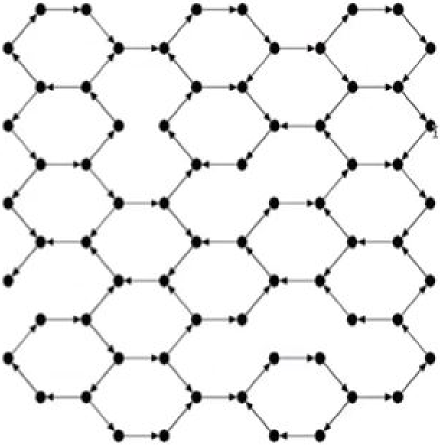
#Pl-3-NAE-Ice

Input: A planar graph $G = (V, E)$ of maximum degree 3.

Output: The number of orientations such that no node has all incident edges directed toward it or all incident edges directed away from it.

- #Pl-3-NAE-Ice is the problem of counting the number of no-sink-no-source orientations.
- We assume every node has degree 2 or 3, since a node of degree 1 will preclude such an orientation.

An instance of #PL-3-NAE-ICE with a valid orientation



A holographic algorithm for #PL-3-NAE-ICE Step 1

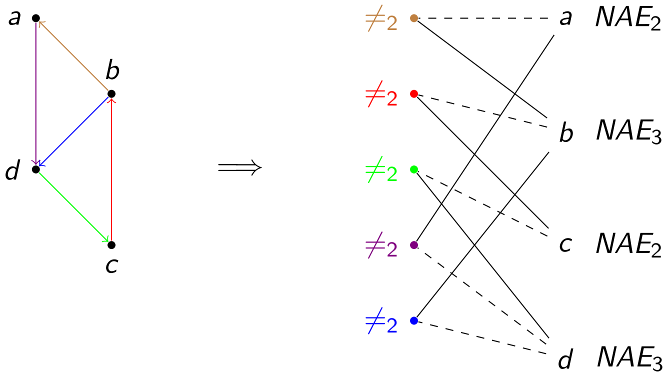
Let $G = (V, E)$ be an input to the problem.

To solve the problem by a holographic algorithm with matchgates, we make the following transformations:

1. We design a signature grid $\Omega = (G', \pi)$ based on G:
   - We attach to each node of degree 3 a *Not-All-Equal* (or *NAE*) gate of arity 3, i.e. the signature $[0, 1, 1, 0]$, as a contravariant tensor (with only left dangling edges).
   - For any node of degree 2 we use a *NAE* gate of arity 2, i.e. a binary *Disequality* signature $(\neq_2) = [0, 1, 0]$, also as a contravariant tensor.
   - For each edge in E we use a binary *Disequality* signature $(\neq_2) = [0, 1, 0]$ as a covariant tensor (with only right dangling edges).

Example of Step 1 (edge-vertex incidence graph)

An orientation of G An assignment σ on the edges of $\Omega = (G', \pi)$



- If G is planar, then G' is also planar.
- Here, a dashed edge e in Ω denotes that $\sigma(e) = 0$, whereas a solid edge e denotes that $\sigma(e) = 1$.

Step 2

2. We replace each signature in Ω by a signature that is realizable as a matchgate signature.
   - The replacement of signatures in Ω is done by a holographic transformation defined by a 2×2 matrix.
   - We use the matrix $H = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ with $H^{-1} = 2H = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$.

Step 2

We replace

- every contravariant signature $[0,1,1,0]$ by the signature

$$(H^{-1})^{\otimes 3} \cdot [0,1,1,0] = [6,0,-2,0]$$

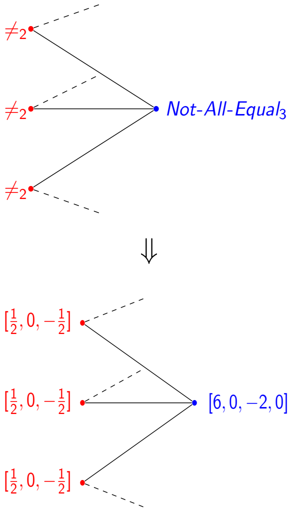
- every contravariant signature $[0,1,0]$ by the signature

$$(H^{-1})^{\otimes 2} \cdot [0,1,0] = [2,0,-2]$$

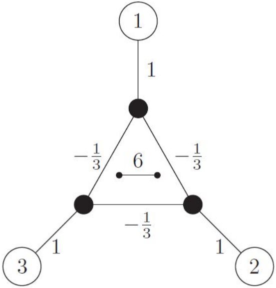
- every covariant signature $[0,1,0]$ by the signature

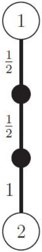
$$[0,1,0] \cdot H^{\otimes 2} = \frac{1}{2}[1,0,-1]$$

A part of Step 2 (holographic transformation)



The signatures $[6, 0, -2, 0]$, $[2, 0, -2]$, $[\frac{1}{2}, 0, -\frac{1}{2}]$ are all realizable as matchgate signatures.

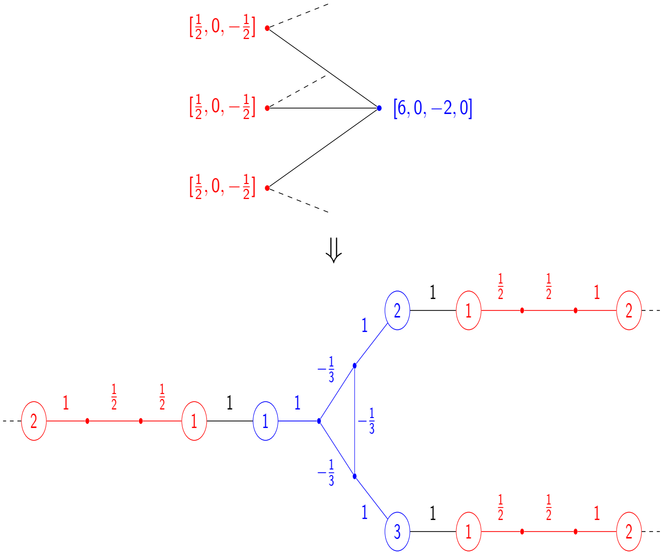




Step 3

3. We obtain a weighted graph G'' as follows.
   - We replace each of the signatures $[6, 0, -2, 0]$, $[2, 0, -2]$ and $[\frac{1}{2}, 0, -\frac{1}{2}]$ in the new signature grid by its corresponding matchgate.
   - The edges that connect the matchgates to each other are of weight 1.

A part of Step 3 (matchgates)



The algorithm for #Pl-3-NAE-Ice revisited

The algorithm consists of the following three reductions and Kasteleyn's algorithm.

1. #Pl-3-NAE-Ice $\leq_T$ Holant$([0,1,0] \mid [0,1,0],[0,1,1,0])$.
2. Holant$([0,1,0] \mid [0,1,0],[0,1,1,0]) \equiv_T$

   Holant$(\frac{1}{2}[1,0,-1] \mid [2,0,-2],[6,0,-2,0])$.
3. Holant$(\frac{1}{2}[1,0,-1] \mid [2,0,-2],[6,0,-2,0]) \leq_T$ #PerfMatch in planar graphs.

Other problems with polynomial-time algorithms

- #Pl-3-NAE-Sat: on input a planar 3-NAE formula ϕ, count the satisfying assignments of ϕ.
- Pl-Node-Bipartition: on input a planar graph G of max degree 3, compute the minimum cardinality of a subset $S \subset V$ such that $G \setminus S$ is a bipartite graph.
- $\#_7$Pl-Rtw-Mon-3CNF: on input a planar, read-twice, monotone, 3-CNF formula, compute the number of satisfying assignments modulo 7.

Note that $\oplus$Pl-Rtw-Mon-3CNF is $\oplus$P-complete.

Simultaneous Realizability Problem

2010: Cai & Lu proved that the following problem can be solved in polynomial time in their paper "Holographic algorithms: From art to science".

Simultaneous Realizability Problem

Input: A set of symmetric signatures for generators or/and recognizers.

Output: A holographic transformation to matchgate signatures, if any exists; 'NO', otherwise.

Holographic Algorithms with matchgates capture precisely tractable planar #CSP

Theorem (Cai & Fu 2016)

Consider the class of Boolean #CSP with local constraints being not necessarily symmetric, complex-valued functions. Every problem in this class belongs to one of the following three categories according to $\mathcal{F}$.

1. *those which are tractable (polynomial-time computable) on general graphs,*
2. *those which are #P-hard on general graphs but tractable on planar graphs,*
3. *those which are #P-hard even on planar graphs.*

Moreover, problems in category (2) are tractable on planar graphs precisely by holographic algorithms with matchgates.

- Cai, Lu and Xia (2010) had shown the same theorem for symmetric real-valued functions.
- Huo and Williams (2013) had shown the same theorem for symmetric complex-valued functions.

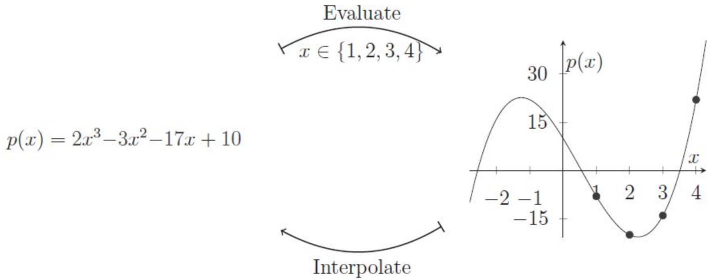
Overview

Polynomial interpolation

- Polynomial interpolation is a powerful technique to prove #P-hardness for counting problems under Turing reductions.

Polynomial interpolation

- Polynomial interpolation is a powerful technique to prove #P-hardness for counting problems under Turing reductions.
- Polynomial interpolation is the inverse of evaluation.



Proposition

#PerfectMatchings $\leq_T$ #Matchings.

Proof. Let $G = (V, E)$.

- Let m_k be the number of matchings in G that omit k vertices.

Proposition

#PerfectMatchings $\leq_T$ #Matchings.

Proof. Let $G = (V, E)$.

- Let m_k be the number of matchings in G that omit k vertices.
- Then, m_0 is the number of perfect matchings in G.

Proposition

#PerfectMatchings $\leq_T$ #Matchings.

Proof. Let $G = (V, E)$.

- Let m_k be the number of matchings in G that omit k vertices.
- Then, m_0 is the number of perfect matchings in G.
- Let the matching polynomial be $P(x) = \sum_k m_k x^k$.

Proposition

#PerfectMatchings $\leq_T$ #Matchings.

Proof. Let $G = (V, E)$.

- Let m_k be the number of matchings in G that omit k vertices.
- Then, m_0 is the number of perfect matchings in G.
- Let the matching polynomial be $P(x) = \sum_k m_k x^k$.
- We will use polynomial interpolation to determine all the coefficients m_k, $0 \leq k \leq n$.

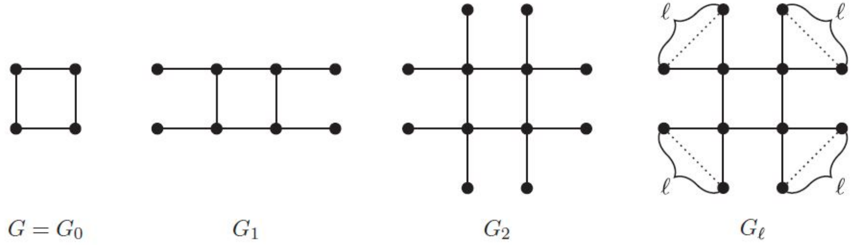
Proposition

#PerfectMatchings $\leq_T$ #Matchings.

Proof. Let $G = (V, E)$.

- Let m_k be the number of matchings in G that omit k vertices.
- Then, m_0 is the number of perfect matchings in G.
- Let the matching polynomial be $P(x) = \sum_k m_k x^k$.
- We will use polynomial interpolation to determine all the coefficients m_k, $0 \leq k \leq n$.
- We will do this by making $n + 1$ oracle calls to the problem of counting (all) matchings.

Proof cont. For every $0 \leq l \leq n$, we construct a graph G_l as below.



Proof cont. Let m_k be the number of matchings in G that omit k vertices. Then the number of matchings in G_l can be expressed as follows.

$$\sum_{k=0}^{n}(l+1)^k m_k = \#\text{Matchings}(G_l).$$

$$\text{So,}\quad P(l+1) = \#\text{Matchings}(G_l).$$

- Each matching in G that omits k vertices can be extended to a matching in G_l in $(l+1)^k$ different ways!
- Each matching in G_l is obtained uniquely this way from a matching of G.

Proof cont. We collect these equations to form the following linear system.

$$\begin{bmatrix} (0+1)^0 & (0+1)^1 & (0+1)^2 & \dots & (0+1)^n \\ (1+1)^0 & (1+1)^1 & (1+1)^2 & \dots & (1+1)^n \\ . & . & . & \dots & . \\ . & . & . & \dots & . \\ . & . & . & \dots & . \\ (n+1)^0 & (n+1)^1 & (n+1)^2 & \dots & (n+1)^n \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ . \\ . \\ . \\ m_n \end{bmatrix} = \begin{bmatrix} \#\textsc{Matchings}(G_0) \\ \#\textsc{Matchings}(G_1) \\ . \\ . \\ . \\ \#\textsc{Matchings}(G_n) \end{bmatrix}.$$

- The linear system is of the form $\mathbf{Vm} = \mathbf{M}$.
- The coefficient matrix $\mathbf{V}$ is a Vandermonde matrix (each row is a geometric progression).
- It is invertible iff the values $(l+1)$ are all distinct, which is true.
- The matrix on the RHS can be computed by $n+1$ oracle calls.
- So, we can solve the system and find $\mathbf{m}$.

Overview

Ladner's Theorem

Ladner's Theorem

If P $\neq$ NP, then there exists a language $L \in$ NP that is neither in P nor NP-complete.

Corollary 1

If P $\neq$ NP, there is an infinite hierarchy of separate complexity classes that lie between P and NP.

Corollary 2

If FP $\neq$ #P, there is an infinite hierarchy of separate complexity classes between FP and #P.

- All the examples of such intermediate problems are based on diagonalization constructions and are very artificial.

- Since the concept of a ‘natural’ problem is somewhat ambiguous, a possible research direction is to pursue dichotomy results for wide classes of problems.

Dichotomy theorems for classes of decision problems

For some broad classes of problems, dichotomy theorems do exist:

1. Schaefer's theorem (1978) is a dichotomy result for the Generalized Satisfiability problem.
2. Hell and Nešetřil (1990) proved a dichotomy theorem for the H-Coloring problem.
3. Vardi and Feder (1993) posed the CSP dichotomy conjecture.
4. Bulatov and Zhuk (2017) independently confirmed the CSP dichotomy conjecture.

Dichotomy theorems for counting graph homomorphisms

- Dyer and Greenhill (2000) proved that, for any undirected unweighted graph H, the corresponding #H-COLORING is either in FP or #P-complete.

Dichotomy theorems for counting graph homomorphisms

- Dyer and Greenhill (2000) proved that, for any undirected unweighted graph H, the corresponding #H-COLORING is either in FP or #P-complete.
- Bulatov and Grohe (2005) gave a classification for the problem of computing the weighted sum of homomorphisms to a graph H with non-negative real weights.

Dichotomy theorems for counting graph homomorphisms

- Dyer and Greenhill (2000) proved that, for any undirected unweighted graph H, the corresponding #H-COLORING is either in FP or #P-complete.
- Bulatov and Grohe (2005) gave a classification for the problem of computing the weighted sum of homomorphisms to a graph H with non-negative real weights.
- Goldberg, Grohe, Jerrum and Thurley (2010) gave a dichotomy result for weighted graphs H, where the weights are real numbers.

Dichotomy theorems for counting graph homomorphisms

- Dyer and Greenhill (2000) proved that, for any undirected unweighted graph H, the corresponding #H-COLORING is either in FP or #P-complete.
- Bulatov and Grohe (2005) gave a classification for the problem of computing the weighted sum of homomorphisms to a graph H with non-negative real weights.
- Goldberg, Grohe, Jerrum and Thurley (2010) gave a dichotomy result for weighted graphs H, where the weights are real numbers.
- Cai, Chen and Lu (2010) extended the dichotomy to complex weights.

Dichotomy theorems for counting graph homomorphisms

- Dyer and Greenhill (2000) proved that, for any undirected unweighted graph H, the corresponding #H-COLORING is either in FP or #P-complete.
- Bulatov and Grohe (2005) gave a classification for the problem of computing the weighted sum of homomorphisms to a graph H with non-negative real weights.
- Goldberg, Grohe, Jerrum and Thurley (2010) gave a dichotomy result for weighted graphs H, where the weights are real numbers.
- Cai, Chen and Lu (2010) extended the dichotomy to complex weights.

The dichotomy criterion is explicit: Given H, we can decide whether $Z_H(G)$ is in FP or #P-hard.

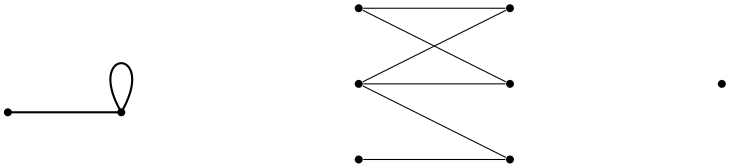
The problem of deciding whether a homomorphism exists

- Given graphs G and H, a homomorphism from G to H is a function $f : V(G) \to V(H)$ such that every edge $(u, v) \in E(G)$ is mapped to an edge $(f(u), f(v)) \in E(H)$.
- Decision problem: Given G as input, is there a homomorphism from G to H?
- We call this problem the GRAPH HOMOMORPHISM problem or the H-COLORING problem.
- G is the input graph, whereas H is fixed, so part of the description of the problem.

Theorem (Hell & Nešetřil 1990)

Let H be a fixed graph. The H-Coloring *problem in in P, if H either has a loop (self-loop) or is bipartite.*
Otherwise, H-Coloring *is NP-complete.*

Easy cases:



The problem of counting graph homomorphisms

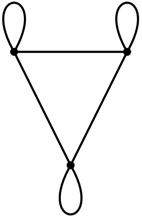
Theorem (Dyer & Greenhill 2000)

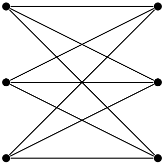
Let H be a fixed graph. The #H-COLORINGS *problem is in FP, if every connected component of H is*

1. *either a complete graph with all loops present*
2. *or a complete bipartite graph with no loops present.*

Otherwise, #H-COLORINGS *is #P-complete.*

Easy cases:





Suppose that H is a complete graph with all loops present, or a complete bipartite graph with no loops. Then, #H-COLORINGS can be solved in polynomial time.

Proof.

1. If H is an isolated vertex without a loop, then $Z_H(G) = 0$, unless G is a collection of isolated vertices, in which case $Z_H(G) = 1$.

Suppose that H is a complete graph with all loops present, or a complete bipartite graph with no loops. Then, #H-COLORINGS can be solved in polynomial time.

Proof.

1. If H is an isolated vertex without a loop, then $Z_H(G) = 0$, unless G is a collection of isolated vertices, in which case $Z_H(G) = 1$.
2. If H is the complete graph on k vertices with all loops present, then if G has n vertices, it holds that $Z_H(G) = k^n$.

Suppose that H is a complete graph with all loops present, or a complete bipartite graph with no loops. Then, #H-COLORINGS can be solved in polynomial time.

Proof.

1. If H is an isolated vertex without a loop, then $Z_H(G) = 0$, unless G is a collection of isolated vertices, in which case $Z_H(G) = 1$.
2. If H is the complete graph on k vertices with all loops present, then if G has n vertices, it holds that $Z_H(G) = k^n$.
3. Suppose H is the complete bipartite graph with vertex bipartition $C_1 \cup C_2$, $|C_i| = k_i$, $i = 1, 2$, and with no loops.
   - If G is not bipartite, then $Z_H(G) = 0$.
   - If G is bipartite with vertex bipartition $V_1 \cup V_2$, $|V_i| = n_i$, $i = 1, 2$, then

$$Z_H(G) = k_1^{n_1} \cdot k_2^{n_2} + k_1^{n_2} \cdot k_2^{n_1}.$$

□