

# Counting algorithms and complexity. A brief overview of the field

Ioannis Nemparis<sup>1</sup>

<sup>1</sup>University of Athens

Advanced Issues of Algorithms and Complexity

## Outline

- 1 Introduction**
  - Warm up
- 2 Exploring the counting class**
  - complexity issues
  - algorithmic issues
- 3 Counting in FP**
  - Counting the spanning trees in a graph
  - Special cases
- 4 Correlation decay**
  - Definitions
  - Algorithmic implications
- 5 Holant problems**
- 6 Above  $\#P$**
- 7 Open questions**
- 8 Bibliography**

## Motivational quote

*“Man is fond of counting his troubles, but he does not count his joys.”*

— Fyodor Dostoevsky, *Notes from the Underground*

## Some definitions and basic results

- $\#P$ : The class of function problems of the form “compute  $f(x)$ ”, where  $f$  is the number of accepting paths of an NP machine.
- The class has complete problems and the one used as a point of reference for our reductions is the Permanent.
- Totally contrary to our intuition, Valiant, the one who defined the class, showed that there are decision problems in P, such that their counting version is  $\#P$  – complete. ( $\#matchings$ , Dimer’s problem).
- Toda showed that all the polynomial hierarchy  $PH \subseteq P^{\#P}$ . For the lowest 3 levels the result was already known thanks to Zachos.

## Any upper bound??

We know that  $NP \subseteq \#P$ . But do we have any knowledge of how high in the class hierarchy  $\#P$  is? It is trivial to show that  $\#P \subseteq PSPACE$ . (Run machine with every input string, reuse space and keep a counter).

## Computing #P – complete functions

#P-complete functions are quite high in the hierarchy. Can we compute anything in reasonable (polynomial time?). Once again like when tackling NP-completeness:

- We can approximate counting.
- We can tradeoff time with randomness.
- We can compute the function exactly in special cases.

## How good our algorithms can be?

Recall that  $NP \subseteq \#P$  and thus we can not expect to find an algorithm, with a better approximation factor for the  $\#P$ -complete problem than the one for its  $NP$ -complete problem counterpart. For example, minimum vertex cover is known not to be approximated in  $2 - \epsilon$  for reasonable assumptions. Thus, we can not expect to approximate the number of minimum vertex covers with  $2 - \epsilon$  factor.

# Counting Dichotomies

3 frameworks:

- Counting CSP problems
- Graph homomorphisms
- Holant Problems

All the above are models, which are used to obtain tractable criteria for the classification of the problems.



## Dichotomy

A theorem, classifying problems in  $\#P$  either in  $P$  or in  $\#P$ -complete would be ideal. However that is not possible due to Ladner's theorem.

**Ladner's theorem:** There is an infinite hierarchy of classes, strictly containing  $P$  and strictly contained in  $NP$ .

So, can we separate the difficult problems from the easier ones?

**Creignou-Hermann Theorem:** For any finite set  $S$  of Boolean predicates,  $\#CSP(S)$  is either solvable in Polynomial time or  $\#P$ -complete.

Boolean predicates can be boolean OR, At-Most-One, Not-All-Equal, XOR, etc. Notice that this does not contradict Ladner's theorem.

## Dichotomy 2

**Feder-Vardi Conjecture:** Any finite set  $S$  of predicates over any finite domain set  $D$ , the decision CSP problem  $\text{CSP}(S)$  is either in  $P$  or  $NP$ -complete. Analogously, for  $\#CSP$ .

The Feder-Vardi Conjecture is open, except for domain size 2 and 3.

## The “3” case. Bulatov theorem using the graph homomorphisms

Suppose  $A$  a  $\mathbb{R}^{N \times N}$  symmetric connection matrix with non-negative entries.

- If  $A$  is bipartite, then  $\text{EVAL}(A)$  is computed in polynomial time, if the rank of  $A$  is at most 2; otherwise  $\text{EVAL}(A)$  is  $\#P$ -complete.
- If  $A$  is not bipartite, then  $\text{EVAL}(A)$  is computed in polynomial time, if the rank of  $A$  is at most 1; otherwise  $\text{EVAL}(A)$  is  $\#P$ -complete.

Connection matrix is the adjacency matrix, corresponding to a graph. The  $\text{EVAL}$  predicate counts the number of structures, we want to count.

## Dichotomy 3

Cai greatly generalised all previous known results. He proved that for every symmetric complex valued matrix,  $\text{EVAL}(A)$  is either in  $P$  or  $\#P$ -hard.

Even better, the criterion that separates the two cases is decidable. Cai's paper (121 pages).

## Another criterion in the $\#CSP$ framework

Depending on function  $F$  one of the following 3 holds:

- $\#CSP(F)$  is in  $P$
- $\#CSP(F)$  is  $\#P$ -hard but belongs to  $P$  for planar graphs.
- $\#CSP(F)$  is  $\#P$ -hard even for planar graphs.

## Strange results

$F$  belongs in the second case, if and only if there is a holographic algorithm based on matchgates. **Holographic** algorithm is an algorithm using a holographic reduction. This type of reductions, between two computational problems, preserves the sum of solutions, without necessarily preserving correspondence between solutions.

That is right. The problems might not have matching solutions. On top of that the sum of solutions can be weighted.

**Matchgates** are a special class of two-qubit quantum logic gates. They have the interesting property, that circuits built only by matchgates, acting on neighbouring qubits, are efficiently simulatable classically. If however we remove the constrain of neighbouring qubits, they allow universal quantum computing.

## Inside #P

If someone tried to separate some problems in  $\#P$  from others (without following the results mentioned above), a good starting point might be to observe the **decision** counterpart of the problem. Thus we can have  $\#PE$ , as the class with easy decision problems and hard counting ones. Can this idea give us any more insight? Pagourtzis defined  $TotP$  as the class of functions that count the number of all computation paths of a poly-time nondeterministic Turing machine. In the same paper,  $\#Perfect\ Matchings$ ,  $\#DNF\text{-}Sat$ , and  $NonNegative\ Permanent$ , all problems in  $\#PE$  are shown to be in  $TotP$  too.

## $\#P$ – complete

We are talking for  $\#P$  – complete problems. You have probably noticed something strange.  $\#matchings$  is a complete problem for  $\#P$ , that is showed to be also in  $TotP$ , which is a subclass. Why do the classes not collapse? Our reductions are **Cook** reductions, not Karp and  $\#P$  is not downwards closed under this reduction. Consequently to prove some collapse we should either find a different type of reduction, or find a problem that is Karp reducible with problems both in  $\#P$  and in  $TotP$ . Both ways are open.



## Complexity results

- The sampling version of a  $NP$  – *complete* problem is at most as difficult as the  $\#P$  problem corresponding to it. Sampling means i.e picking a vertex cover U.A.R from all the vertex covers that exist.
- One  $\#P$  oracle call is enough to solve any problem in the polynomial hierarchy efficiently. [Toda]
- $PP$  class gives the most important bit of the  $\#P$  computation and  $oddP$  gives its least significant bit.
- Any problem in  $\#P$  there exists a randomized algorithm using oracle for SAT, that can be used to create an FPRAS [Stockmeyer]
- Any  $\#P$  – *complete* problem has either an FPRAS or it is inapproximable [Jerrum Valiant Vazirani]

## The first result

The first efficient algorithm for a counting problem is an FPRAS for the  $\#DNF$  problem. Notice that again the decision version is easy.  
Algorithm:

---

Choose a random clause  $\theta_i$ , with probability equal to  $\frac{w_i}{Z}$ .

Choose a random assignment  $\sigma$  satisfying  $\theta_i$ .

Calculate  $S = \frac{Z}{N(\sigma)}$ , where  $N(\sigma)$  is the number of clauses satisfied by  $\sigma$ .

Repeat steps 1-3  $t := \frac{4M}{\epsilon^2}$  times and output the mean.

---

$w_i = p^{p_i} (1-p)^{n_i}$  with  $p_i$   $n_i$  the number of positive and negative literals of  $\theta_i$  and  $Z = \sum w_i$ .

## THE result

Although there were not many, all FPRASes up until recently were for  $\#PE$  problems. At the mid 80s the first FPTAS also appeared for a  $\#P$  – *complete* problem (DNF) [Ajtai and Wigderson] and the second one in mid 90s [Luby and Velickovic]. After another decade Weitz in his doctorate thesis greatly developed an existing method, to create FPTASes for some problems, called “correlation decay” in statistical physics. We will talk about it a bit later. The big result however came up on 2011 by [Gopalan, Klivans, Meka, Stefankovic, Vempala, Vigoda].

**There exists an FPTAS for the  $\#Knapsack$  problem.** For the first time, there exists an FPTAS for counting solutions of an  $NP$  – *complete* problem. Counter-intuitively, the trivial bound of the knapsack inapproximability is reached for the counting version. The algorithm works also for the general case where there are many identical items.

## The algorithm in a few words

The algorithm greatly resembles the FPTAS of the knapsack problem. The max of the dynamic programming is replaced by the addition of the subproblems. Now each value of the array holds a certain value of the capacity of the knapsack (minimum capacity to create  $j$  solutions with the  $i$  items =  $A[i][j]$ ). More precisely, the 2 indices of the matrix are: a) The prefix of items. b) Number of solutions. Been inspired again by the decision version algorithm, we scale down the capacity and the weight by some factor. OPT value for the new instance is a very good approximation for the initial instance. Using a similar algorithm, we can have an FPRAS using rejection sampling, for sampling solutions for the initial problem [Dyer].

**Rejection sampling** is based on the observation, that to sample a random variable one can sample uniformly from the region under the graph of its density function.

## The algorithm in a few words 2. Derandomizing

Instead of having every separate possible capacity up until  $C$ , we can have only  $\log C$  many capacities which will create a geometric series back in the original instance (think of it). With the specific scaledown suggested in the paper, we succeed in being arbitrarily close to the OPT value of the counting function depending polynomially in  $n$  and  $\frac{1}{\epsilon}$ . It is almost unbelievable, that nobody thought of that earlier, although the analysis is a bit more delicate than the original algorithm for the NP-complete problem.

## What counting algorithms we have in FP?

Probably you are asking yourselves if there is any natural problem, whose solutions are counted optimally in polynomial time. The answer is yes. In the following sections there are some such problems.

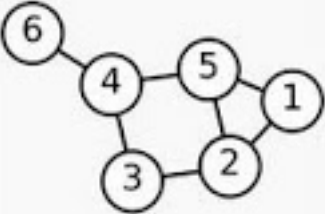
## Basic idea

The **laplacian** matrix of a graph is a matrix with equal dimension and indices with the adjacency matrix of the graph. Every value contains:  $f(n) =$

$$\begin{cases} \text{deg}(u_i) & \text{if } u_i \text{ on primary diagonal} \\ -1 & \text{if } u_i \text{ } u_j \text{ adjacent} \\ 0 & \text{otherwise} \end{cases}$$

The determinant of the laplacian matrix is equal to the number of spanning trees. This is also known as the Kirchoff's theorem, in graph theory. Computing the determinant is in P.

## A Laplacian matrix

Labeled graph	Laplacian matrix
	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$



## Counting in specific graphs

*“A mathematic lecture without a proof is like a movie without a love scene.”*

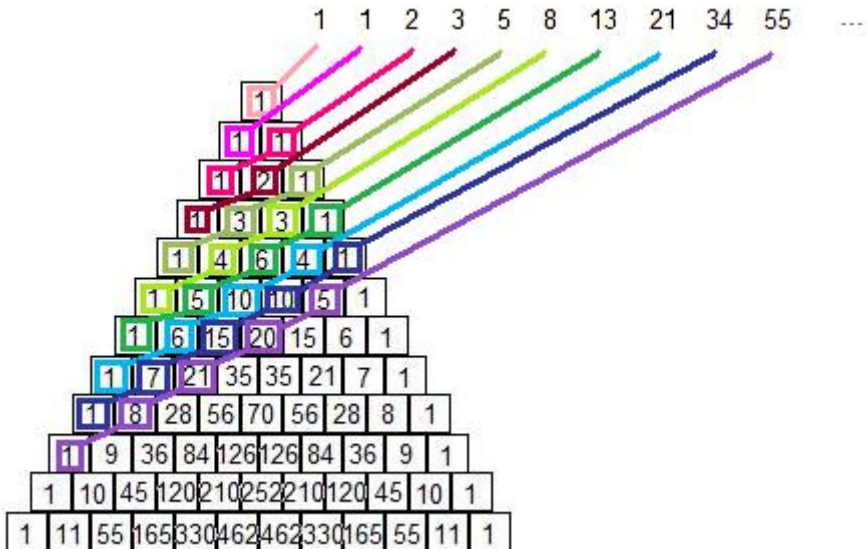
— Hendrik Lenstra

In general, we can have a polynomial counting algorithm for special instances of a hard problem. Recall the Dichotomy theorem. Usually planar graphs are good candidates. Here, we will present an easy algorithm for counting edge covers in path graphs and ring graphs. Proof on board.

## Special cases 2 and a small extra result

We can also count fast the edge covers in trees. We will not show it here. However, it is shown, that counting is difficult, even for 3-regular graphs. In the specific case of the counting in path graphs, we have made an extra observation not yet known. More specifically, it is shown that we can compute efficiently the number of edge covers. But is there a way to count the exact number of covers of every size that form this computation? It is and it corresponds to the values summing to the fibonacci number in the laplace triangle (diagonals), with bigger size covers being the leftmost member value of the diagonal.

# Pascal Triangle



## Time and Space mixing in Spin systems

- Spin systems are models in statistical physics. We can think of them as graphs.
- A configuration is an assignment of a specific spin (finitely many spins) to its vertices.
- The sites (vertices) interact depending on the system. Different configurations have different likelihoods.
- The interactions can form a distribution over configurations of every finite subset of sites. It is called the Gibbs distribution and it is the equilibrium of the subset.

## Izing Model and Gibbs measure

In this specific model, we have only 2 possible spins, and a  $\beta$  parameter, which is the inverse of the temperature. So the Gibbs measure is given by the following formula for a configuration  $\sigma$ :

$$(\mu_G(\sigma)) \propto \exp(\beta \sum_{xy \in E} \sigma_x \sigma_y)$$

### Observations

- Higher probabilities for aligned spins.
- High temperature  $\rightarrow$  almost independent spins.
- Low temperature  $\rightarrow$  a global order.

# Glauber Dynamics

The Glauber dynamics is a Markov chain on the set of spin configurations of a finite graph.

## Glauber dynamics

- At each step, pick a vertex u.a.r and replace it with a random spin conditional on all the neighbour spins.

The stationary distribution of the Glauber dynamics is  $\mu_G$  as above.

## Why the Glauber metric is important?

- 1 It is a tool for analysing Markov chain Monte Carlo algorithms, other than random walks.
- 2 It exhibits the evolution of the system towards the equilibrium.

The aim is determining the mixing time (number of steps till the Glauber dynamics is close to stationary).

## Importance in computational complexity

### Big result

- On finite  $n$ -vertex cubes in the 2-dimensional lattice, there exists a  $\beta_c$  such that for smaller  $\beta$  mixing is performed in  $O(\log n)$ , but for greater ones in  $\exp(\Omega(\sqrt{n}))$ .

This result shows a connection between phase transition and computational complexity. It reminds of the  $k$  – SAT problem when there is a transition in difficulty for  $k=3$ .



## Time and Space mixing

### Time mixing

- Number of steps, till the Glauber dynamics is close to stationary.

### Space mixing

- The degree of correlation between spins of vertices according to distance and the rate it changes.

We call fast time mixing, if the required time of the chain to reach its steady state is a small polynomial and fast spacial mixing, if correlation decays exponentially according to the distance.

## Algorithmic implications

The above notions can be used to construct a “local” algorithm as the problem seems to be of “local” nature, if the correlation decays exponentially fast. In other words, the local algorithm is the Glauber dynamics and the problem is sampling from the Gibbs measure. The Glauber dynamics can solve the problem, if the correlations decay fast enough and spins can be regarded as independent. Many known problems exhibit such “localities” as edge cover, independent set.

## Holographic algorithms

The holant problem are another way to model constraint problems. They are considered more general than the #CSP setting and take their name from the Holant sum introduced by Valiant. As with other reduction, a holographic reduction does not, yield a polynomial time algorithm. In order to get a polynomial time algorithm, the problem being reduced to must also have a polynomial time algorithm. Valiant's original application of holographic algorithms used a holographic reduction to a problem where every constraint is realizable by matchgates, which he had just proved is tractable by a further reduction to counting the number of perfect matchings in a planar graph. The latter problem is tractable by the FKT (Fisher, Kasteleyn, and Temperley) algorithm. Valiant found such problems. The #7PI-Rtw-Mon-3CNF and #7PI-3/2Bip-VC. After some time Cai generalised the result and showed that if the number parameter is a mersenne number the reduction can be performed.

## Beyond $\#P$ in counting complexity

The question arises quite naturally. Is there any counting class, containing natural problems, which is above  $\#P$ ? The answer is yes. There is a result, in a very interesting paper, showing that the very well known problem of N-queens has a counting version that is above  $\#P$ . The class is not given a name to our knowledge. The paper itself aims in counting the number of complete mappings in an abelian group. The two problems are shown to be equivalent.

## Food for thought I

- 1 Find some structural attributes of this strange class above  $\#P$ .
- 2 Any other candidate problems for using the correlation decay method to get some FPTAS. It should exhibit some locality.
- 3 The N-queens problem belongs in TFNP. Does the result above tell us anything about counting versions of TFNP problems?
- 4 This “dynamic” scaledown programming can be used in any other problem?
- 5 There are very few problems in NP having an FPTAS so we can expect even less in  $\#P$ . How about PTAS?
- 6 Can we extend the inapproximability result of  $\#$ edge cover?
- 7  $P^{\#P} = P^{\#BQP}$ . What does this imply for quantum counting?

## Food for thought II

- 8 Hamming distance of compressed texts is  $\#P$  – *complete* can we find an FPTAS?
- 9 Longest common subsequence in compressed texts?
- 10 Modelling problems as multi spin systems in order to use the Weitz method.
- 11 Counting NPI problems? Most problems seem to not have counting versions. Why that happens? What about the rest?



Jin-Yi Cai, Xi Chen, and Pinyan Lu.

Graph homomorphisms with complex values: A dichotomy theorem.

*SIAM J. Comput.*, 42(3):924–1029, 2013.



Jin-Yi Cai, Pinyan Lu, and Mingji Xia.

Holographic reduction, interpolation and hardness.

*Computational Complexity*, 21(4):573–604, 2012.



Martin E. Dyer, Alistair Sinclair, Eric Vigoda, and Dror Weitz.

Mixing in time and space for lattice spin systems: A combinatorial view.

*Random Struct. Algorithms*, 24(4):461–479, 2004.



Jieh Hsiang, D. Frank Hsu, and Yuh-Pyng Shieh.

On the hardness of counting problems of complete mappings.

*Discrete Mathematics*, 277(1-3):87–100, 2004.



Richard E. Ladner, Nancy A. Lynch, and Alan L. Selman.

Comparisons of polynomial-time reducibilities.

In *STOC*, pages 110–121, 1974.



Jian Li and Tianlin Shi.

A fully polynomial-time approximation scheme for approximating a sum of random variables.

*Oper. Res. Lett.*, 42(3):197–202, 2014.



Liang Li, Pinyan Lu, and Yitong Yin.

Correlation decay up to uniqueness in spin systems.

In *SODA*, pages 67–84, 2013.



Yury Lifshits.

Processing compressed texts: a tractability border.

In *Proc. CPM 2007*, pages 228–240. Springer, 2007.



Chengyu Lin, Jingcheng Liu, and Pinyan Lu.

A simple fptas for counting edge covers.

In *SODA*, pages 341–348, 2014.





Daniel Stefankovic, Santosh Vempala, and Eric Vigoda.

A deterministic polynomial-time approximation scheme for counting knapsack solutions.

*SIAM J. Comput.*, 41(2):356–366, 2012.



Leslie G. Valiant.

The complexity of computing the permanent.

*Theor. Comput. Sci.*, 8:189–201, 1979.



Dror Weitz.

Counting independent sets up to the tree threshold.

In *STOC*, pages 140–149, 2006.



Jin yi Cai and Vinay Choudhary.

Valiant's holant theorem and matchgate tensors.

*Theor. Comput. Sci.*, 384(1):22–32, 2007.