Complexity of #CSP with Complex Weights

Xi Chen
Joint work with Jin-Yi Cai

Columbia University

Nov 3, 2011

Constraint Satisfaction Problem

- Let $D = \{1, 2, \ldots, d\}$ be a domain.
- A language is a finite set of predicates $\Gamma = \{\Theta_1, \ldots, \Theta_h\}$.
- An instance of $\#\mathrm{CSP}(\Gamma)$ consists of a set of variables $x_1, \ldots, x_n$ and a set of constraints from Γ, each applied to a subset of variables. It defines an n-ary relation R, where $(x_1, \ldots, x_n) \in R$ if all constraints are satisfied.

Examples

- 3-coloring: $D = \{1,2,3\}$ and $\Gamma = \{\Theta\}$, where

$$\Theta = \Big\{(i,j) : i,j \in D \text{ and } i \neq j\Big\}.$$

- Independent set: $D = \{1,2\}$ and $\Gamma = \{\Theta\}$, where

$$\Theta = \Big\{(1,1),(1,2),(2,1)\Big\}.$$

- 2SAT: $D = \{0,1\}$ and

$$\Gamma = \Big\{x_1 \vee x_2, \neg x_1 \vee x_2, x_1 \vee \neg x_2, \neg x_1 \vee \neg x_2\Big\}$$

- 3SAT ...

Constraint Satisfaction Problem

One of the most important classes of problems in TCS:

- Decision: whether a solution exists?

  [Schaefer 78, Hell and Nesetril 90, Feder and Vardi 98, Bulatov 06, Kun and Szegedy 09, . . .]

  The CSP dichotomy conjecture of Feder and Vardi is open

- Optimization: satisfy as many constraints as possible

  [Hastad 01, Khot, Kindler, Mossel and O'Donnell 07, Austrin and Mossel 08, Raghavendra 08, Dinur, Mossel and Regev 09, Tulsiani 09, Raghavendra and Steurer 09, . . .]

- Counting: count the solutions

Unweighted Counting CSP (#CSP)

- Let $D = \{1, 2, \ldots, d\}$ be a domain.
- A language is a finite set of predicates $\Gamma = \{\Theta_1, \ldots, \Theta_h\}$.
- An instance of #CSP(Γ) consists of a set of variables $x_1, \ldots, x_n$ and a set of constraints from Γ, each applied to a subset of variables. It defines an n-ary relation R, where $(x_1, \ldots, x_n) \in R$ if all constraints are satisfied.
- Compute $|R|$.

Examples

- Counting 3-colorings: $D = \{1, 2, 3\}$ and $\Gamma = \{\Theta\}$, where

$$\Theta = \{(i, j) : i, j \in D \text{ and } i \neq j\}.$$

- Counting independent sets: $D = \{1, 2\}$ and $\Gamma = \{\Theta\}$, where

$$\Theta = \{(1, 1), (1, 2), (2, 1)\}.$$

- #2SAT: $D = \{0, 1\}$ and

$$\Gamma = \{x_1 \vee x_2, \neg x_1 \vee x_2, x_1 \vee \neg x_2, \neg x_1 \vee \neg x_2\}$$

- #3SAT . . .

Weighted #CSP

- A weighted constraint language $\mathcal{L} = \{f_1, \ldots, f_h\}$:

$$f_i : D^{r_i} \to \mathbb{C}$$

- An instance of $\#\mathrm{CSP}(\mathcal{L})$ consists of variables $x_1, \ldots, x_n$ over D and a finite set of constraint functions from $\mathcal{L}$, each applied to a subset of these variables. It defines a new n-ary function F: for any assignment $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$, $F(\mathbf{x})$ is the product of the constraint function evaluations.
- Given an input instance F, compute:

$$\sum_{\mathbf{x} \in D^n} F(\mathbf{x})$$

A Complexity Dichotomy

Theorem (Main)

Given any domain set D and any finite set $\mathcal{L}$ of complex-valued functions, $\#\mathrm{CSP}(\mathcal{L})$ is either in polynomial time or #P-hard.

Tractability Criteria

If $\mathcal{L}$ satisfies the following three conditions, we give a polynomial time algorithm for #CSP($\mathcal{L}$); otherwise we show it is #P-hard.

1. the Block Orthogonality condition
2. the Mal'tsev condition
3. the Type Partition condition

Some Notation

- Let $F : D^n \to \mathbb{C}$ be the function defined by an input instance
  For each $t \in [n]$, let $F^{[t]} : D^t \to \mathbb{C}$ be

$$F^{[t]}(x_1, \ldots, x_t) = \sum_{x_{t+1},\ldots,x_n} F(x_1, \ldots, x_t, x_{t+1}, \ldots, x_n)$$

- Consider $F^{[t]}$ as a $d^{t-1} \times d$ matrix:
  1. Rows: $\mathbf{x} = (x_1, \ldots, x_{t-1}) \in D^{t-1}$ and columns: $a \in D$
  2. The $(\mathbf{x}, a)$th entry of the matrix is $F^{[t]}(\mathbf{x}, a)$
  3. Use $F^{[t]}(\mathbf{x}, *)$ to denote the d-dim row vector indexed by $\mathbf{x}$

With a Little Help from an Oracle

An oracle that provides information about $F^{[2]}, \ldots, F^{[n]}$:

1. send any $\mathbf{x} \in D^{t-1}$ to the oracle
2. return a vector $\mathbf{v}$ that is linearly dependent with $F^{[t]}(\mathbf{x}, *)$:
   - $\mathbf{v} = \mathbf{0}$ if $F^{[t]}(\mathbf{x}, *) = \mathbf{0}$;
   - otherwise, $\mathbf{v}$ is normalized: its first nonzero entry $= 1$.

A Framework for Solving #CSP($\mathcal{L}$)

To compute $F^{[1]}(a_1)$ for some $a_1 \in D$:

1. send a_1 to the oracle
2. receive a vector $\mathbf{v}$ that is linearly dependent with $F^{[2]}(a_1, *)$
3. if $\mathbf{v} = 0$, then $F^{[1]}(a_1) = 0$
4. otherwise, let v_{a_2} be the first nonzero entry (so $v_{a_2} = 1$)

$$F^{[1]}(a_1) = \sum_{b \in D} F^{[2]}(a_1, b) = F^{[2]}(a_1, a_2) \cdot \sum_{b \in D} v_b$$

A Framework for Solving #CSP($\mathcal{L}$)

To compute $F^{[2]}(a_1, a_2)$:

1. send (a_1, a_2) to the oracle
2. receive $\mathbf{w}$ that is linearly dependent with $F^{[3]}((a_1, a_2), *)$
3. if $\mathbf{w} = 0$, then $F^{[2]}(a_1, a_2) = 0$
4. otherwise, let w_{a_3} be the first nonzero entry (so $w_{a_3} = 1$)

$$F^{[2]}(a_1, a_2) = \sum_{b \in D} F^{[3]}\big((a_1, a_2), b\big) = F^{[3]}(a_1, a_2, a_3) \cdot \sum_{b \in D} w_b$$

A Framework for Solving #CSP($\mathcal{L}$)

- After $n-1$ steps, we reduce

$$F^{[1]}(a_1) \longrightarrow F^{[n]}(a_1, a_2, \ldots, a_n)$$

  for some appropriate $a_2, \ldots, a_n$, with the help of the oracle. Note that $F = F^{[n]}$ can be evaluated efficiently

- Almost the whole proof of the theorem is trying to understand how and when we can implement this oracle efficiently?

What does the Oracle Need

- Fix $t \in [n]$. Compute a set of d-dimensional vectors

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_h$$

s.t. every row $F^{[t]}(\mathbf{x}, *)$ is linearly dependent with one of them

- Also need to "know":

$$S_1, S_2, \ldots, S_h \subseteq D^{t-1}$$

s.t. $\mathbf{x} \in S_i$ iff $F^{[t]}(\mathbf{x}, *)$ is linearly dependent with $\mathbf{v}_i$

Two Difficulties

1. In general, an $m \times d$ matrix may have m pairwise linearly independent rows. For $F^{[t]}$, a $d^{t-1} \times d$ matrix, we cannot afford to keep track of d^{t-1} many such vectors $\mathbf{v}_i$.
2. In general, the sets S_i's may be exponentially large in t.

Counting Graph Homomorphisms (or Partition Function)

With real weights [Goldberg, Grohe, Jerrum and Thurley] and with complex weights [Cai, C and Lu]

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \zeta & \zeta^{-1} & \zeta^{2} & \zeta^{-2} \\ 1 & \zeta^{2} & \zeta^{-2} & \zeta^{-1} & \zeta \\ 1 & \zeta^{-1} & \zeta & \zeta^{-2} & \zeta^{2} \\ 1 & \zeta^{-2} & \zeta^{2} & \zeta & \zeta^{-1} \end{pmatrix}$$

If any two rows of $F^{[t]}$ are either linearly dependent or orthogonal then it can have no more than d pairwise independent rows.

The Block Orthogonality condition

Let $F : D^n \to \mathbb{C}$ be a function defined by an input instance of $\#\text{CSP}(\mathcal{L})$, and $t \in [n]$. Every two rows of $F^{[t]}$ are either linearly dependent or orthogonal.

Lemma

If $\mathcal{L}$ does not satisfy the Block Orthogonality condition, then the problem $\#CSP(\mathcal{L})$ is #P-hard.

[Bulatov] and [Dyer and Richerby]: Mal'tsev polymorphism

Witness Function (or Frame) [Dyer and Richerby]

Let $R \subseteq D^n$. If R has a Mal'tsev polymorphism φ, then it has a succinct representation, called a witness function. A witness function ω of R is of linear size in n. Given ω and $\mathbf{x} \in D^n$, one can decide whether $\mathbf{x} \in R$ efficiently.

The Mal'tsev condition

Let $F : D^n \to \mathbb{C}$ be a function defined by an input instance of $\#\mathrm{CSP}(\mathcal{L})$, and $t \in [n]$. Then every $S_i \subseteq D^{t-1}$ has a Mal'tsev polymorphism. Indeed the condition requires all such sets to share a common Mal'tsev polymorphism.

Lemma

If $\mathcal{L}$ does not satisfy the Mal'tsev condition, then the problem $\#CSP(\mathcal{L})$ is $\#$P-hard.

The New Plan

Let $F : D^n \rightarrow \mathbb{C}$ denote the function defined by the input

For each $t \in [n]$:

1. compute $\mathbf{v}_1, \ldots, \mathbf{v}_h$, for some $h \leq d$, such that every $F^{[t]}(\mathbf{x}, *)$ is linearly dependent with one of the $\mathbf{v}_i$'s
2. compute a witness function ω_i for each S_i

How to compute these objects efficiently?

New Difficulty

Consider $t = n$ and $F^{[n]} = F$: need $\mathbf{v}_1, \ldots, \mathbf{v}_h$ and ω_i for S_i

1. By [Dyer and Richerby] and the Mal'tsev condition, one can construct a witness function ω for $R \subseteq D^{n-1}$:

$$\mathbf{x} \in R \iff \exists\, b \in D,\ F(\mathbf{x}, b) \neq 0.$$

2. By definition, $R = S_1 \cup S_2 \cup \cdots \cup S_h$

Can we use ω to compute a witness function ω_i for each S_i

Wanted: The Splitting Operation

The setting:

- Let $R \subset D^n$ and $S_1, \ldots, S_h$ be an h-way partition of R. It is known that all these sets share a Mal'tsev polymorphism φ.
- We DO NOT know h, though it is guaranteed that $h \leq d$.
- We have a witness function ω of R.
- There is a black box we can query: Upon receiving an $\mathbf{x} \in R$, it returns the unique index $j \in [h]$ such that $\mathbf{x} \in S_j$.

Can we compute h and a witness function ω_i for S_i efficiently?

Algorithm for the Splitting Operation

If R and the $S_1, \ldots, S_h$ satisfy the following condition:

- For any $\mathbf{y} \in D^\ell$, $\ell \in [n]$, let

$$\text{type}(\mathbf{y}) = \Big\{ j \in [h] : \exists\, \mathbf{z} \in D^{n-\ell} \text{ such that } \mathbf{y} \circ \mathbf{z} \in S_j \Big\} \subseteq [h]$$

- The partition condition: For any $\mathbf{y}, \mathbf{y}' \in D^\ell$, type($\mathbf{y}$) and type($\mathbf{y}'$) are either disjoint or the same.

we have an efficient algorithm for splitting.

Algorithm for the Splitting Operation

1. A recursive algorithm that, given $\mathbf{x} \in D^{\ell}$, computes $\text{type}(\mathbf{x})$. Here the partition condition is crucial!
2. A recursive algorithm that, given $\mathbf{x} \in D^{\ell}$ and $j \in \text{type}(\mathbf{x})$, finds a $\mathbf{y} \in D^{n-\ell}$ such that $\mathbf{x} \circ \mathbf{y} \in S_j$.
3. Finally, construct a witness function ω_i for each S_i

The Type Partition Condition

The Type Partition condition

Essentially it requires that, every time we need to apply the splitting operation when implementing the oracle, the sets R and $S_1, \ldots, S_h$ satisfy the partition condition.

Lemma

If $\mathcal{L}$ does not satisfy the Type Partition condition, then the problem $\#CSP(\mathcal{L})$ is $\#P$-hard.

Putting the Pieces Together

Let $F : D^n \to \mathbb{C}$ denote the function defined by the input

Inductively, for t from n to 2:

Use the oracles for $F^{[t+1]}, F^{[t+2]} \ldots, F^{[n]}$ to

1. compute $\mathbf{v}_1, \ldots, \mathbf{v}_h$, where $h \leq d$, such that every $F^{[t]}(\mathbf{x}, *)$ is linearly dependent with one of the $\mathbf{v}_i$'s
2. compute a witness function ω_i for each S_i,
3. both done by using the algorithm for splitting

Finally, compute $\sum_{\mathbf{x}} F(\mathbf{x})$ using these oracles

Open Problem

Determine the decidability of these tractability conditions:

- Given a finite set of complex-valued functions $\mathcal{L}$, can we decide whether $\mathcal{L}$ satisfies these conditions in finite time?

Thank you!