Max-2-CSP in expected polynomial time

Ravelomanana Vlady

LIPN – UMR CNRS 7030, University of Paris Nord
1 Introduction

2 MAX-2-CSP

3 Reductions

4 Expected running time (the analysis)

5 Conclusion
By CSP, we mean Constraint Satisfaction Problems (cf. A. Montanari AofA’08).

- **n variables** $x_1, x_2, \cdots x_n$ belonging to finite domains, i.e., $x_i \in \xi_i$.

- A set of **constraints** (clauses) over these variables: where by constraint we mean relations between the variables defining *authorized* combinations.

- **Decision problem**: does it exist a solution (affectation of the variables) satisfying all the constraints?

- **Optimization problem**: maximize the number of satisfiable clauses by some affectation(s).
Example

Domains: \((x, y) \in \{0, 1\}^2, (z, t) \in \{0, 1, 2, 3\}^2\)

Constraints:

\[
\begin{cases}
(x, y) \in \{(1, 0), (0, 1)\} \\
x \neq z \\
y + z = 0 \mod 2 \\
t \geq y
\end{cases}
\]

Our 2-CSP settings

- All the domains are of **size 2** : the variables can take 2 values.
- Each clause (constraint) concerns **exactly 2 variables**.
Summary

1. Introduction

2. MAX-2-CSP

3. Reductions

4. Expected running time (the analysis)

5. Conclusion
Formal description

- An instance is given by \((G, S)\) where \(G = (V, E)\) is the **underlying graph** and \(S\) is a **score** function.
- W. l. o. g. the vertices can take two **colors** Red or Blue.
- For each edge \(e = (x, y)\) and for each vertex \(v\) of the graph, we have their resp. associated scores:
  \[ s_e = s_{xy} : \{R, B\}^2 \rightarrow \mathbb{R} \quad \text{AND} \quad s_v : \{R, B\} \rightarrow \mathbb{R}. \]

Goals:

**Decision problem**: Find a solution (or a coloring) of the vertices which is a function \(\Phi\) satisfying all the constraints.

**Optimization problem**: Find \(\max_{\Phi \in \{\text{all colorings}\}} s(\Phi) : \)

\[ \Phi : V \rightarrow \{R, B\}, \ s(\Phi) = \sum_{v \in V} s_v(\Phi(v)) + \sum_{xy \in E} s_{xy}(\Phi(x), \Phi(y)) \in \mathbb{R}. \]
Under these assumptions and settings

1. The problem is sufficiently general: The settings encompass \textsc{MaxCut}, \textsc{MaxDicut}, \textsc{MaxIs}, \textsc{Max2Sat}, ...

2. Main facts: best known algorithms need $c \#edges - \#vertices$ global iterations (with $c > 1$) with the worst cases, for instance \textsc{MaxCut} in $O(2^{19/100\#edges})$ \textsc{Scott – Sorkin 2007}

3. What about average-case analysis?
The instances are randomly generated using a graph $G(n, M)$ as support.

Main steps:

- Use some reductions (same as in Scott – Sorkin).

Running time of the algorithm:

Check under what conditions this problem has **expected polynomial running time**.
Summary

1. Introduction
2. MAX-2-CSP
3. Reductions
4. Expected running time (the analysis)
5. Conclusion
MAX-2-CSP: the reduction of vertices of degree 1 (Type I)

Let \( y \) be a vertex of degree 1 (with \( x \) as unique neighbor). The initial problem is reduced from \( G = (V, E) \) to \( V' = V \setminus \{y\} \) and \( E' = E \setminus \{(x, y)\} \). The new score function \( S' \) is given by the restriction of \( s \) to \( V' \) and \( E' \) except that for \((c_1, c_2) \in \{R, B\}^2\) we have

\[
s'_x(c_1) = s_x(c_1) + \max_{c_2} \{s_{xy}(c_1, c_2) + s_y(c_2)\}.\]

Termed in other words,

\[
s'_x(R) = s_x(R) + \max(s_{xy}(R, R) + s_y(R) + s_{xy}(R, B) + s_y(B))
\]
\[
s'_x(B) = s_x(B) + \max(s_{xy}(B, B) + s_y(B) + s_{xy}(B, R) + s_y(R)).\]

**Optimal Coloration over \( S' \) \( \rightarrow \) Optimal Coloration over \( S \)**

in time : \( T_{S'} = T_S + O(1) \)
Let $y$ be a vertex with neighbors $x$ and $z$. We reduce the graph by deleting $y$ and replacing it by an edge $xz$. The new problem is then over $V' = V \setminus \{y\}$ and $E' = (E \setminus \{(x, y), (y, z)\}) \cup \{(x, z)\}$, and the new score function $S'$ is the restriction of $S$ over $V'$ and $E'$ except that for $(c_1, c_2, c_3) \in \{R, B\}^3$ we have

$$s'_{xz}(c_1, c_2) = \max_{c_3} \{s_{xy}(c_1, c_3) + s_{yz}(c_3, c_2) + s_y(c_3)\}$$

Idem ... \textbf{Optimal Coloration over $S'$ $\rightarrow$ Optimal Coloration over $S$}

in time : $T_{S'} = T_S + O(1)$
Let $y$ be a vertex of degree $\deg(y) > 2$. We define two reductions corresponding to the two assignments of the color of $y$: color $y = \text{Red}$ or color $y = \text{Blue}$.

Then we define TWO new problems accordingly. Suppose that $y$ is colored Red. For every neighbor $x$ of $y$, a new score function is defined:

$$s^R_x(R) = s_x(R) + s_{xy}(R,R) + s_y(R),$$
$$s^R_x(B) = s_x(B) + s_{xy}(B,R) + s_y(R).$$

**The Algorithm:**
Do all the reductions of the vertices of degree 1 and 2; Then for each vertex $v$ of degree $\geq 3$, solve recursively the TWO instances: ’$v$ in Blue’ and ’$v$ in Red’.
Summary

1. Introduction
2. MAX-2-CSP
3. Reductions
4. Expected running time (the analysis)
5. Conclusion
The expected running time of the algorithm is related to

$$\sum_{R=0}^{M} 2^R p_R(n, M) = \text{function of } n,$$

where

$$p_R(n, M) \overset{\text{def}}{=} \text{proba that } G(n, M) \text{ produces a graph with EXACTLY } R \text{ reductions of type III}.$$

$$p_R(n, M) \text{ is } \textbf{close to} (T \equiv \text{CAYLEY, cf. B. SALVY AofA’08})$$

$$\frac{n!}{2\pi i \binom{n}{M}} \int \frac{c_R}{(1 - T)^{3R}} \frac{(T - T^2/2)^{n-M+R}}{(n-M+R)!} \frac{e^{-T-T^2/2}}{(1 - T)^{1/2}} \frac{dz}{z^{n+1}} \sim \text{giant component unrooted trees unicycles}.$$
After calculus implying Enumeration (cf. J. Gao AofA’08) Analytic Combinatorics (cf. M. Drmota and B. Salvy AofA’08), we get

**Th.**

- If \( M = n/2 + O(1) \log n^{1/3} \ n^{2/3} \) \( \) MAX-2-CSP generated with \( G(n, M) \) can be solved in **EXPECTED POLYNOMIAL TIME**.

- If \( M = n/2 + \omega(n) \log n^{1/3} \ n^{2/3} \), there are \( \exp (\Omega (\omega(n)^3 \log n)) \) global iterations! **AVERAGE EXPONENTIAL TIME**.

- The order \( O(\log n^{1/3}) \) is **optimal** for all algorithms requiring (on worst cases) \( c \#\text{edges} - \#\text{vertices} \) with \( c > 1 \).
Summary

1 Introduction

2 MAX-2-CSP

3 Reductions

4 Expected running time (the analysis)

5 Conclusion
This work answers a problem left open by Scott and Sorkin (2004).

Similar results hold if the variables (vertices) can take any finite number of values (Red, Blue, Green, Yellow, etc ...).

The algorithm works fine (polynomial time) until around the famous "critical $n/2$ edges" for any MAX-2-CSP and for MAX-2-SAT but is **weak** for this latter (recall that the threshold for 2-SAT is $n$).

What about these issues (algorithm + analysis)?

Are dense instances of these MAX-2-CSP-like problems **really hard** even on **average**?
THANK YOU