied}] =$

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

Proof.

- Let $Y_j \in \{0, 1\}$ and W be random variables where Y_j is the truth value of C_j and W is the weight of satisfied clauses.

$$E[W] = E\left[\sum_{j=1}^{m} w_j Y_j\right] = \sum_{j=1}^{m} w_j E[Y_j] = \sum_{j=1}^{m} w_j \mathrm{Pr}[C_j \text{ sat.}]$$

- Let $l_j :=$ length of C_j. $\mathrm{Pr}[C_j \text{ satisfied}] = 1 - (\frac{1}{2})^{l_j} \geq \frac{1}{2}$

A simple randomized algorithm

Thm. 1 Independently setting each variable to 1 (`true`) with probability $\frac{1}{2}$ provides an expected $\frac{1}{2}$-approximation for $\textsc{Max Sat}$.

Proof.

- Let $Y_j \in \{0, 1\}$ and W be random variables where Y_j is the truth value of C_j and W is the weight of satisfied clauses.

$$E[W] = E\left[\sum_{j=1}^{m} w_j Y_j\right] = \sum_{j=1}^{m} w_j E[Y_j] = \sum_{j=1}^{m} w_j \text{Pr}[C_j \text{ sat.}]$$

- Let $l_j :=$ length of C_j. $\text{Pr}[C_j \text{ satisfied}] = 1 - (\frac{1}{2})^{l_j} \geq \frac{1}{2}$
- Thus, $E[W] \geq \frac{1}{2} \sum_{j=1}^{m} w_j \geq \frac{1}{2} \cdot \text{OPT}$ $\square$

Derandomization by Conditional Expectation

Thm. 2 The previous algorithm can be derandomized, i.e., there is a deterministic $\frac{1}{2}$-approximation algorithm for $\textsc{Max Sat}$.

Proof.

- Set x_1 deterministically, but $x_2, \ldots, x_n$ randomly.

Derandomization by Conditional Expectation

Thm. 2 The previous algorithm can be derandomized, i.e., there is a deterministic $\frac{1}{2}$-approximation algorithm for $\textsc{Max Sat}$.

Proof.

- Set x_1 deterministically, but $x_2, \ldots, x_n$ randomly.
- Namely: set $x_1 = 1$ iff $E[W|x_1 = 1] \geq E[W|x_1 = 0]$, where W is the same as in Thm. 1

Derandomization by Conditional Expectation

Thm. 2 The previous algorithm can be derandomized, i.e., there is a deterministic $\frac{1}{2}$-approximation algorithm for $\textsc{Max Sat}$.

Proof.

- Set x_1 deterministically, but $x_2, \ldots, x_n$ randomly.
- Namely: set $x_1 = 1$ iff $E[W | x_1 = 1] \geq E[W | x_1 = 0]$, where W is the same as in Thm. 1

  Note: we can compute $E[W | x_1 = 1]$ and $E[W | x_1 = 0]$ as described in the proof of Thm. 1 (formalized later).

Derandomization by Conditional Expectation

Thm. 2 The previous algorithm can be derandomized, i.e., there is a deterministic $\frac{1}{2}$-approximation algorithm for $\textsc{Max Sat}$.

Proof.

- Set x_1 deterministically, but $x_2, \ldots, x_n$ randomly.
- Namely: set $x_1 = 1$ iff $E[W|x_1 = 1] \geq E[W|x_1 = 0]$, where W is the same as in Thm. 1

  Note: we can compute $E[W|x_1 = 1]$ and $E[W|x_1 = 0]$ as described in the proof of Thm. 1 (formalized later).

- $E[W] = \frac{1}{2} \cdot (E[W|x_1 = 0] + E[W|x_1 = 1])$

Derandomization by Conditional Expectation

Thm. 2 The previous algorithm can be derandomized, i.e., there is a deterministic $\frac{1}{2}$-approximation algorithm for $\textsc{Max Sat}$.

Proof.

- Set x_1 deterministically, but $x_2, \ldots, x_n$ randomly.
- Namely: set $x_1 = 1$ iff $E[W|x_1 = 1] \geq E[W|x_1 = 0]$, where W is the same as in Thm. 1

  Note: we can compute $E[W|x_1 = 1]$ and $E[W|x_1 = 0]$ as described in the proof of Thm. 1 (formalized later).

- $E[W] = \frac{1}{2} \cdot (E[W|x_1 = 0] + E[W|x_1 = 1])$
- $\rightsquigarrow$ for $x_1 = b_1$ chosen in this way, we have: $E[W|x_1 = b_1] \geq E[W] \geq \frac{1}{2} \cdot \mathrm{OPT}$

Derandomization by Conditional Expectation

- (by induction) we have set $x_1, \ldots, x_i$ to $b_1, \ldots, b_i$ so that

$$E[W | x_1 = b_1, \ldots, x_i = b_i] \geq E[W] \geq \frac{1}{2} \cdot \mathsf{OPT}$$

Derandomization by Conditional Expectation

- (by induction) we have set $x_1, \ldots, x_i$ to $b_1, \ldots, b_i$ so that

$$E[W|x_1 = b_1, \ldots, x_i = b_i] \geq E[W] \geq \frac{1}{2} \cdot \mathsf{OPT}$$

- Now (similarly to the base case):

$$\begin{aligned} &E[W|x_1 = b_1, \ldots, x_i = b_i] \\ &\qquad = \tfrac{1}{2}(E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0] \\ &\qquad\qquad + E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1]) \end{aligned}$$

Derandomization by Conditional Expectation

- (by induction) we have set $x_1, \ldots, x_i$ to $b_1, \ldots, b_i$ so that

$$E[W|x_1 = b_1, \ldots, x_i = b_i] \geq E[W] \geq \frac{1}{2} \cdot \mathsf{OPT}$$

- Now (similarly to the base case):

$$\begin{aligned} E[W|x_1 = b_1, \ldots, x_i = b_i] \\ &= \tfrac{1}{2}(E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0] \\ &\qquad + E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1]) \end{aligned}$$

- $\rightsquigarrow$ set $x_{i+1} = 1$ if and only if

$$\begin{aligned} &E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1] \\ &\geq E[W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0] \end{aligned}$$

$$\rightsquigarrow E[W|x_1 = b_1, \ldots, x_i = b_i, x_i = b_{i+1}] \geq \ldots \geq \tfrac{1}{2} \cdot \mathsf{OPT}$$

Derandomization by Conditional Expectation

- Thus, the algorithm can be derandomized if the conditional expectation can be computed efficiently.

Derandomization by Conditional Expectation

- Thus, the algorithm can be derandomized if the conditional expectation can be computed efficiently.

- Consider a partial assignment $x_1 = b_1, \ldots, x_i = b_i$ and a clause C_j.

Derandomization by Conditional Expectation

- Thus, the algorithm can be derandomized if the conditional expectation can be computed efficiently.
- Consider a partial assignment $x_1 = b_1, \ldots, x_i = b_i$ and a clause C_j.
- If C_j is already satisfied, then it contributes w_j to $E[W | x_1 = b_1, \ldots, x_i = b_i]$.

Derandomization by Conditional Expectation

- Thus, the algorithm can be derandomized if the conditional expectation can be computed efficiently.

- Consider a partial assignment $x_1 = b_1, \ldots, x_i = b_i$ and a clause C_j.

- If C_j is already satisfied, then it contributes w_j to $E[W|x_1 = b_1, \ldots, x_i = b_i]$.
- If C_j is not satisfied, and contains k unassigned variables, then it contributes precisely $w_j(1 - (\frac{1}{2})^k)$ to $E[W|x_1 = b_1, \ldots, x_i = b_i]$.

Derandomization by Conditional Expectation

- Thus, the algorithm can be derandomized if the conditional expectation can be computed efficiently.

- Consider a partial assignment $x_1 = b_1, \ldots, x_i = b_i$ and a clause C_j.

- If C_j is already satisfied, then it contributes w_j to $E[W|x_1 = b_1, \ldots, x_i = b_i]$.
- If C_j is not satisfied, and contains k unassigned variables, then it contributes precisely $w_j(1 - (\frac{1}{2})^k)$ to $E[W|x_1 = b_1, \ldots, x_i = b_i]$.

- Note: the conditional expectation is simply the sum of the contributions from each clause. $\square$

Derandomization by Conditional Expectation

Standard procedure with which many randomized algorithms can be derandomized.

Derandomization by Conditional Expectation

Standard procedure with which many randomized algorithms can be derandomized.

Requirement: respective conditional probabilities can be appropriately estimated for each random decision.

Derandomization by Conditional Expectation

Standard procedure with which many randomized algorithms can be derandomized.

Requirement: respective conditional probabilities can be appropriately estimated for each random decision.

The algorithm simply chooses the best option at each step.

Derandomization by Conditional Expectation

Standard procedure with which many randomized algorithms can be derandomized.

Requirement: respective conditional probabilities can be appropriately estimated for each random decision.

The algorithm simply chooses the best option at each step.

Quality of the obtained solution is then at least as high as the expected value.

An ILP

$$
\begin{array}{rll}
\text{maximize} & \displaystyle\sum_{j=1}^{m} w_j z_j & \\
\text{subject to} & \displaystyle\sum_{i \in P_j} y_i + \sum_{i \in N_j} (1 - y_i) \geq z_j, & j = 1, \ldots, m \\
& y_i \in \{0, 1\}, & i = 1, \ldots, n \\
& 0 \leq z_j \leq 1, & j = 1, \ldots, m
\end{array}
$$

where $C_j = \bigvee_{i \in P_j} x_i \vee \bigvee_{i \in N_j} \bar{x}_i$ for each $j = 1, \ldots, m$

Note: $z_j = 1$ when C_j is satisfied, and $z_j = 0$ otherwise.

… and its relaxation

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{m} w_j z_j \\ \text{subject to} \quad & \sum_{i \in P_j} y_i + \sum_{i \in N_j} (1 - y_i) \geq z_j, & j = 1, \ldots, m \\ & 0 \leq y_i \leq 1, & i = 1, \ldots, n \\ & 0 \leq z_j \leq 1, & j = 1, \ldots, m \end{aligned}$$

where $C_j = \bigvee_{i \in P_j} x_i \vee \bigvee_{i \in N_j} \bar{x}_i$ for each $j = 1, \ldots, m$

Note: $z_j = 1$ when C_j is satisfied, and $z_j = 0$ otherwise.

Randomized Rounding

Thm. 3 Let $(\mathbf{y}^*, \mathbf{z}^*)$ be an optimal solution to the LP-relaxation. Independently setting each variable x_i to 1 (`true`) with probability y_i^* provides a $(1 - \frac{1}{e})$-approximation for $\textsc{Max Sat}$.

Randomized Rounding

Thm. 3 Let $(\mathbf{y}^*, \mathbf{z}^*)$ be an optimal solution to the LP-relaxation. Independently setting each variable x_i to 1 (`true`) with probability y_i^* provides a $(1 - \frac{1}{e})$-approximation for $\textsc{Max Sat}$.

Proof.

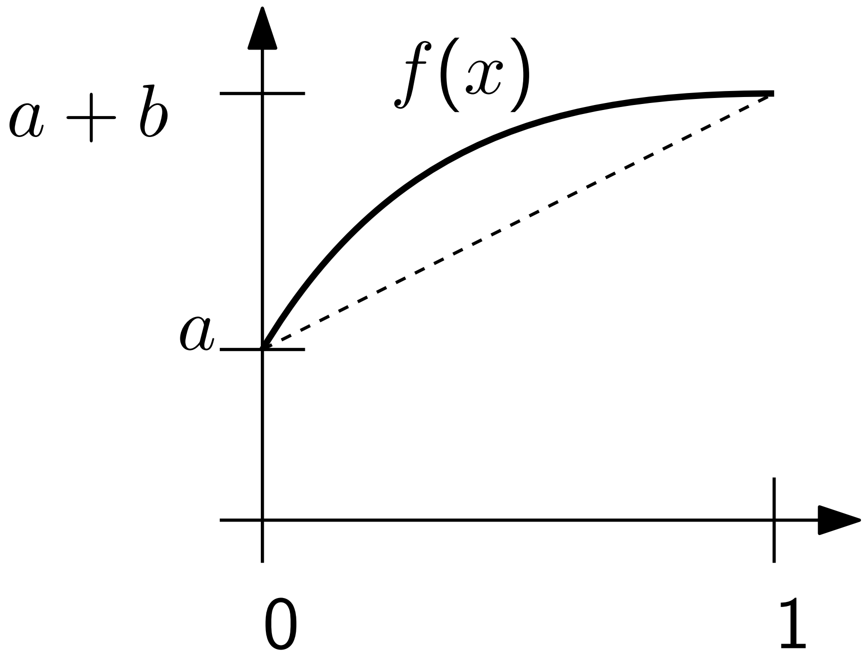
Fact#1: arithmetic-geometric mean inequality (agmi)

For all non-negative numbers $a_1, \ldots, a_k$:

$$\left(\prod_{i=1}^{k} a_i\right)^{1/k} \leq \frac{1}{k}\left(\sum_{i=1}^{k} a_i\right)$$

Randomized Rounding (proof)

Fact#2: Let $f(0) = a$ and $f(1) = a + b$ for a function which is concave on $[0, 1]$ (i.e., $f''(x) \leq 0$ on $[0, 1]$). Then we have $f(x) \geq bx + a$ for $x \in [0, 1]$



Randomized Rounding (proof)

Consider a fixed clause C_j of length l_j. We have:

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

Randomized Rounding (proof)

Consider a fixed clause C_j of length l_j. We have:

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

$$\overset{\text{(agmi).}}{\leq} \left[\frac{1}{l_j} \left(\sum_{i \in P_j} (1 - y_i^*) + \sum_{i \in N_j} y_i^* \right) \right]^{l_j}$$

Randomized Rounding (proof)

Consider a fixed clause C_j of length l_j. We have:

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

$$\overset{\text{(agmi).}}{\leq} \left[\frac{1}{l_j} \left(\sum_{i \in P_j} (1 - y_i^*) + \sum_{i \in N_j} y_i^* \right) \right]^{l_j}$$

$$= \left[1 - \frac{1}{l_j} \left(\sum_{i \in P_j} y_i^* + \sum_{i \in N_j} (1 - y_i^*) \right) \right]^{l_j}$$

Randomized Rounding (proof)

Consider a fixed clause C_j of length l_j. We have:

$$\begin{aligned}
\Pr[C_j \text{ not sat.}] &= \prod_{i \in P_j} (1 - y_i^*) \prod_{i \in N_j} y_i^* \\
&\overset{\text{(agmi).}}{\leq} \left[\frac{1}{l_j} \left(\sum_{i \in P_j} (1 - y_i^*) + \sum_{i \in N_j} y_i^* \right) \right]^{l_j} \\
&= \left[1 - \frac{1}{l_j} \underbrace{\left(\sum_{i \in P_j} y_i^* + \sum_{i \in N_j} (1 - y_i^*) \right)}_{\geq z_j^*} \right]^{l_j} \\
&\overset{\text{LP-Relax.}}{\leq} \left(1 - \frac{z_j^*}{l_j} \right)^{l_j}
\end{aligned}$$

Randomized Rounding (proof)

The function $f(z_j^*) = 1 - \left(1 - \frac{z_j^*}{l_j}\right)^{l_j}$ is concave.
Note: $f(0) = 0$

Randomized Rounding (proof)

The function $f(z_j^*) = 1 - \left(1 - \frac{z_j^*}{l_j}\right)^{l_j}$ is concave.

Note: $f(0) = 0$

Thus

$$\Pr[C_j \text{ sat.}] \geq f(z_j^*)$$

$$\geq \left[1 - \left(1 - \frac{1}{l_j}\right)^{l_j}\right] z_j^*$$

$$\text{Note :} \forall k \in \mathbb{Z}^+, \left(1 - \frac{1}{k}\right)^k > \frac{1}{e}$$

$$\geq \left(1 - \frac{1}{e}\right) z_j^*$$

Randomized Rounding (proof)

Therefore,

$$
\begin{aligned}
E[W] &= \sum_{j=1}^{m} \mathsf{Pr}[C_j \text{ sat.}] \cdot w_j \\
&\geq \left(1 - \frac{1}{e}\right) \sum_{j=1}^{m} w_j z_j^* \\
&\geq \left(1 - \frac{1}{e}\right) \mathsf{OPT}
\end{aligned}
$$

$\square$

Thm. 4 The above algorithm can be derandomized by the method of conditional expectation.

Take the better between the two solutions!

Thm. 5 The better solution among the randomized algorithm (Thm. 1) and the randomized LP-rounding algorithm (Thm. 3), provides a $\frac{3}{4}$-approximation for $\textsc{MaxSat}$

Take the better between the two solutions!

Thm. 5 The better solution among the randomized algorithm (Thm. 1) and the randomized LP-rounding algorithm (Thm. 3), provides a $\frac{3}{4}$-approximation for $\textsc{MaxSat}$

Proof.

We use another probabilistic argument. With probability $\frac{1}{2}$ choose the solution of Thm. 1 otherwise choose Thm. 3.

Take the better between the two solutions!

Thm. 5 The better solution among the randomized algorithm (Thm. 1) and the randomized LP-rounding algorithm (Thm. 3), provides a $\frac{3}{4}$-approximation for $\textsc{MaxSat}$

Proof.

We use another probabilistic argument. With probability $\frac{1}{2}$ choose the solution of Thm. 1 otherwise choose Thm. 3.

The better solution is at least as good as the expectation of the above algorithm.

Take the better between the two solutions!

The probability that clause C_j is satisfied is at least:

$$P = \frac{1}{2}\left[\overbrace{\left(1-\left(1-\frac{1}{l_j}\right)^{l_j}\right)}^{\text{LP-Rounding}} + \overbrace{\left(1-2^{-l_j}\right)}^{\text{rand. Alg.}}\right] z_j^*$$

Take the better between the two solutions!

The probability that clause C_j is satisfied is at least:

$$P = \frac{1}{2}\left[\overbrace{\left(1-\left(1-\frac{1}{l_j}\right)^{l_j}\right)}^{\text{LP-Rounding}} + \overbrace{\left(1-2^{-l_j}\right)}^{\text{rand. Alg.}}\right] z_j^*$$

We claim that this is at least $\frac{3}{4} \cdot z_j^*$. (the rest follows similarly to Thm. 1 and Thm. 3 by the linearity of expectation).

Take the better between the two solutions!

The probability that clause C_j is satisfied is at least:

$$P = \frac{1}{2}\left[\overbrace{\left(1 - \left(1 - \frac{1}{l_j}\right)^{l_j}\right)}^{\text{LP-Rounding}} + \overbrace{\left(1 - 2^{-l_j}\right)}^{\text{rand. Alg.}}\right] z_j^*$$

We claim that this is at least $\frac{3}{4} \cdot z_j^*$. (the rest follows similarly to Thm. 1 and Thm. 3 by the linearity of expectation).

For $l_j = 1, 2$, a simple calculation shows $P = \frac{3}{4} \cdot z_j^*$

Take the better between the two solutions!

The probability that clause C_j is satisfied is at least:

$$P = \frac{1}{2}\left[\overbrace{\left(1-\left(1-\frac{1}{l_j}\right)^{l_j}\right)}^{\text{LP-Rounding}} + \overbrace{\left(1-2^{-l_j}\right)}^{\text{rand. Alg.}}\right] z_j^*$$

We claim that this is at least $\frac{3}{4} \cdot z_j^*$. (the rest follows similarly to Thm. 1 and Thm. 3 by the linearity of expectation).

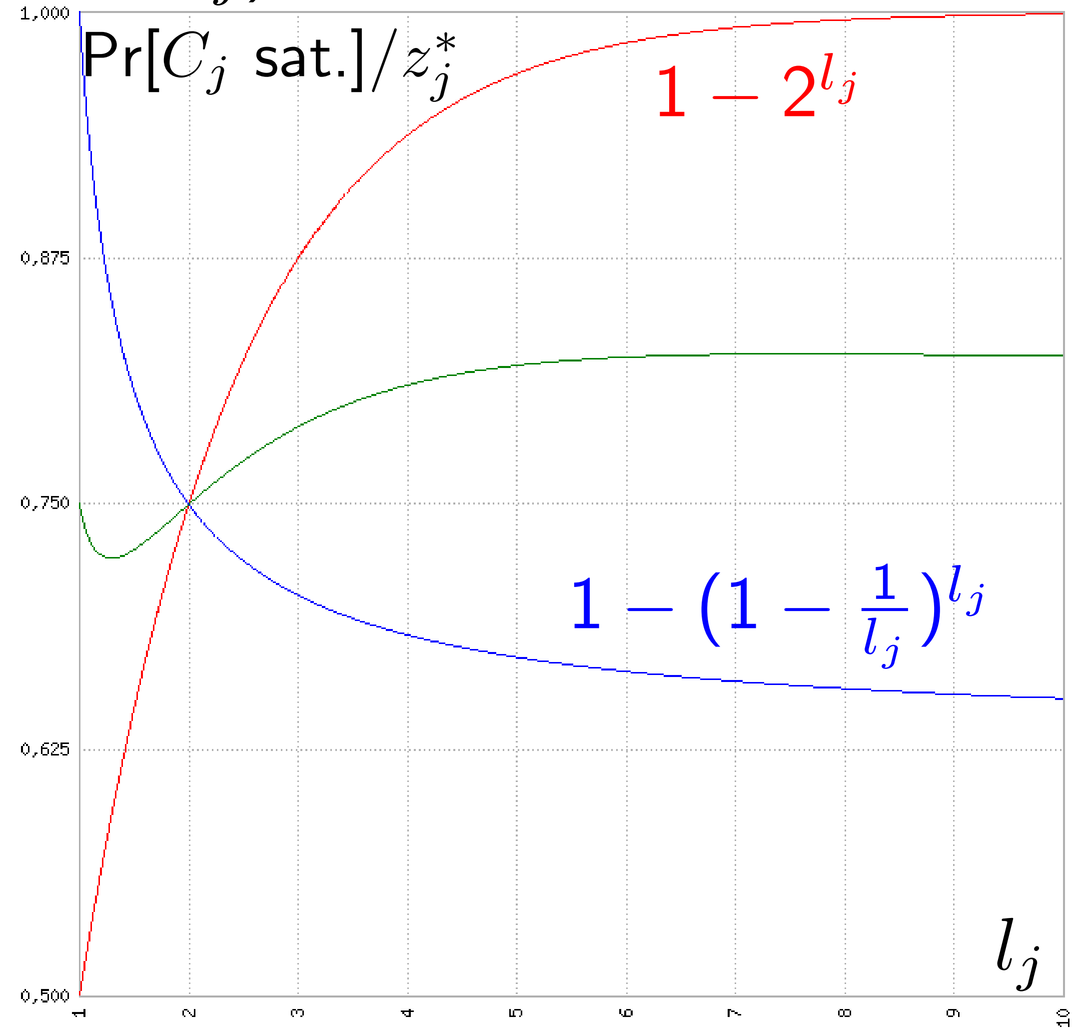
For $l_j = 1, 2$, a simple calculation shows $P = \frac{3}{4} \cdot z_j^*$

For $l_j \geq 3$, $1 - (1 - \frac{1}{l_j})^{l_j} \geq (1 - \frac{1}{e})$ and $1 - 2^{-l_j} \geq 7/8$. Thus, we have:

$$\frac{P}{z_j^*} \geq \frac{1}{2}\left[\left(1-\frac{1}{e}\right) + \frac{7}{8}\right] \approx 0{,}753 > \frac{3}{4} \qquad \square$$

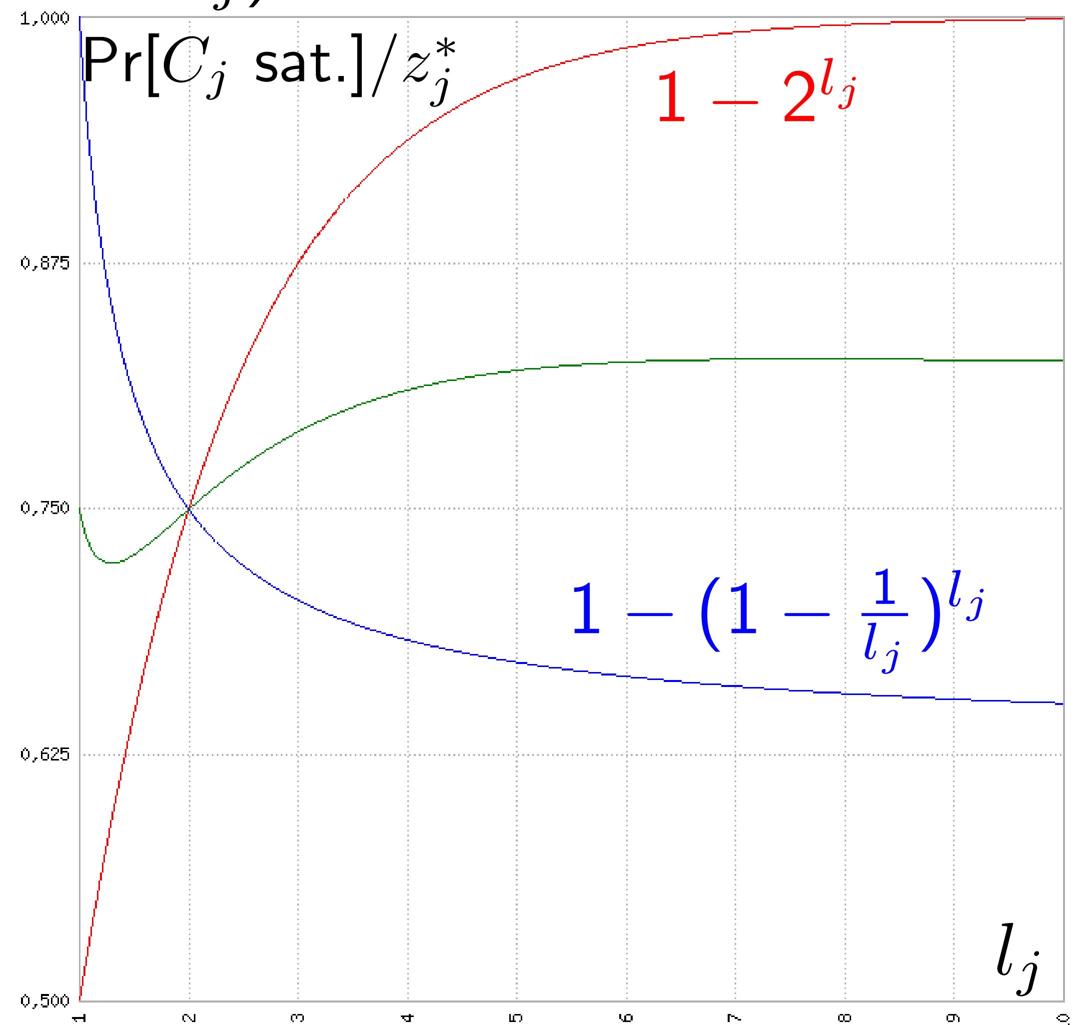
Visualization and Derandomization

Randomized alg. is better for large values of l_j
Randomized LP-rounding is better for small values of l_j
($\rightsquigarrow$ probability of satisfying clause C_j)



Visualization and Derandomization

Randomized alg. is better for large values of l_j
Randomized LP-rounding is better for small values of l_j
($\rightsquigarrow$ probability of satisfying clause C_j)



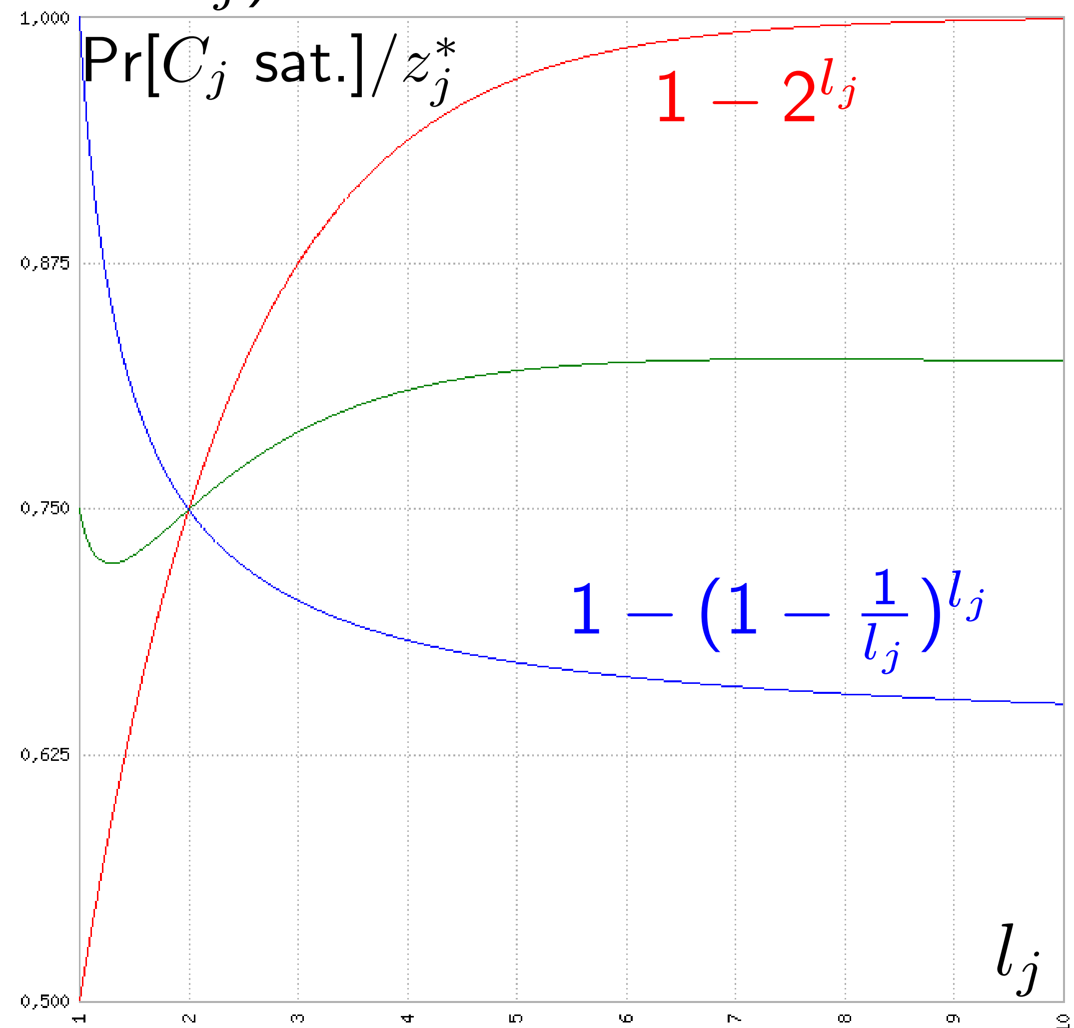
Mean of the two solutions is at least $\frac{3}{4}$ for all values of l_j.

Visualization and Derandomization

Randomized alg. is better for large values of l_j
Randomized LP-rounding is better for small values of l_j
($\leadsto$ probability of satisfying clause C_j)

Mean of the two solutions is at least $\frac{3}{4}$ for all values of l_j.

And, the maximum is at least as good as the mean.



Visualization and Derandomization

Randomized alg. is better for large values of l_j
Randomized LP-rounding is better for small values of l_j
($\rightsquigarrow$ probability of satisfying clause C_j)

Mean of the two solutions is at least $\frac{3}{4}$ for all values of l_j.

And, the maximum is at least as good as the mean.

This algorithm can also be derandomized by conditional expectation.