Approximate Counting and the Lovasz Local Lemma

Ankur Moitra (MIT)

BACKGROUND

Fundamental tool in the probabilistic method:

Lovasz Local Lemma (informal): A collection of events with bdded dependence has positive probability that no event occurs

BACKGROUND

Fundamental tool in the probabilistic method:

Lovasz Local Lemma (informal): A collection of events with bdded dependence has positive probability that no event occurs

In this talk, we'll focus on a well-known corollary for CNFs

BACKGROUND

Fundamental tool in the probabilistic method:

Lovasz Local Lemma (informal): A collection of events with bdded dependence has positive probability that no event occurs

In this talk, we'll focus on a well-known corollary for CNFs

Ex: $(\overline{x}_1 \vee x_3 \vee x_8) \wedge (x_1 \vee \overline{x}_6 \vee x_7) \wedge \ldots \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_7)$

Corollary: Any CNF formula with

(1) At least k variables per clause **(width)**

(2) Every clause intersects at most D others **(dependency degree)**

and $eD \leq 2^k$ has a satisfying solution

Corollary: Any CNF formula with

(1) At least k variables per clause **(width)**

(2) Every clause intersects at most D others **(dependency degree)**

and $eD \leq 2^k$ has a satisfying solution

Corollary: Any CNF formula with

(1) At least k variables per clause **(width)**

(2) Every clause intersects at most D others **(dependency degree)**

and $eD \leq 2^k$ has a satisfying solution

Some remarkable facts:

(1) This is tight – i.e. when the condition is violated, there are CNF formulas with no satisfying assignment

Corollary: Any CNF formula with

(1) At least k variables per clause **(width)**

(2) Every clause intersects at most D others **(dependency degree)**

and $eD \leq 2^k$ has a satisfying solution

Some remarkable facts:

(1) This is tight – i.e. when the condition is violated, there are CNF formulas with no satisfying assignment

(2) A random assignment can be exponentially unlikely to satisfy the formula

Corollary: Any CNF formula with

(1) At least k variables per clause **(width)**

(2) Every clause intersects at most D others **(dependency degree)**

and $eD \leq 2^k$ has a satisfying solution

Some remarkable facts:

(1) This is tight – i.e. when the condition is violated, there are CNF formulas with no satisfying assignment

(2) A random assignment can be exponentially unlikely to satisfy the formula

(3) There is an efficient algorithm to find a satisfying assignment due to **[Moser, Tardos '10]**

THE CONSTRUCTIVE LOCAL LEMMA

[Beck '91] gave an algorithm that works under the stronger precondition $D \leq 2^{k/8}/\text{poly}(k)$

THE CONSTRUCTIVE LOCAL LEMMA

[Beck '91] gave an algorithm that works under the stronger precondition $D \leq 2^{k/8}/\text{poly}(k)$

Theorem [Moser, Tardos '10]: There is an efficient algorithm for finding a satisfying assignment if $eD \leq 2^k$

THE CONSTRUCTIVE LOCAL LEMMA

[Beck '91] gave an algorithm that works under the stronger precondition $D \leq 2^{k/8}/\text{poly}(k)$

Theorem [Moser, Tardos '10]: There is an efficient algorithm for finding a satisfying assignment if $eD \leq 2^k$

Simplest, most elegant algorithm you can think of: while there is an unsatisfied clause, re-randomize its variables

THE CONSTRUCTIVE LOCAL LEMMA

[Beck '91] gave an algorithm that works under the stronger precondition $D \leq 2^{k/8}/\text{poly}(k)$

Theorem [Moser, Tardos '10]: There is an efficient algorithm for finding a satisfying assignment if $eD \leq 2^k$

Simplest, most elegant algorithm you can think of: while there is an unsatisfied clause, re-randomize its variables

---

Moser-Tardos works under some constraints on how events are described, many improvents and generalizations:

[Haeupler, Saha, Srinivasan '11] [Harris, Srinivasan '14]

[Achlioptas, Iliopoulis '14] [Harvey, Vondrak '15] [Kolmogorov '16]

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

Sampling Local Lemma:

When the LLL guarantees a solution exists, can we sample from the set of satisfying assignments uniformly at random?

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Sampling Local Lemma:

When the LLL guarantees a solution exists, can we sample from the set of satisfying assignments uniformly at random?

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Sampling Local Lemma:

When the LLL guarantees a solution exists, can we sample from the set of satisfying assignments uniformly at random?

Can we sample a needle uniformly at random?

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Sampling Local Lemma:

When the LLL guarantees a solution exists, can we sample from the set of satisfying assignments uniformly at random?

Can we sample a needle uniformly at random?

The Moser-Tardos algorithm can be thought of as a random walk that terminates when it finds a solution

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Sampling Local Lemma:

When the LLL guarantees a solution exists, can we sample from the set of satisfying assignments uniformly at random?

Can we sample a needle uniformly at random?

The Moser-Tardos algorithm can be thought of as a random walk that terminates when it finds a solution

But the distribution on solutions it finds can be far from uniform!

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Approximate Counting Local Lemma:

When the LLL guarantees a solution exists, can we count the number of satisfying assignments?

Can we count the number of needles?

Constructive Local Lemma:

When the LLL guarantees a solution exists, we can find it

A canonical example of finding a needle in a haystack

Approximate Counting Local Lemma:

When the LLL guarantees a solution exists, can we count the number of satisfying assignments?

Can we count the number of needles?

Counting under the LLL conditions is not self-reducible, but nevertheless we'll solve both problems simultaneously!

A FORAY INTO APPROXIMATE COUNTING

Many tight thresholds known for specific problems

A FORAY INTO APPROXIMATE COUNTING

Many tight thresholds known for specific problems, e.g.

Counting independent sets in the hard-core model: Given a graph $G = (V, E)$ with max degree d, approximate

$$Z(\lambda) = \sum_{I:\text{ independent set}} \lambda^{|I|}$$

A FORAY INTO APPROXIMATE COUNTING

Many tight thresholds known for specific problems, e.g.

Counting independent sets in the hard-core model: Given a graph G = (V, E) with max degree d, approximate

$$Z(\lambda) = \sum_{I\text{: independent set}} \lambda^{|I|}$$

[Weitz '06] gave an algorithm that works whenever

$$\lambda \leq \frac{(d-1)^{d-1}}{(d-2)^{d}}$$

A FORAY INTO APPROXIMATE COUNTING

Many tight thresholds known for specific problems, e.g.

Counting independent sets in the hard-core model: Given a graph G = (V, E) with max degree d, approximate

$$Z(\lambda) = \sum_{I:\text{ independent set}} \lambda^{|I|}$$

[Weitz '06] gave an algorithm that works whenever

$$\lambda \leq \frac{(d-1)^{d-1}}{(d-2)^{d}}$$

[Sly '10] showed that approximate counting is NP-hard above this

CONFLUENCE OF THRESHOLDS

All of the following happen together, for independent set:

(1) **Correlation decay:** When does fixing states of nodes far away from u have negligible effect of u's state?

CONFLUENCE OF THRESHOLDS

All of the following happen together, for independent set:

(1) **Correlation decay:** When does fixing states of nodes far away from u have negligible effect of u's state?

(2) **Uniqueness:** When is the Gibbs measure on the infinite tree unique?

CONFLUENCE OF THRESHOLDS

All of the following happen together, for independent set:

(1) **Correlation decay:** When does fixing states of nodes far away from u have negligible effect of u's state?

(2) **Uniqueness:** When is the Gibbs measure on the infinite tree unique?

(3) **Temporal mixing:** When does Gibbs sampling mix quickly?

CONFLUENCE OF THRESHOLDS

All of the following happen together, for independent set:

(1) **Correlation decay:** When does fixing states of nodes far away from u have negligible effect of u's state?

(2) **Uniqueness:** When is the Gibbs measure on the infinite tree unique?

(3) **Temporal mixing:** When does Gibbs sampling mix quickly?

(4) **Computational:** When does approximate counting go from easy to hard?

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

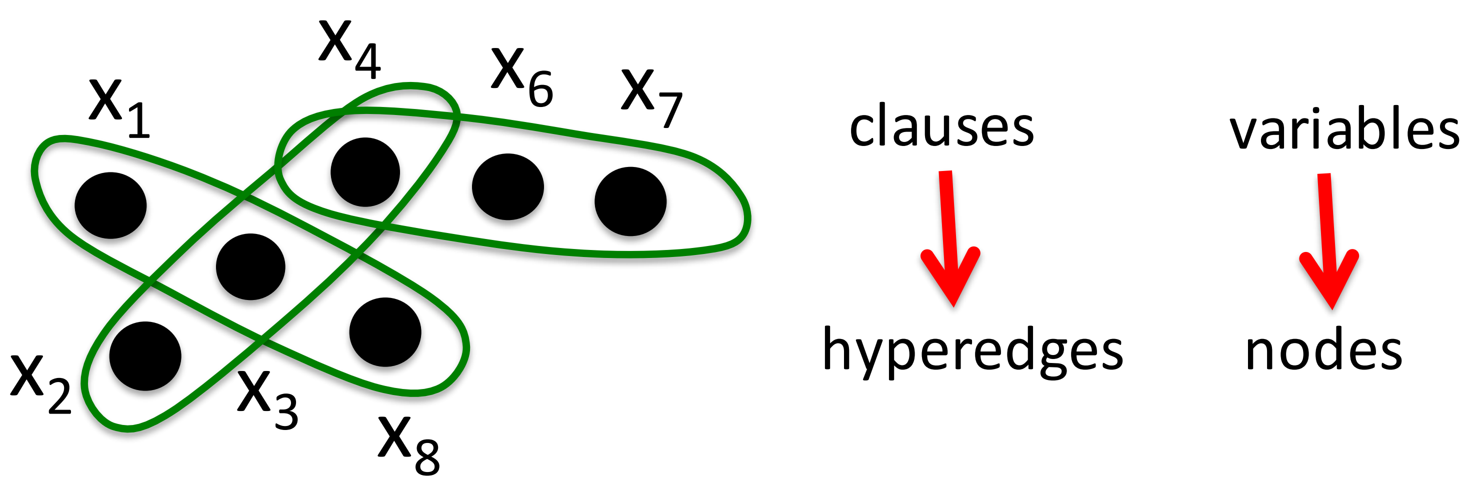
$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:

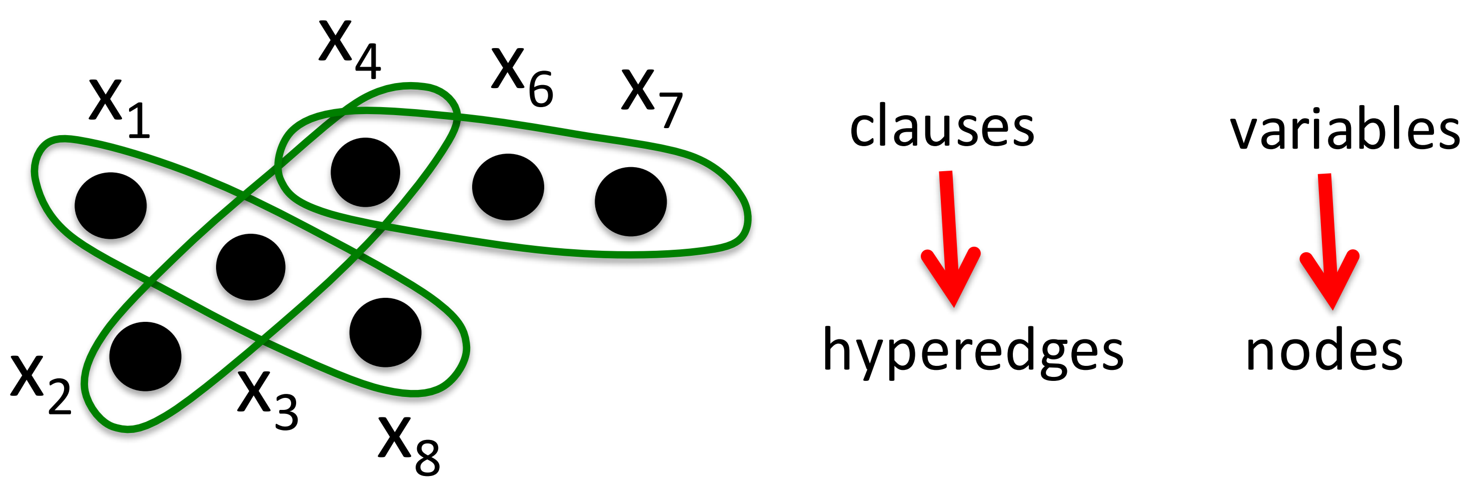


The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:



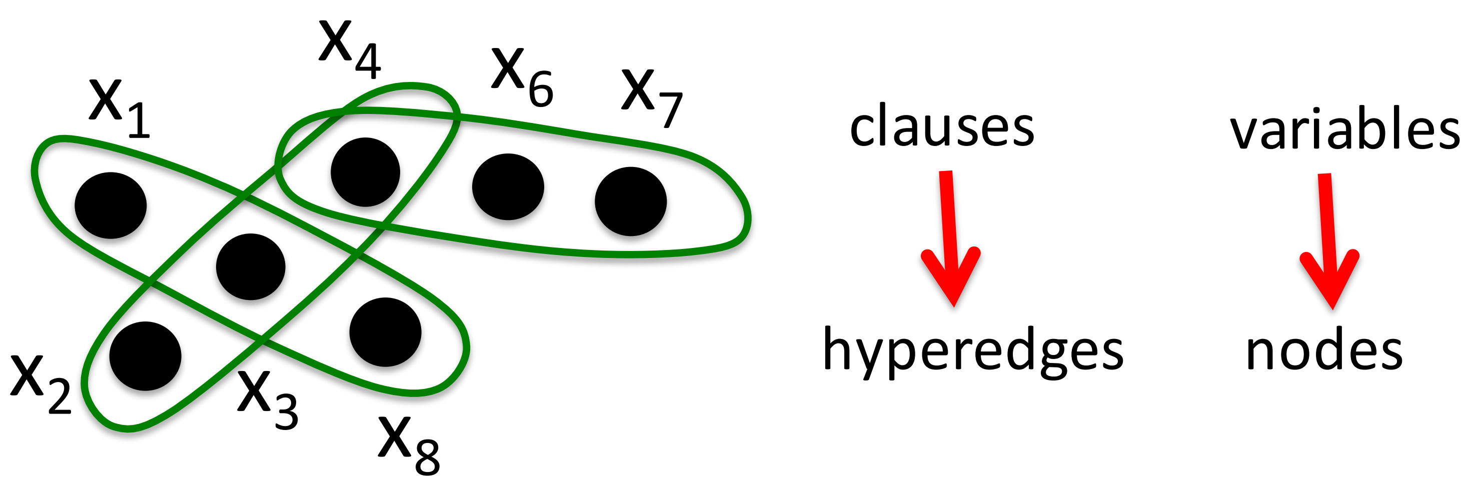
Definition: An independent set (in a hypergraph) is a set of nodes with no induced hyperedge

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:

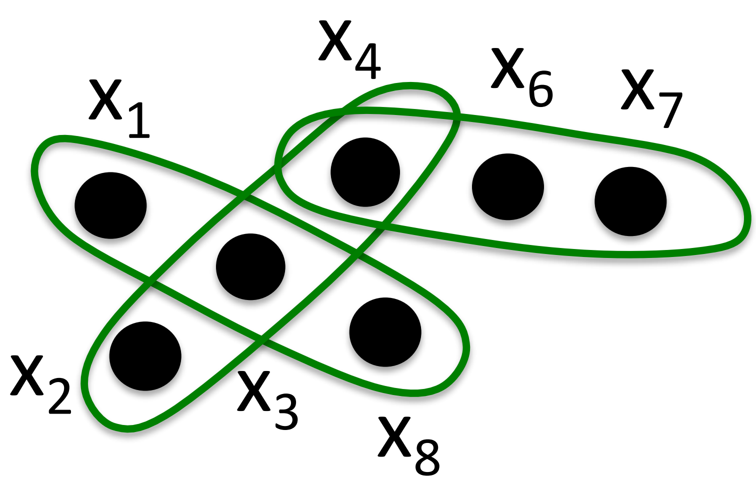


The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:



clauses → hyperedges

variables → nodes

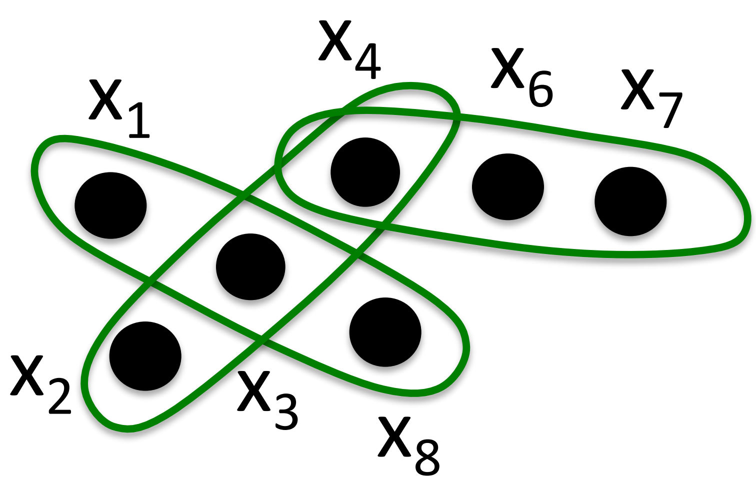
Claim: The number of satisfying assignments is equal to the number of independent sets

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:



clauses → hyperedges

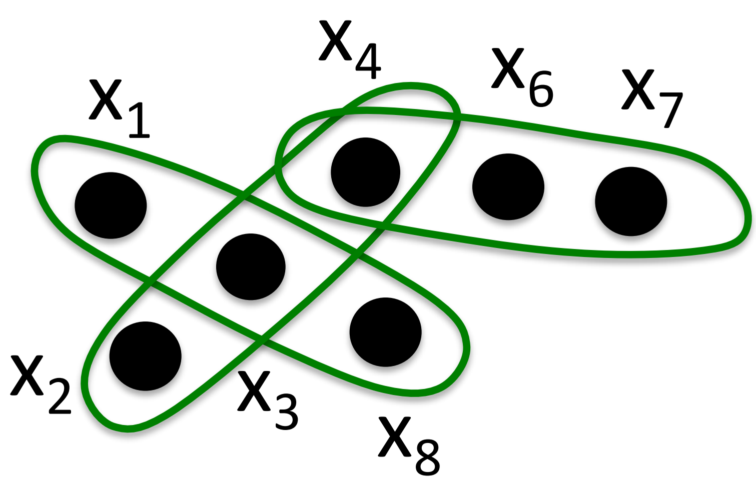
variables → nodes

The trouble is our problem is really about **hypergraphs**, where we have wide gaps in our understanding

Special Case: no variable is negated, e.g.

$$(x_1 \vee x_3 \vee x_8) \wedge (x_4 \vee x_6 \vee x_7) \wedge \ldots \wedge (x_2 \vee x_3 \vee x_4)$$

we get the **hypergraph independent set** problem:



clauses → hyperedges

variables → nodes

Comment: Let d be maximum degree, then $d \leq D \leq 2kd$ if at most 2k variables per clause

COMPLEXITY OF HYPERGRAPH INDEP. SET

Theorem [Bezakova et al. ‘16]: It is NP-hard to approximately count the number of hypergraph independent sets within an exponential factor if $d > 5 \cdot 2^{k/2}$

COMPLEXITY OF HYPERGRAPH INDEP. SET

Theorem [Bezakova et al. '16]: It is NP-hard to approximately count the number of hypergraph independent sets within an exponential factor if $d > 5 \cdot 2^{k/2}$

If you can approximately sample you can approximately count:

"It is NP-hard to approximately count/sample under the sharp Lovasz Local Lemma conditions"

COMPLEXITY OF HYPERGRAPH INDEP. SET

Theorem [Bezakova et al. '16]: It is NP-hard to approximately count the number of hypergraph independent sets within an exponential factor if $d > 5 \cdot 2^{k/2}$

If you can approximately sample you can approximately count:

"It is NP-hard to approximately count/sample under the sharp Lovasz Local Lemma conditions"

Best known algorithm:

Theorem [Bordewich, Dyer, Karpinski '06]: There is a randomized algorithm to approximately count if $d \leq k - 2$

COMPLEXITY OF HYPERGRAPH INDEP. SET

Theorem [Bezakova et al. '16]: It is NP-hard to approximately count the number of hypergraph independent sets within an exponential factor if $d > 5 \cdot 2^{k/2}$

If you can approximately sample you can approximately count:

"It is NP-hard to approximately count/sample under the sharp Lovasz Local Lemma conditions"

Best known algorithm:

Theorem [Bordewich, Dyer, Karpinski '06]: There is a randomized algorithm to approximately count if $d \leq k - 2$

[Bezakova et al.] gave a deterministic algorithm under same conds.

STRATIFICATION OF THRESHOLDS

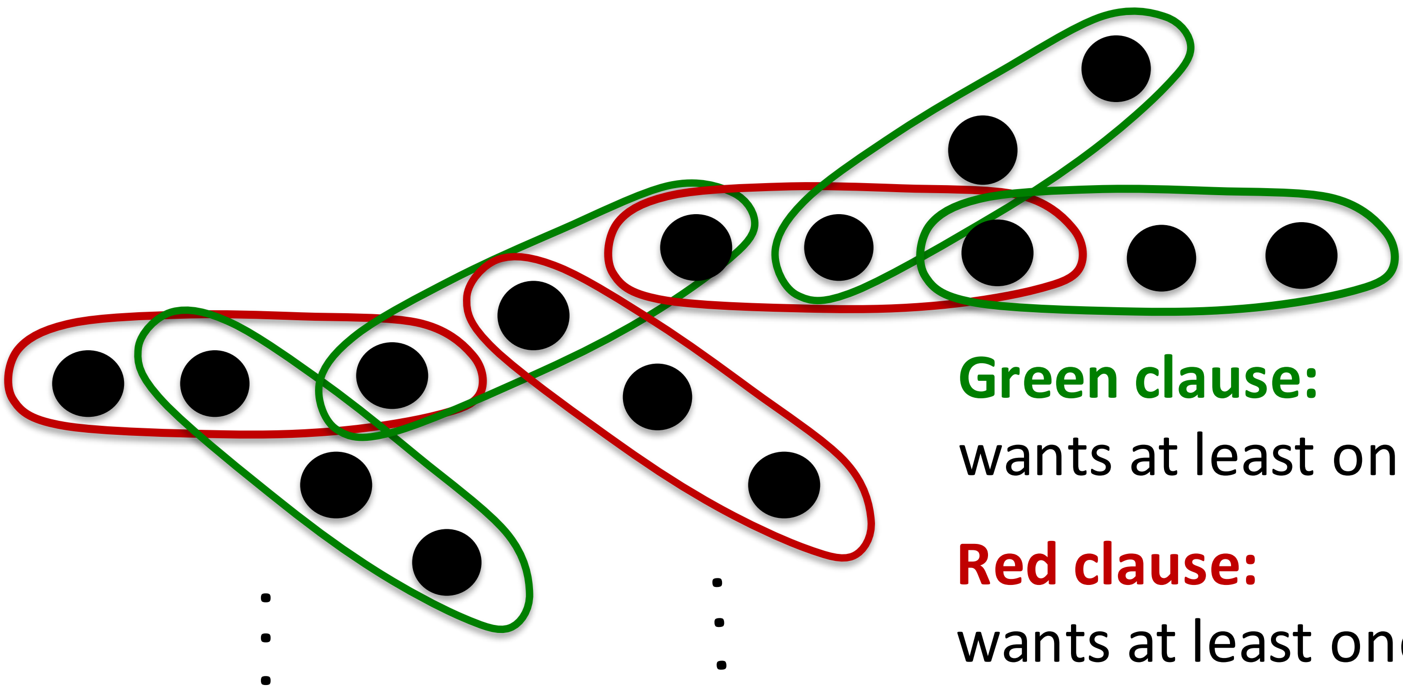
For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations



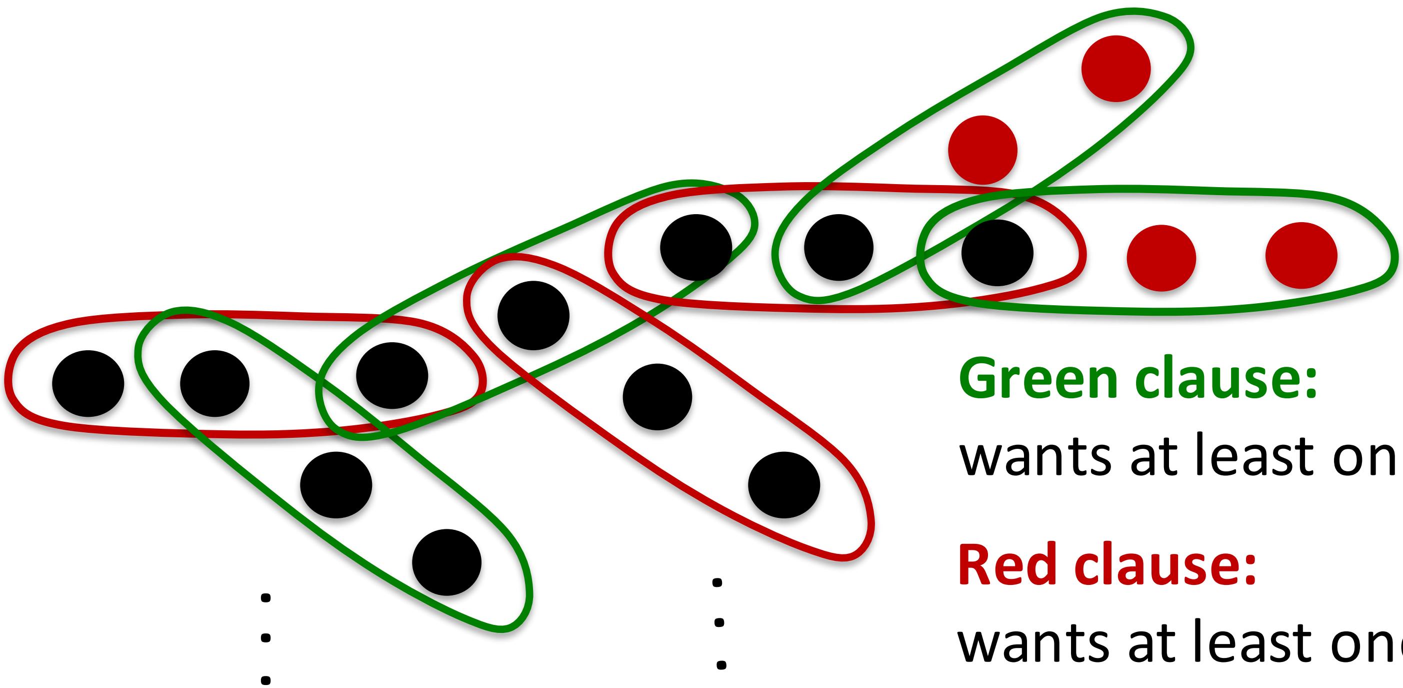
Green clause:
wants at least one green

Red clause:
wants at least one red

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations



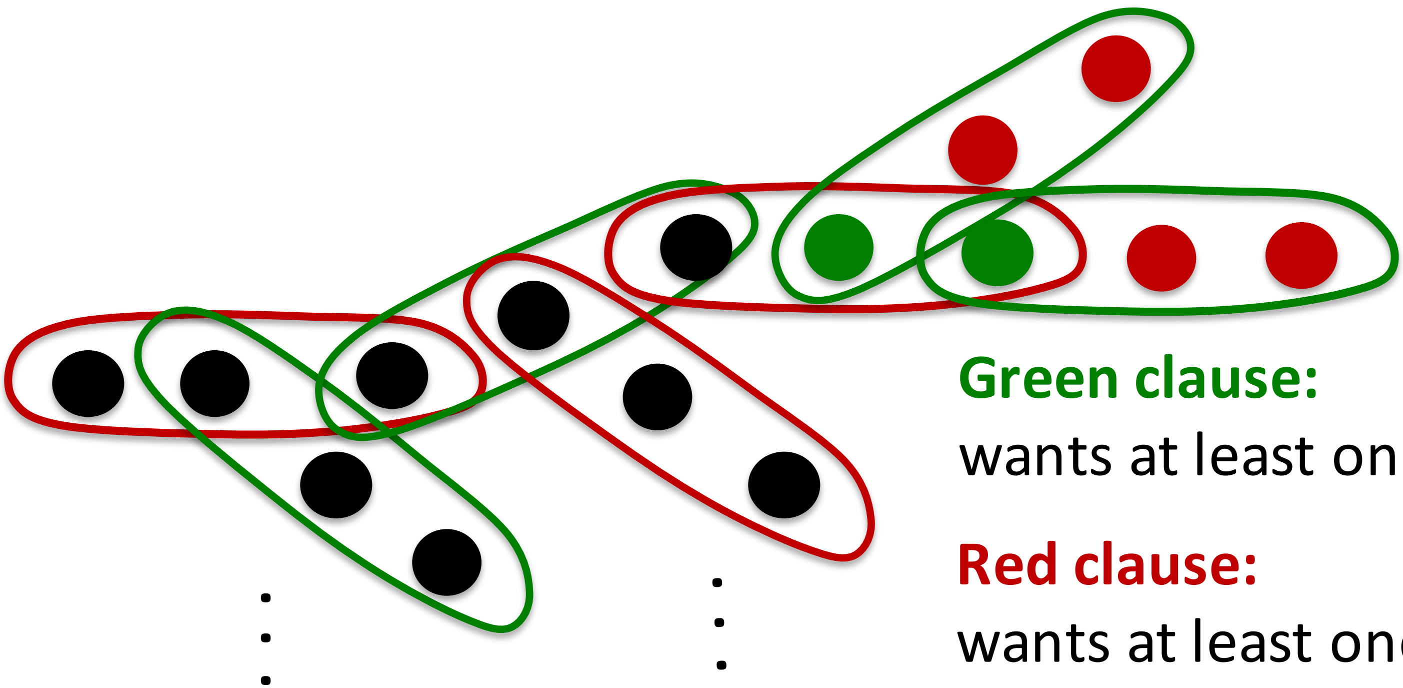
Green clause:
wants at least one green

Red clause:
wants at least one red

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations



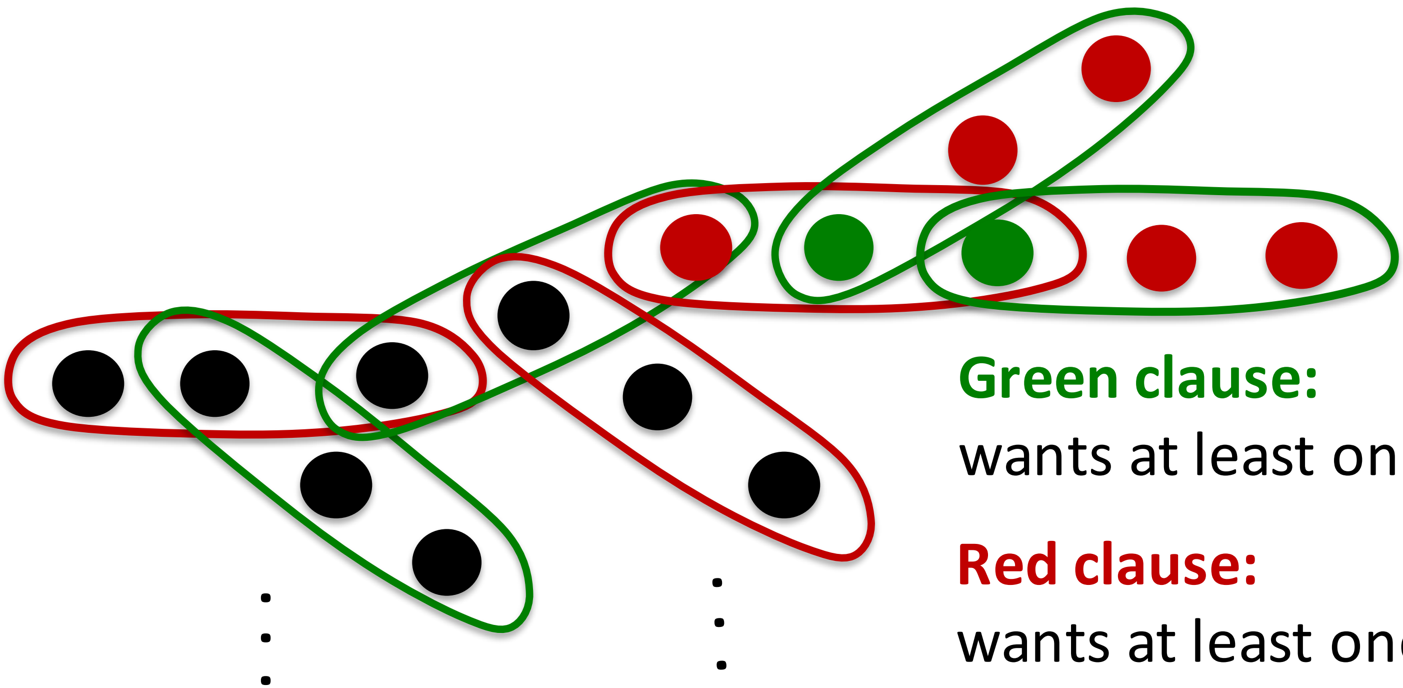
Green clause:
wants at least one green

Red clause:
wants at least one red

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations



Green clause:
wants at least one green

Red clause:
wants at least one red

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations

(2) **Uniqueness:** Approximate counting is NP-hard even when the Gibbs measure is unique, see **[Bezakova et al. '16]**

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations

(2) **Uniqueness:** Approximate counting is NP-hard even when the Gibbs measure is unique, see **[Bezakova et al. '16]**

(3) **Temporal mixing:** Uhhh, in **non-monotone case** the solution space is disconnected

STRATIFICATION OF THRESHOLDS

For approximate counting in bounded degree CNFs:

(1) **Correlation decay:** There can be long-range correlations

(2) **Uniqueness:** Approximate counting is NP-hard even when the Gibbs measure is unique, see **[Bezakova et al. '16]**

(3) **Temporal mixing:** Uhhh, in **non-monotone case** the solution space is disconnected

(4) **Computational:** Can we approximately count when the degree is exponential in the width?

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \bullet k^5 \leq D \leq c \bullet 2^{k/60}$

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

i.e. the algorithm outputs Z that satisfies:

$$Z \leq \# \begin{matrix}\text{satisfying} \\ \text{assignments}\end{matrix} \leq (1+n^{-T})\, Z$$

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

i.e. the algorithm outputs Z that satisfies:

$$Z \leq \#\,\substack{\text{satisfying} \\ \text{assignments}} \leq (1+n^{-T})\,Z$$

The degree of the polynomial depends polynomially on D and T

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

i.e. the algorithm outputs Z that satisfies:

$$Z \leq \#\,\substack{\text{satisfying} \\ \text{assignments}} \leq (1+n^{-T})\, Z$$

The degree of the polynomial depends polynomially on D and T

This is typical for deterministic algorithms, open: Can randomized algorithms do much better?

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

i.e. the algorithm outputs Z that satisfies:

$$Z \leq \# \begin{matrix}\text{satisfying} \\ \text{assignments}\end{matrix} \leq (1+n^{-T})\, Z$$

The technique is rather bizarre (even to me)

OUR RESULTS (COUNTING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a deterministic algorithm to approximately count the number of satisfying assignments if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

i.e. the algorithm outputs Z that satisfies:

$$Z \leq \# \begin{matrix}\text{satisfying} \\ \text{assignments}\end{matrix} \leq (1+n^{-T})\, Z$$

The technique is rather bizarre (even to me)

Also extends to non-binary counting problems

e.g. **red**, **green**, **blue** assignments with NAE constraints

OUR RESULTS (SAMPLING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a randomized algorithm to approximately sample from the set of satisfying assigs if $C \cdot k^5 \leq D \leq c \cdot 2^{k/60}$

OUR RESULTS (SAMPLING)

For general CNFs with between k and 2k variables per clause

Theorem: There is a randomized algorithm to approximately sample from the set of satisfying assigs if $C \bullet k^5 \leq D \leq c \bullet 2^{k/60}$

i.e. the output of the algorithm is n^{-T}-close in total variation distance to the uniform distribution on satisfying assignments

Concurrently and independently:

Theorem [Hermon, Sly, Zhang]: In the **monotone case**, there is a randomized algorithm to approx. count/sample if $D \leq c \cdot 2^{k/2}$

Concurrently and independently:

Theorem [Hermon, Sly, Zhang]: In the **monotone case**, there is a randomized algorithm to approx. count/sample if $d \leq c \cdot 2^{k/2}$

Theorem [Guo, Jerrum, Liu]: If **every pair of intersecting clauses shares at least min(log(dk), k/2) variables**, there is a randomzed algorithm to approx. count/sample if $d \leq c \cdot 2^{k/2}$

Concurrently and independently:

Theorem [Hermon, Sly, Zhang]: In the **monotone case**, there is a randomized algorithm to approx. count/sample if $d \leq c \bullet 2^{k/2}$

Theorem [Guo, Jerrum, Liu]: If **every pair of intersecting clauses shares at least min(log(dk), k/2) variables**, there is a randomzed algorithm to approx. count/sample if $d \leq c \bullet 2^{k/2}$

Both of these results are tight – it is NP-hard for larger d even for the types of restricted instances they consider

OUR APPROACH

OUR APPROACH

(1) Start with an oracle that can answer queries about the probability x=T/F under the uniform distribution on assignments given the current partial assignment **(thought experiment)**

OUR APPROACH

(1) Start with an oracle that can answer queries about the probability x=T/F under the uniform distribution on assignments given the current partial assignment **(thought experiment)**

(2) Use the oracle to build out a coupling between the satisfying solutions with x=T and with x=F

OUR APPROACH

(1) Start with an oracle that can answer queries about the probability x=T/F under the uniform distribution on assignments given the current partial assignment **(thought experiment)**

(2) Use the oracle to build out a coupling between the satisfying solutions with x=T and with x=F

This coupling is very special in how it is concise/avoids double counting, so that if I gave you it you could verify the coupling and compute the ratio Pr[x=T]/Pr[x=F]

OUR APPROACH

(1) Start with an oracle that can answer queries about the probability x=T/F under the uniform distribution on assignments given the current partial assignment **(thought experiment)**

(2) Use the oracle to build out a coupling between the satisfying solutions with x=T and with x=F

This coupling is very special in how it is concise/avoids double counting, so that if I gave you it you could verify the coupling and compute the ratio Pr[x=T]/Pr[x=F]

(3) Use linear programming to find this special type of coupling that we now know exists

Thanks!

Main Open Question:

Is $D \leq c \cdot 2^{k/2}$ the true threshold for algorithmically counting and sampling in k-CNFs?

Any Questions?