Fast Minimum Cross-Entropy Algorithms for Solving NP-hard Counting and Optimization Problems:

Presented Melbourne 2007

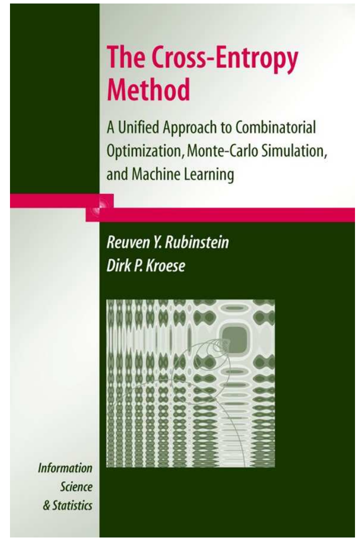
Reuven Rubinstein

Faculty of Industrial Engineering and Management,
Technion, Israel

Contents

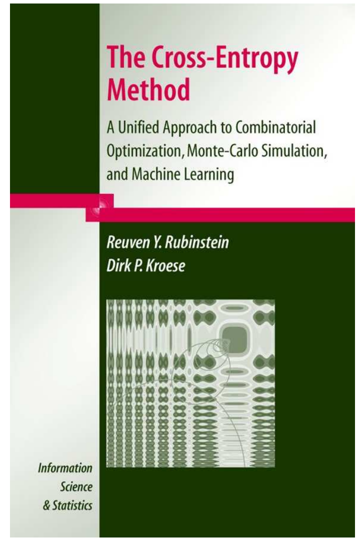
CE Matters

Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.



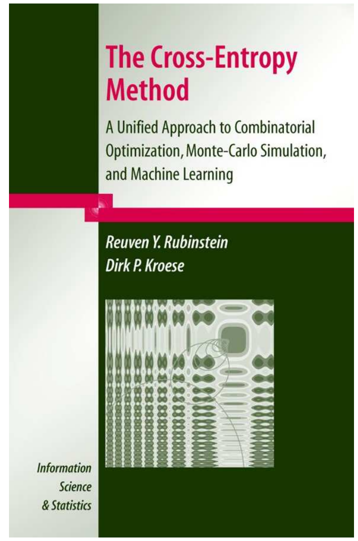
CE Matters

Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.



Special Issue: *Annals of Operations Research, 2005*

CE Matters



Book: R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*, Springer-Verlag, New York, 2004.

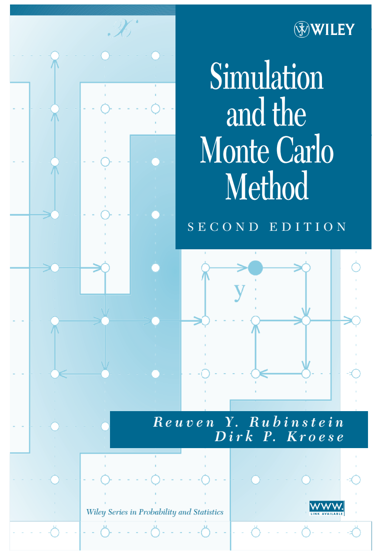
Special Issue: *Annals of Operations Research, 2005*

The CE **home page**:

`http://www.cemethod.org`

CE Matters

Book: R.Y. Rubinstein and D.P. Kroese. *Simulation and the Monte Carlo Method: Second Edition*, Wiley, 2007.



Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering
- Multi-extremal Continuous Optimization

Applications

- Combinatorial Optimization, like TSP, Maximal Cut, Scheduling and Production Lines.
- Machine Learning
- Pattern Recognition, Clustering and Image Analysis
- DNA Sequence Alignment
- Simulation-based (noisy) Optimization, like Optimal Buffer Allocation and Optimization in Finance Engineering
- Multi-extremal Continuous Optimization
- NP- hard Counting problems: Hamiltonian Cycles, SAW's, calculation the Permanent, Satisfiability Problem, etc.

#P-Complete Counting Problems

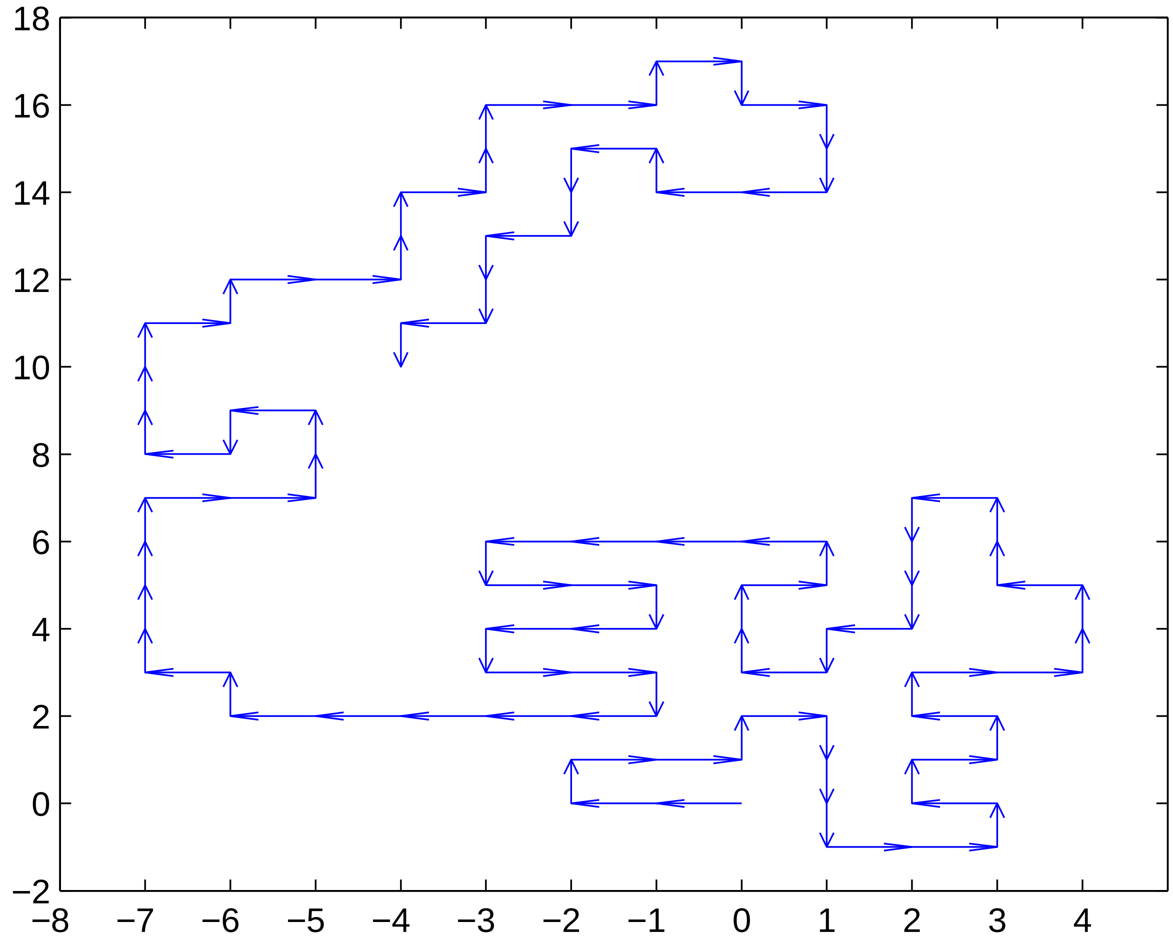
- The Hamiltonian cycle problem: How many Hamiltonian cycles has a graph?
- The permanent problem: What is the permanent of a matrix?
- The connectivity problem: Given two vertices in a graph, how many distinct paths are there between them?
- The self-avoiding random walk problem: How many self-avoiding random walks exist of size n?

More Examples

- The *k*-coloring problem. Given k distinct colors, in how many different ways can we color the nodes/edges, so that each two adjacent nodes/edges has a different color?
- The satisfiability problem. How many sets of Boolean variables satisfy a given set of Boolean clauses?

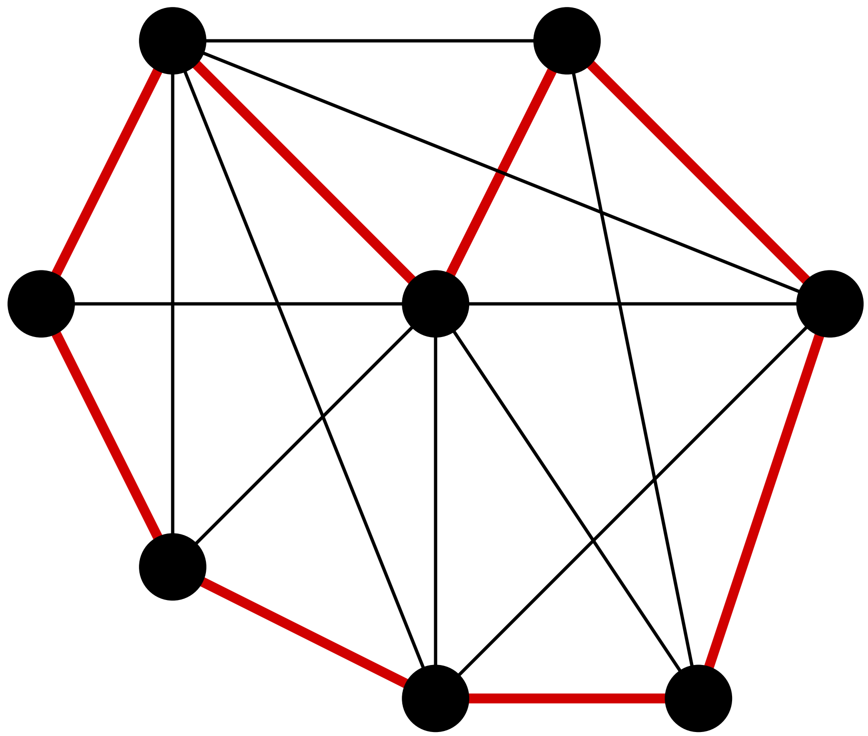
Self-Avoiding Walk (SAW)

This is an example of a SAW of length 100?



How many SAW's are there of length n?

Counting Hamiltonian Cycles



How many Hamiltonian cycles does this graph have?

Einstein's Statement

My intention is to give a simple intoduction to the Minimum Cross-Entropy (MinxEnt) methods for counting and optimization. But as Eintsain said:

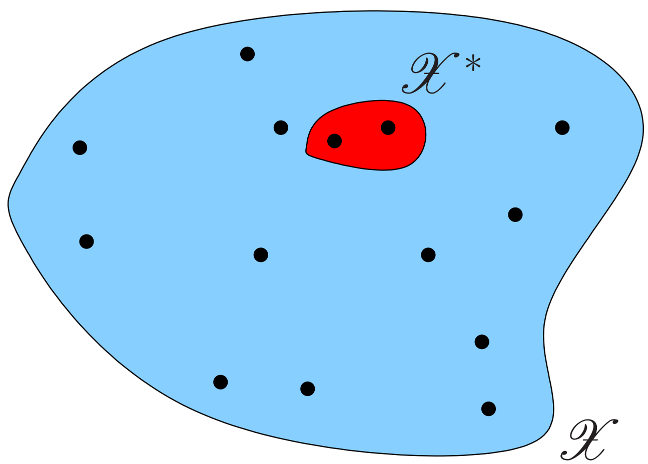
Every thing should be made as simple as possible but not simpler.

Counting via Monte Carlo

We start with the following basic

Example.

Assume we want to calculate an area of same “irregular" region $\mathcal{X}^*$. The Monte-Carlo method suggests inserting the ”irregular" region $\mathcal{X}^*$ into a nice “regular" one $\mathcal{X}$ as per figure below



$\mathscr{X}$: Set of objects (paths in a graph, colorings of a graph, etc.)

$\mathscr{X}^*$: Subset of special objects (cycles in a graph, colorings of a certain type, etc).

Counting via Monte Carlo

To calculate $|\mathcal{X}^*|$ we apply the following sampling procedure:

(i) Generate a random sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$, *uniformly* distributed over the "regular" region $\mathcal{X}$.

(ii) Estimate the desired area $|\mathcal{X}^*|$ as

$$\widehat{|\mathcal{X}^*|} = \widehat{\ell}|\mathcal{X}|,$$

where

$$\widehat{\ell} = \frac{N_{\mathcal{X}^*}}{N_{\mathcal{X}}} = \frac{1}{N}\sum_{k=1}^{N} I_{\{\boldsymbol{X}_k \in \mathcal{X}^*\}},$$

$I_{\{\boldsymbol{X}_k \in \mathcal{X}^*\}}$ denotes the indicator of the event $\{\boldsymbol{X}_k \in \mathcal{X}^*\}$ and $\{\boldsymbol{X}_k\}$ is a sample from $f(\boldsymbol{x})$ over $\mathcal{X}$, where $f(\boldsymbol{x}) = \frac{1}{|\mathcal{X}|}$.

Counting via Monte Carlo

The above formula

$$\widehat{|\mathcal{X}^*|} = \widehat{\ell}|\mathcal{X}|$$

is also valid for **counting problems**, that is where $\mathcal{X}^*$ presents a discrete rather a continuous set of points. For example, in HC problem

1. $\mathscr{X}$ is the entire set of tours in the graph. Note that $|\mathscr{X}| = (n-1)!$
2. $\mathscr{X}^*$ is the subset of tours of length n.

Counting via Rare-Events

Note that for counting problems

$$\ell = \frac{|\mathcal{X}^*|}{|\mathcal{X}|} = \mathbb{E}_{\mathsf{U}}[I_{\{\boldsymbol{X} \in \mathcal{X}^*\}}]$$

is typically very small, so the naive, crude Monte Carlo estimator of ℓ is useless. It is easy to show that using importance sampling we obtain

$$|\mathcal{X}^*| = \mathbb{E}_g[I_{\{\boldsymbol{X} \in \mathcal{X}^*\}} \frac{1}{g(\boldsymbol{X})}].$$

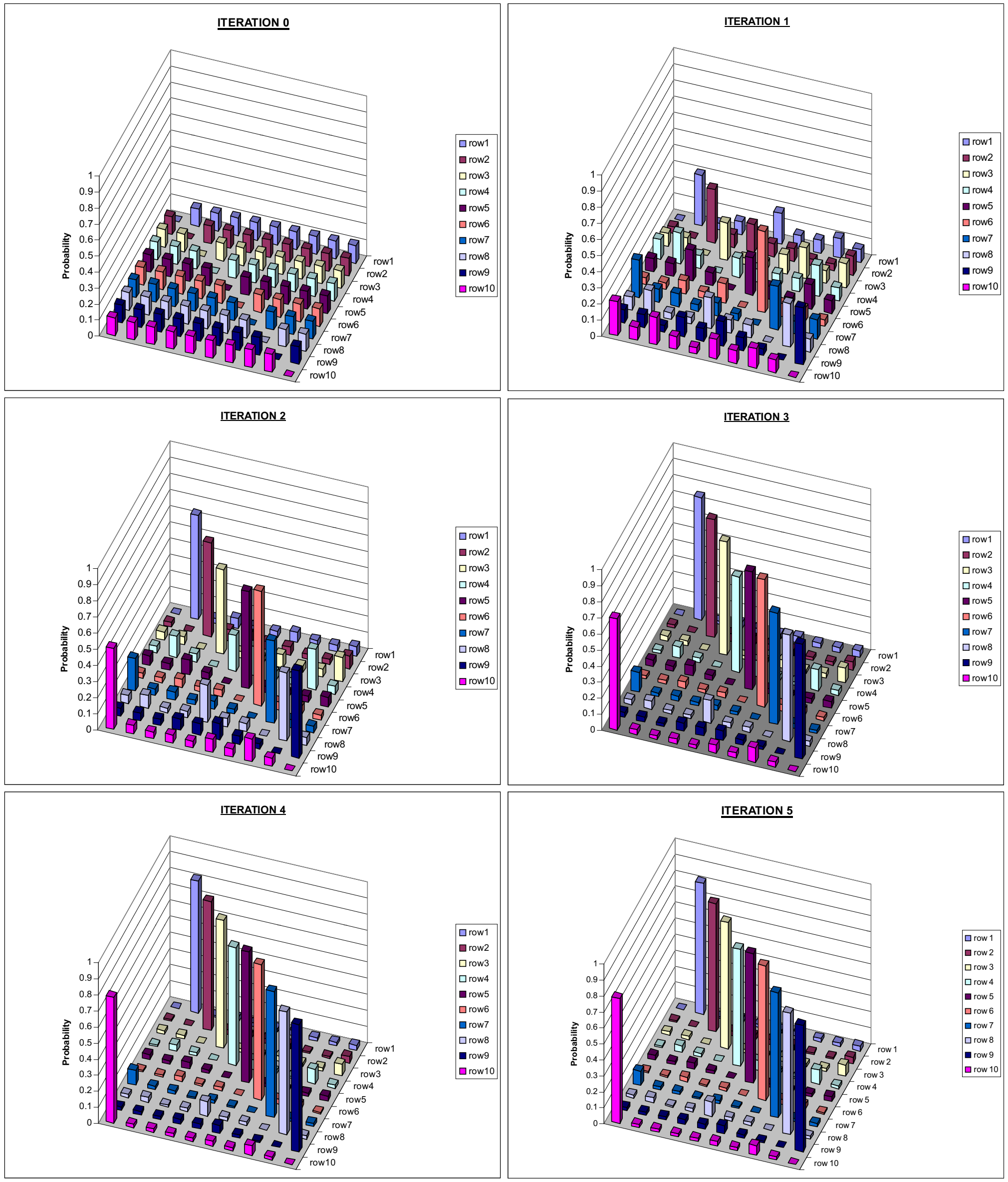
The IS estimate of $|\mathcal{X}^*|$ is therefore

$$\widehat{|\mathcal{X}^*|} = \frac{1}{N} \sum_{k=1}^{N} I_{\{\boldsymbol{X}_k \in \mathcal{X}^*\}} \frac{1}{g(\boldsymbol{X}_k)} = \sum_{\boldsymbol{X}_k \in \mathcal{X}^*} \frac{1}{g(\boldsymbol{X}_k)} \,.$$

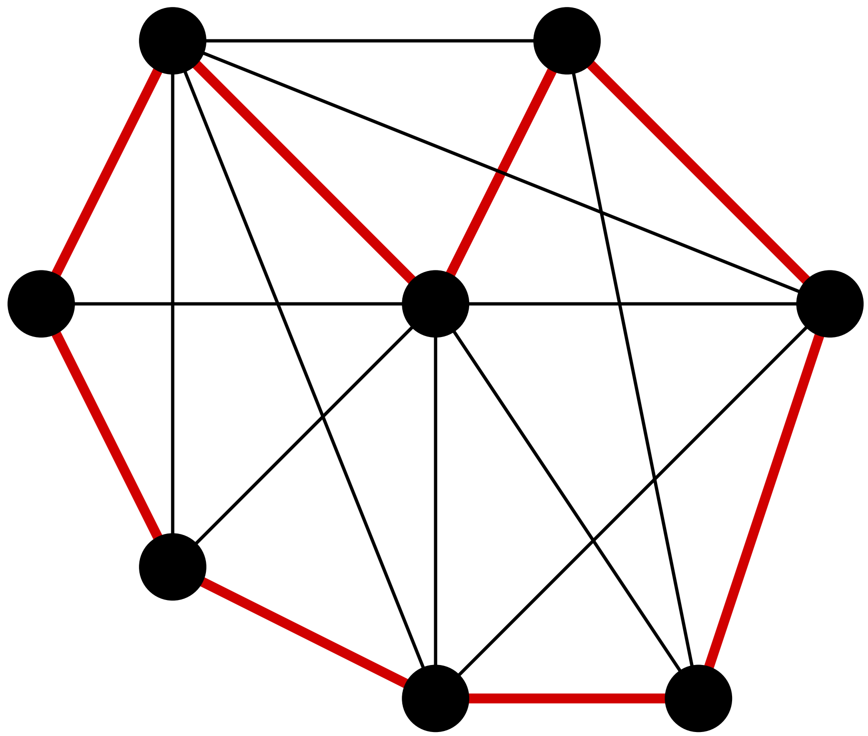
Chosing the IS pdf $g(\boldsymbol{x})$

The best choice for g is, clearly, $g^*(\boldsymbol{x}) = 1/|\mathcal{X}^*|$, $\boldsymbol{x} \in \mathcal{X}^*$, which is the *uniform distribution on* $\mathcal{X}^*$. Under g^* the estimator has **zero variance**, since the random variable $\widehat{|\mathcal{X}^*|} = const$, so that only one sample is required. However, sampling from such g^* is impractical, since it requires availability of our target value $|\mathcal{X}^*|$. To overcome this difficulty we shall show in the following sections how to construct ”good" (law variance) IS sample pdf’s $g(\boldsymbol{x})$ (parametric and nonparametric) for different #P-complete counting problems.

Combinatorial Optimization: TSP

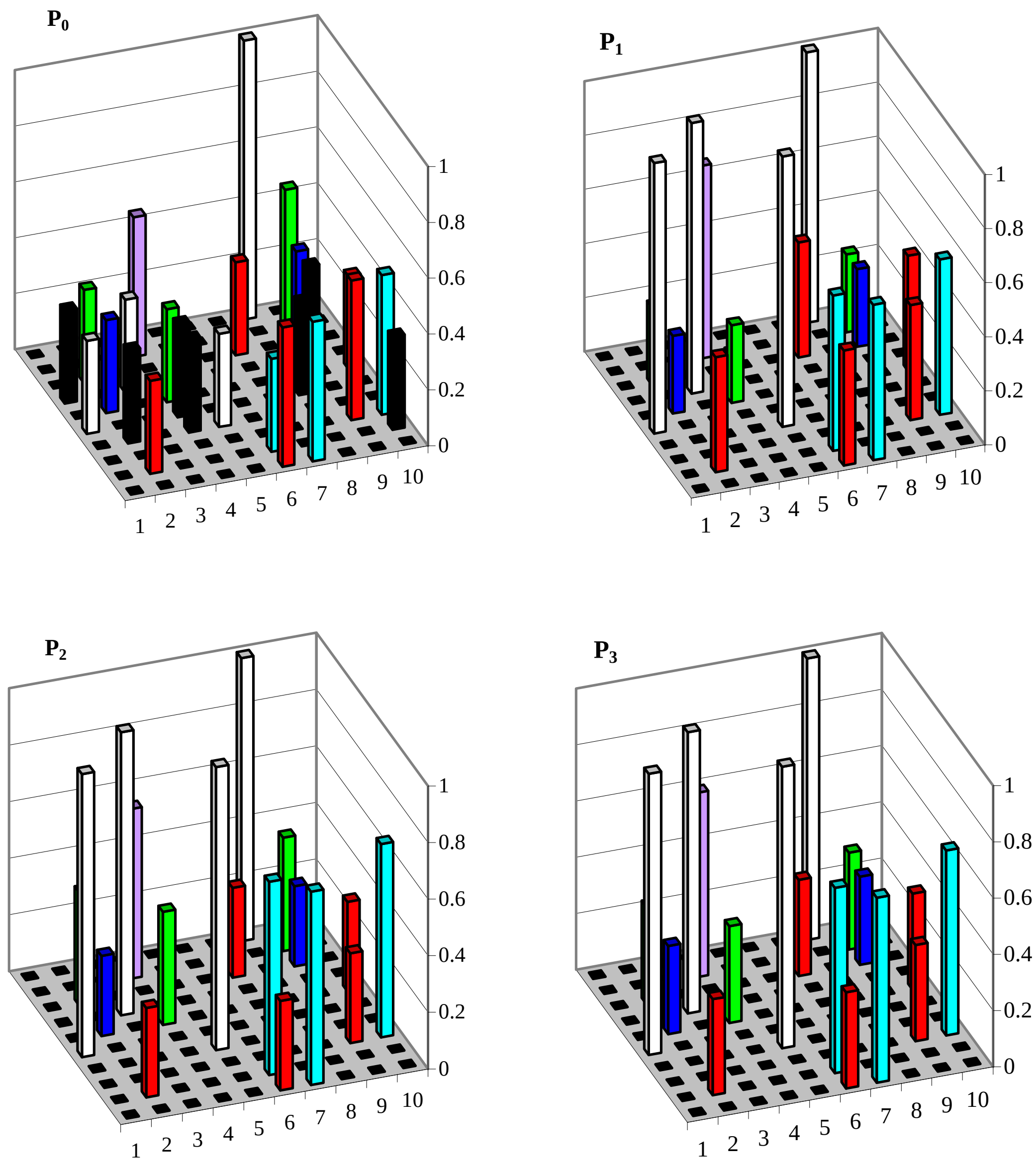


Counting Hamiltonian Cycles



How many Hamiltonian cycles does this graph have?

Calculating the Number of HC's



Hamiltonian Cycles: Performance of our Algorithm

Performance of Algorithm for $n = 30$, $\eta = 0.15$ and $N = 100,000$

t		0	1	2	3	4	5	6
$\widehat{\|\mathcal{X}^-\|}$	Average	8.1E+04	7.2E+04	6.1E+04	7.7E+04	6.6E+04	6.5E+04	6.5E+04
	Min	1.2E+04	4.8E+04	3.9E+04	5.1E+04	5.4E+04	4.5E+04	4.4E+04
	Max	1.9E+05	1.5E+05	7.8E+04	3.5E+05	1.0E+05	8.8E+04	8.8E+04
ε	$\bar{\varepsilon}$	0.535	0.237	0.132	0.571	0.122	0.144	0.210
	ε_*	0.007	0.010	0.006	0.053	0.009	0.028	0.029
	ε^*	1.324	1.106	0.354	1.630	0.567	0.356	0.355
κ		0.677	0.405	0.383	0.251	0.211	0.188	0.124

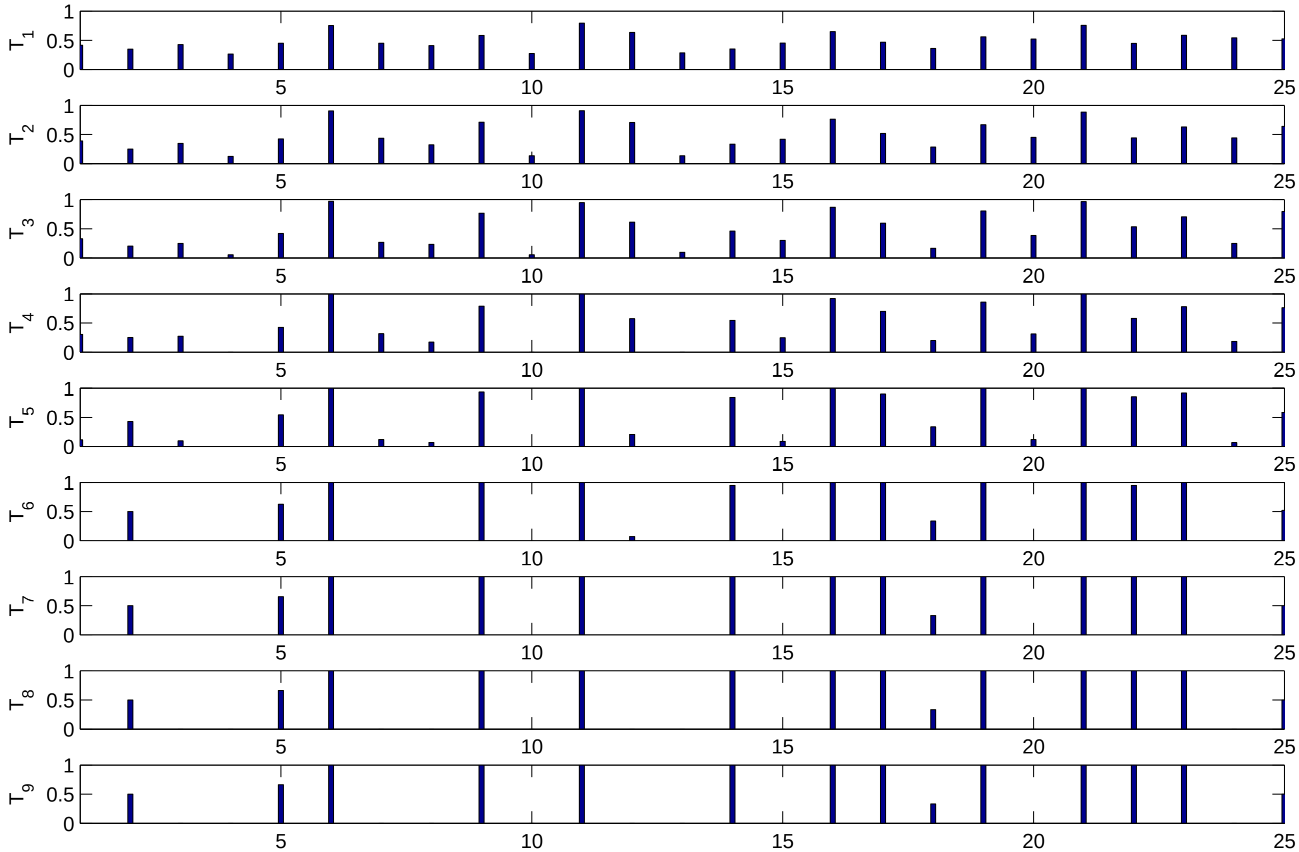
Satisfiability Problem: Numerical Results

Table 1: Performance of the MinxEnt algorithm for the random 2-SAT with the clause matrix $A = (100 \times 110)$ and $N = 150000$

	$\widehat{\lvert\mathcal{X}^*\rvert}$				
t	Mean	Max	Min	PV	RE
3	0.00	0	0	0.0000	NaN
4	5.1e+012	2.9e+013	0	0.0000	1.8781
6	8.9e+013	2.4e+014	5.3e+008	0.0062	0.9833
8	5.3e+013	1.6e+014	6.0e+011	0.1256	0.9699
10	8.3e+013	1.4e+014	3.4e+013	0.1344	0.3398
12	8.5e+013	1.1e+014	6.0e+013	0.1019	0.1748

Satisfiability Problem: Numerical Results

Figure 2: Typical dynamics of the MinxEnt algorithm



Rare-Events and MinxEnt (MCE)

Assume we want to estimate a rare events probability
$\ell = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \leq b\}}$.
The flow chart of the connection between rare events probability
$\ell = \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \leq b\}}$, MCE and counting with a single constraint looks like

$$\{\boldsymbol{x} : S(\boldsymbol{x}) \leq b\} \longrightarrow \mathbb{E}_{\boldsymbol{u}} I_{\{S(\boldsymbol{X}) \leq b\}} \longrightarrow \text{MCE} \longrightarrow \text{Count.}$$

Consider next counting on the set
$\mathcal{X}^* = \{\boldsymbol{x} : S_i(\boldsymbol{x}) \geq b_i,\ j = 1, \ldots, m\}$.
In this case the flow chart can be extended as

$$\{\bigcap_{i=1}^{m} (S_i(\boldsymbol{x}) \leq b_i)\} \rightarrow \mathbb{E}_{\boldsymbol{u}} \prod_{i=1}^{m} I_{\{S_i(\boldsymbol{X}) \leq b_i\}} \rightarrow \text{MCE} \rightarrow \text{Count } |\mathcal{X}^*|.$$

Background on MinxEnt Program (MCE)

$$\min_g \left\{ \mathcal{D}(g|h) = \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})} g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_g \ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right\}$$

$$(\mathrm{P}_0) \qquad \text{s.t.} \quad \int S_j(\boldsymbol{x}) g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_g S_j(\boldsymbol{X}) = b_j, \; j = 1, \ldots, k,$$

$$\int g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1, \; \int h(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1. \qquad (1)$$

Here g and h are joint n-dimensional pdf's or n-dimensional pmf's, $S_j(\boldsymbol{x}), \; j = 1, \ldots, k$, are known functions of an n-dimensional vector $\boldsymbol{x}$ and h is a known pdf, called the prior pdf.

The MaxEnt Program

When h is unknown, it is taken as a uniform pdf. In this case

$$\min_{g(\boldsymbol{x})} \left\{ \mathcal{D}(g|h) = \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})} g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \mathbb{E}_g \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})} \right\} \tag{2}$$

reduces to

$$\max_{g(\boldsymbol{x})} \left\{ \mathcal{H}(g) = -\int g(\boldsymbol{x}) \ln g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = -\mathbb{E}_g \ln g(\boldsymbol{x}) \right\}. \tag{3}$$

The original program (P_0) is called the Kullback MinxEnt program, while the one with uniform h is called the Janes MaxEnt program.

Single Constraint MinxEnt Program

We shall deal mainly with the single constrained program

$$\mathbb{E}_g S(\boldsymbol{X}) = b, \ \left(\int g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1\right). \tag{4}$$

In this case the solution of the program (P_0) is

$$g(\boldsymbol{x}) = \frac{h(\boldsymbol{x}) \exp\{-S(\boldsymbol{x})\lambda\}}{\mathbb{E}_h \exp\{-S(\boldsymbol{X})\lambda\}} \tag{5}$$

and

$$\frac{\mathbb{E}_h S(\boldsymbol{X}) \exp\{-\lambda S(\boldsymbol{X})\}}{\mathbb{E}_h \exp\{-\lambda S(\boldsymbol{X})\}} = b, \tag{6}$$

respectively. Note that λ should be obtained numerically.

Die Tossing Example

The Program:

$$\min_{\boldsymbol{p}} \mathcal{D}(\boldsymbol{p}|\boldsymbol{u}) = \min_{\boldsymbol{p}} \sum_{i=1}^{6} p_i \ln \frac{p_i}{u_i}$$

$$\textbf{s. t.} \quad \sum_{i=1}^{6}, \; i p_i = b, \; \sum_{i=1}^{6} p_i = 1, \; p_i \geq 0.$$

The solution:

$$p_i^* = \frac{u_i \exp\{-i\lambda^*\}}{\sum_{r=1}^{6} u_r \exp\{-r\lambda^*\}} = \frac{\mathbb{E}_{\boldsymbol{u}} I_{\{X=i\}} \exp\{-X\lambda^*\}}{\mathbb{E}_{\boldsymbol{u}} \exp\{-X\lambda^*\}}, \; i = 1, \ldots, 6,$$

and similarly λ.

Die Tossing Example

λ, $\boldsymbol{p}$ and $\mathcal{H}(\boldsymbol{p})$ as function of b for a fair die.

b	λ	p_1	p_2	p_3	p_4	p_5	p_5	$\mathcal{H}(\boldsymbol{p})$
1.0	∞	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.00000
1.5	1.0870	0.6637	0.2238	0.0755	0.0255	0.0086	0.0029	0.95356
2.0	0.6296	0.4781	0.2548	0.1357	0.0723	0.0385	0.0205	1.36724
2.5	0.3710	0.3475	0.2398	0.1654	0.1142	0.0788	0.9544	1.61373
3.0	0.1746	0.2468	0.2072	0.1740	0.1461	0.1227	0.1031	1.74843
3.5	0.0000	0.1666	0.1666	0.1666	0.1666	0.1666	0.1666	1.79176
4.0	-0.1746	0.1031	0.1227	0.1461	0.1740	0.2072	0.2468	1.74843
4.5	-0.3710	0.0544	0.0788	0.1142	0.1654	0.2398	0.3475	1.61373
5.0	-0.6296	0.0205	0.0385	0.0723	0.1357	0.2548	0.4781	1.36724
5.5	-1.0870	0.0029	0.0086	0.0255	0.0755	0.2238	0.6637	0.95356
6.0	$-\infty$	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.00000

Rare-Events and MCE

Statement: While estimating $\ell = \mathbb{E}_h \left\{ I_{\{S(\boldsymbol{X}) \geq b\}} \right\}$ we shall use the IS estimator

$$\widetilde{\ell} = \frac{1}{N} \sum_{i=1}^{N} \left\{ I_{\{S(\boldsymbol{X}_i) \geq b\}} \frac{h(\boldsymbol{X}_i)}{g(\boldsymbol{X}_i)} \right\},$$

where the pdf g is obtained from the solution of the MinxEnt program

$$\begin{aligned} & \min_g \mathbb{E}_h \mathcal{D}(g,h) = \min_g \int \ln \tfrac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\, g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \min_g \mathbb{E}_g \left[\ln \tfrac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right] \\ \text{s.t.} \quad & \mathbb{E}_g [S(\boldsymbol{X})] \geq b, \\ & \int g(\boldsymbol{x}) \mathrm{d}\, \boldsymbol{x} = 1. \end{aligned} \tag{7}$$

Rare-Events and MCE

We shall often use the parametric IS for ℓ,that is

$$\widehat{\ell} = \frac{1}{N} \sum_{k=1}^{N} \left[I_{\{S(\boldsymbol{X}_k) \geq b\}} \frac{f(\boldsymbol{X}_k; \boldsymbol{u})}{f(\boldsymbol{X}_k; \boldsymbol{p})} \right], \tag{8}$$

where $h(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{u})$ and the components of $\boldsymbol{p}$ in $f(\boldsymbol{x}, \boldsymbol{p})$ are

$$p_j = \frac{\mathbb{E}_{\boldsymbol{u}} X_j \exp\{-S(\boldsymbol{X})\lambda\}}{\mathbb{E}_{\boldsymbol{u}} \exp\{-S(\boldsymbol{X})\lambda\}}, \; i = 1, \ldots, n \tag{9}$$

This is acheaved as follows: by summing $g(\boldsymbol{x})$ over all $x_k, k \neq j$, we obtain the marginal pdf for the j-th component and then we calculate $p_j = \mathbb{E}_g[X_j]$

Multiple Rare-Events and MCE

Consider next counting on the set

$$\mathcal{X}^* = \{S_i(\boldsymbol{x}) \geq b_i,\ j = 1, \ldots, m\},$$

where all components of $\boldsymbol{x}$ are integers, say are binary variables. In this case we associated with the above set the following extended version of $\ell = \mathbb{P}_{\boldsymbol{u}}\{S(\boldsymbol{X}) \geq b]\}$

$$\ell = \mathbb{P}_{\boldsymbol{u}}\left\{\bigcap_{i=1}^{m}[S_i(\boldsymbol{X}) \geq b_i]\right\} = \mathbb{E}_{\boldsymbol{u}}\left[\prod_{i=1}^{m} I_{\{S_i(\boldsymbol{X}) \geq b_i\}}\right].$$

Multiple Rare-Events and MCE

We write $\ell = \mathbb{P}_{\boldsymbol{u}}\left\{\bigcap_{i=1}^{m}[S_i(\boldsymbol{X}) \geq b_i]\right\}$ as

$$\ell = \mathbb{P}_{\boldsymbol{u}}\left\{\left(\sum_{i=1}^{m} C_i(\boldsymbol{X})\right) = m\right\} = \mathbb{E}_{\boldsymbol{u}}\left[I_{\{\sum_{i=1}^{m} C_i(\boldsymbol{X})=m\}}\right].$$

where

$$\mathcal{C} = \sum_{i=1}^{m} C_i(\boldsymbol{X}) = \sum_{i=1}^{m} I_{\{S_i(\boldsymbol{X}) \geq b_i\}},$$

and $C_i(\boldsymbol{X}) = I_{\{S_i(\boldsymbol{X}) \geq b_i\}}$. Thus, ℓ is the probability of the sum of random variables $C_i(\boldsymbol{X})$ being equal to m, where m is the number of constraints.

Multiple Rare-Events and MCE

To estimate ℓ via IS we use the following *single-constrained* MCE program

$$\begin{aligned} &\min_g \mathbb{E}_g \mathcal{D}(g,h) = \min_g \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\, g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \min_g \mathbb{E}_g \left[\ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right] \\ \text{s.t.} \quad &\mathbb{E}_g [\textstyle\sum_{i=1}^m C_i(\boldsymbol{X})] = m \\ &\int g(\boldsymbol{x}) \mathrm{d}\,\boldsymbol{x} = 1. \end{aligned} \tag{10}$$

Note that in order for the above program to satisfy all m indicator random variables each $C_i(\boldsymbol{X})$ must be equal 1.

Multiple Rare-Events and MCE

The solution is of the original MCE problem with the constraint

$$\mathbb{E}_g[\sum_{i=1}^{m} C_i(\boldsymbol{X})] = m$$

is

$$g(\boldsymbol{x}) = \frac{h(\boldsymbol{x}, \boldsymbol{u}) \exp\{-\sum_{i=1}^{m} \lambda C_i(\boldsymbol{X})\}}{\mathbb{E}_{\boldsymbol{u}}\left[\exp\{-\sum_{i=1}^{m} \lambda C_i(\boldsymbol{X})\}\right]}, \tag{11}$$

where

$$\frac{\mathbb{E}_{\boldsymbol{u}}\left[\sum_{i=1}^{m} C_i(\boldsymbol{X}) \exp\left\{-\sum_{j=1}^{m} \lambda C_j(\boldsymbol{X})\right\}\right]}{\mathbb{E}_{\boldsymbol{u}}\left[\exp\left\{-\sum_{j=1}^{m} \lambda C_j(\boldsymbol{X})\right\}\right]} = m. \tag{12}$$

Multiple Rare-Events and MCE

Lemma 1 The optimal λ is

$$\lambda = -\infty.$$

Lemma 2 The optimal pdf $g(\boldsymbol{x})$ in SME corresponds to a

$$uniform$$

pdf over the set $\{\boldsymbol{x} : S_i(\boldsymbol{x}) \geq b_i,\ i = 1, \ldots, m\}$.

Lemma 3 For $\lambda = -\infty$ the optimal SME pdf $g(\boldsymbol{x})$ coincides with the classic IS

$$zero\ \ variance\ \ pdf.$$

Sampling from $g(\boldsymbol{x})$ is typically difficult. The only way is MCMC.

Pincus Theorem

Let $S(\boldsymbol{x})$ be a real-valued continuous function over closed bounded domain $\mathcal{X}$. Further assume that there is as a unique minimum point $\boldsymbol{x}^*$ over $\mathcal{X}$ at which $\min_{\boldsymbol{x}} S(\boldsymbol{x})$ attains. Then the coordinates of $x_k^*, \ k = 1, \ldots, n$ of $\boldsymbol{x}^*$ are given by

$$x_k^* = \lim_{\lambda \to \infty} \frac{\int_{\mathcal{X}} x_k exp{-\lambda S(\boldsymbol{x})} d\boldsymbol{x}}{\int_{\mathcal{X}} exp{-\lambda S(\boldsymbol{x})} d\boldsymbol{x}}, \ k = 1, \ldots, n.$$

and the optimal pdf is

$$g(\boldsymbol{x}) = \lim_{\lambda \to \infty} \frac{exp{-\lambda S(\boldsymbol{x})} d\boldsymbol{x}}{\int_{\mathcal{X}} exp{-\lambda S(\boldsymbol{x})} d\boldsymbol{x}}.$$

Pincus Theorem

The proof of the theorem is based on Laplace formula, which for sufficiently large λ can be written as

$$\int_{\mathcal{X}} x_k exp{-\lambda S(\boldsymbol{x})} d\boldsymbol{x} \approx x_k^* exp{-\lambda S(\boldsymbol{x}^*)}.$$

Multiple Rare-Events and MCE

The parametric versions is

$$p_i = \frac{\mathbb{E}_{\boldsymbol{u}}\left[X_i \exp\left\{-\sum_{i=1}^{m} \lambda C_i(\boldsymbol{X})\right\}\right]}{\mathbb{E}_{\boldsymbol{u}}\left[\exp\left\{-\sum_{i=1}^{m} \lambda C_i(\boldsymbol{X})\right\}\right]}, \; i = 1, \ldots, n. \qquad (13)$$

Since here only a sequence the vector $\{\boldsymbol{p}_t\}$ is updated (as in contrast to the standard CE, where a sequence of tuples $\{(\gamma_t, \boldsymbol{p}_t)\}$ is updated) we call the above new method, the SME (*semi-iterative CE)*.

Integer Programming: Counting

Our goal is to count the number of solutions $|\mathcal{X}^*|$ on the set $\mathcal{X}^*$ of constraints of integer programs defined as

$$\begin{aligned} &\textstyle\sum_{k=1}^{n} a_{ik} X_k = b_i, \ i = 1, \ldots, m_1, \\ &\textstyle\sum_{k=1}^{n} a_{jk} X_k \geq b_j, \ j = 1, \ldots, m_2, \\ &\boldsymbol{x} \geq \mathbf{0}, \ \ x_k \text{ integer } \forall k = 1, \ldots, n. \end{aligned} \tag{14}$$

Arguing as before we associate with the above set $\mathcal{X}^*$ the following rare event probability

$$\ell = \mathbb{P}_h\{\boldsymbol{X} \in \mathcal{X}^*\} = \mathbb{E}_{\boldsymbol{u}}\left[\prod_{i=1}^{m_1} I_{\left(\sum_{k=1}^{n} a_{ik} X_k = b_i\right)} \prod_{j=1}^{m_2} I_{\left(\sum_{k=1}^{n} a_{jk} X_k \geq b_j\right)}\right].$$

An Example: The SAT Problem

As a simple example consider the following SAT assignment

$$C_1 \wedge C_2 = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3).$$

This can be written either as

$$(x_1 + \bar{x}_2) \cdot (\bar{x}_1 + \bar{x}_2 + x_3)$$

or (in the form of constraints) as

$$x_1 + (1 - x_2) \geq 1$$

$$(1 - x_1) + (1 - x_2) + x_3 \geq 1.$$

An Example: The SAT Problem

In terms of ℓ we have $\ell = \mathbb{P}_{\boldsymbol{u}}(C_1 + C_2 = 2)$, where $C_1 = I_{\{X_1 - X_2 \geq 0\}}$ and $C_2 = I_{\{X_1 + X_2 - X_3 \leq 1\}}$, and each $x_1, x_2, x_3 \in \{0, 1\}$.

Integer Programming: Counting

Recall that the MCE program is

$$\min_g \mathbb{E}_g \mathcal{D}(g,h) = \min_g \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\, g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \min_g \mathbb{E}_g \left[\ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right]$$

$$\text{s.t.} \quad \mathbb{E}_g[\textstyle\sum_{i=1}^m C_i(\boldsymbol{X})] = m, \; \sum_{\boldsymbol{x}} g(\boldsymbol{x}) \mathrm{d}\, \boldsymbol{x} = 1. \tag{15}$$

The solution is

$$\frac{\mathbb{E}_{\boldsymbol{u}} \left[\sum_{i=1}^m C_i(\boldsymbol{X}) \exp\left\{ -\sum_{j=1}^m \lambda C_j(\boldsymbol{X}) \right\} \right]}{\mathbb{E}_{\boldsymbol{u}} \left[\exp\left\{ -\sum_{j=1}^m \lambda C_j(\boldsymbol{X}) \right\} \right]} = m$$

$$p_i = \frac{\mathbb{E}_{\boldsymbol{u}} \left[X_i \exp\{ -\sum_{i=1}^m \lambda C_i(\boldsymbol{X}) \} \right]}{\mathbb{E}_{\boldsymbol{u}} \left[\exp\{ -\sum_{i=1}^m \lambda C_i(\boldsymbol{X}) \} \right]}, \; i = 1, \ldots, n. \tag{16}$$

The Counting Algorithm

Associate with the set

$$\begin{aligned} &\sum_{k=1}^{n} a_{ik} X_k = b_i,\ i = 1, \ldots, m_1, \\ &\sum_{k=1}^{n} a_{jk} X_k \geq b_j,\ j = 1, \ldots, m_2, \\ &\boldsymbol{x} \geq \mathbf{0},\ \ x_k \text{ integer } \forall k = 1, \ldots, n. \end{aligned} \tag{17}$$

a *single-constrained* MinxEnt program

$$\begin{aligned} &\min_g \mathbb{E}_g \mathcal{D}(g, h) = \min_g \int \ln \frac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\, g(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \min_g \mathbb{E}_g \left[\ln \frac{g(\boldsymbol{X})}{h(\boldsymbol{X})} \right] \\ \text{s.t.}\quad &\mathbb{E}_g \left[\sum_{i=1}^{m} C_i(\boldsymbol{X})\right] = m \\ &\int g(\boldsymbol{x}) \mathrm{d}\,\boldsymbol{x} = 1. \end{aligned} \tag{18}$$

The Counting Algorithm

Recall that

$$\widehat{p}_{t,j} = \frac{\sum_{k=1}^{N} X_{kj} \, \exp\{-\lambda_t \, \mathcal{C}(\boldsymbol{X}_k)\} \, W(\boldsymbol{X}_k; \boldsymbol{u}, \widehat{\boldsymbol{p}}_{t-1})}{\sum_{k=1}^{N} \exp\{-\lambda_t \, \mathcal{C}(\boldsymbol{X}_k)\} \, W(\boldsymbol{X}_k; \boldsymbol{u}, \widehat{\boldsymbol{p}}_{t-1})}, \qquad (19)$$

where $\lambda = -\infty$

Algorithm 0.1 (SME Algorithm for Counting)

1. *Define* $\hat{\boldsymbol{p}}_0 = \boldsymbol{u}$. *Set* $t = 0$, $\lambda = M$, *say* $M = -100$.
2. $t \leftarrow t + 1$. *Generate a sample* $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ *from the density* $f(\boldsymbol{x}; \widehat{\boldsymbol{p}}_{t-1})$ *and compute* $\widehat{\boldsymbol{p}}_t$ *as above.*
3. *Smooth out the vector* $\widehat{\boldsymbol{p}}_t$ *according to*

$$\overline{\boldsymbol{p}}_t = \alpha \widehat{\boldsymbol{p}}_t + (1-\alpha)\widehat{\boldsymbol{p}}_{t-1}, \; 0 < \alpha < 1.$$

4. *If* $\mathcal{C}(\boldsymbol{X}) < m$, *reiterate from step 2. Else proceed with 5.*
5. *Estimate the counting quantity* $|\mathscr{X}^*|$ *as*

$$\widehat{|\mathscr{X}^*|} = \frac{1}{N} \sum_{k=1}^{N} I_{\{\mathcal{C}(\boldsymbol{X}_k)=m\}} \frac{1}{f(\boldsymbol{X}_k; \hat{\boldsymbol{p}}_t)}.$$

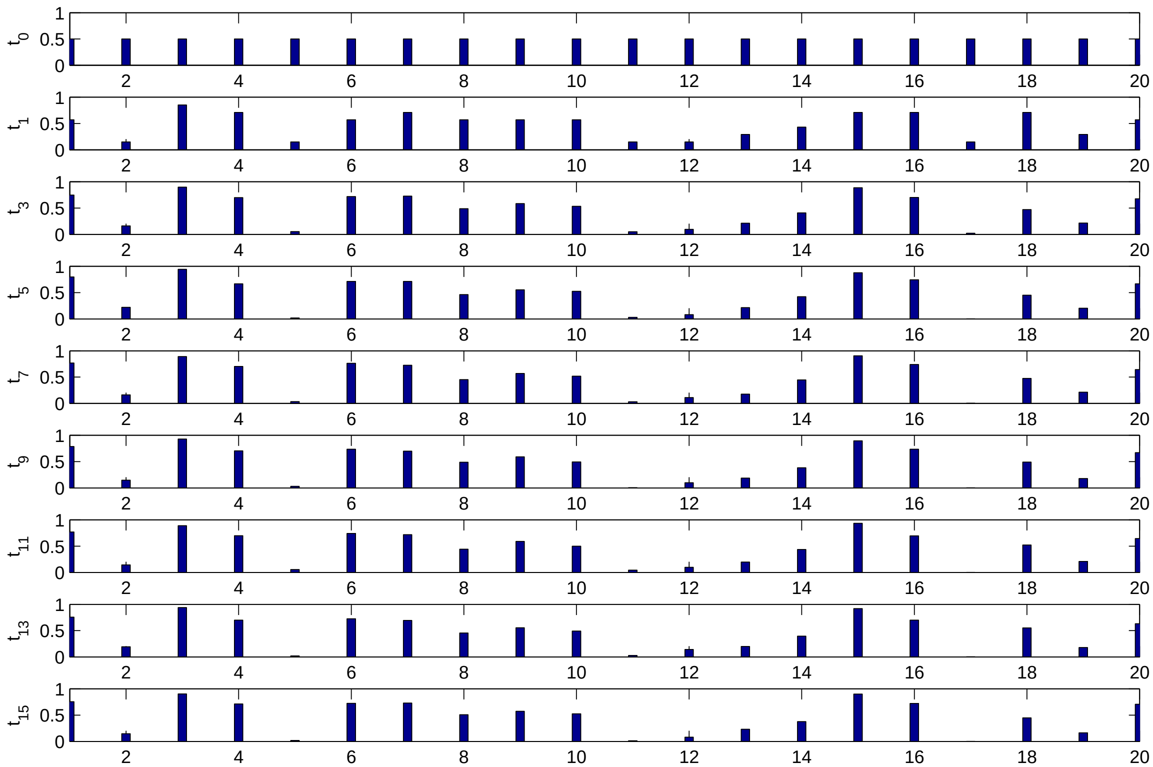
A Knapsack Problem

Performance of the CE Algorithm for the knapsack problem with the instance matrix $A = (20 \times 11)$ and $N = 10,000$. This problem was taken from the website http://elib.zib.de. Using full enumeration we found that the total number of multiple exrema is 612.

t	Mean	Max	Min	PV	RE	S	m	RD_c
0	639.6	943.7	419.4	0.00	0.225	6.93	10	55.73
1	619.2	697.6	564.8	0.03	0.072	5.78	11	0.02
2	630.8	706.5	557.0	0.07	0.059	5.18	11	0.03
3	628.1	698.1	533.2	0.08	0.083	4.95	11	0.03
4	573.7	671.2	504.9	0.09	0.083	4.88	11	0.06
5	599.3	719.6	525.7	0.09	0.100	4.72	11	0.03
6	576.9	646.4	508.0	0.09	0.071	4.76	11	0.06

A Knapsack Problem

A typical dynamics of the CE Algorithm for the knapsack problem with the instance matrix $A = (20 \times 11)$ and $N = 10,000$.



SUMMARY

Lemma 1: *We have to learn to live with uncertainty*

Theorem 1: *We can model the uncertainty*

Corollary 1: *We can make a living out of uncertainty*

THANK YOU