Random Logic Programs

Kewen Wang (Griffith University)
Joint work with Lian Wen (Griffith University)
Yi-Dong Shen (Institute of Software, CAS)

2nd KRW, Guizhou, June 17, 2012
Phase Transition (SAT and CSP)
- Answer Set Programs
- Random ASP
- Quadratic Model
- Linear Model
- Summary

What is the probability that $P$ has an answer set?

$a_1, a_2, \ldots, a_n$

$a \leftarrow \text{not } b$

$a \leftarrow \text{not } a$

n(n-1) rules with probability $p$

n rules with probability $d$

random program

What is the probability that $P$ has an answer set?
Random NP-Hard Problems

- Average-case complexity vs. Worst-case complexity
- Goldberg 1979: SAT is “easy on average”
- Franco and Paull 1983: Analysis of random instances of k-SAT
- Random $k$-CNF formulas: $F_k(n, l)$ is in CNF with $l$ clauses over $n$ atoms, clauses are chosen uniformly, independently and without replacement among all $2^k \times \binom{n}{k}$ non-trivial clauses of length $k$. 
Phase Transition

- Thermal Physics: the transformation of a thermodynamic system from one phase or state of matter to another.
- Unlike standard phase transition systems, phase transition here means radical changes of AI systems:
  - Satisfiable/Consistency $\iff$ Unsatisfiable/Inconsistency
  - Hard/Long Search $\iff$ Easy/Short Search
  - $\ldots$ $\iff$ $\ldots$
Phase Transition for 3-SAT

- Satisfiability Threshold Conjecture: Let $k \geq 3$ and the number $l$ of clauses be a function $s(r, n)$ with $\lim_{n \to \infty} s(r, n) = \infty$. Then there exists a constant $r_k > 0$ such that, if $r > r_k$,

$$\lim_{n \to \infty} Pr(F_k(n, l) \text{ is satisfiable}) = 1.$$  

- The ratio $r = l/n$ can determine the satisfiability and the average solving time of algorithms for 3-SAT.
Extant Work for Random ASP

- Zhao and Lin, 2003 (first attempt?)
- Schlipf, Truszczynski and Wong, 2005
- Namasivayam and Truszczynski, 2009
  two-literal program $\Leftrightarrow$ random graph
  answer set $\Leftrightarrow$ kernel

Why only few attempts are made?
- Mathematical models for SAT and CSP heavily rely on the monotonicity.
- Our experiments show that random ASP does not demonstrate similar phase transition as SAT.
- Experimental studies are limited by the performance of current ASP solvers.
Consistency Threshold Conjecture for ASP

- $P_k(n, l)$: random logic programs, $k \geq 2$, $l = s(r, n)$: the number of rules, $\lim_{n \to \infty} s(r, n) = \infty$. Then $\exists r_k > 0$ s.t., if $r > r_k$,
  \[ \lim_{n \to \infty} Pr(P_k(n, l) \text{ is consistent}) = 1. \]

- $k = 2$ (negative two-literal ASP programs) has a similar status in ASP as random 3-SAT in SAT:
  - The problem of deciding a two-literal program has an answer set is NP-complete.
  - Every normal program is equivalent to a negative two-literal program in ASP
  - Many important NP-complete problems can be easily encoded as two-literal programs.
  - Such simple programs allow us to conduct experiments with relatively large sizes of programs.
A logic program is a finite set of rules of the form

\[ a \leftarrow b_1, \ldots, b_s, \text{not } c_1, \ldots, \text{not } c_t, \]

A rule is positive, if \( t = 0 \); negative, if \( s = 0 \).

Program \( P \) is positive (resp. negative), if every rule in \( P \) is positive (resp. negative).

The reduct \( P^X \) of \( P \) on \( X \) of literals:

\[ P^X = \{ \text{head}(r) \leftarrow \text{body}^+(r) \mid r \in P, \text{body}^-(r) \cap X = \emptyset \}. \]

\( X \) is an answer set of \( P \) if \( X \) is the minimal model of \( P^X \).

\( \text{AS}(P) \): the collection of all answer sets of \( P \).
A two-literal program is a logic program s. t. each rule has exactly two literals.

A rule \( a \leftarrow \text{not } b \) is said to be a pure rule if \( a \neq b \) while \( a \leftarrow \text{not } a \) is a constraint rule.

Fact rules and constraints can be expressed in two-literal programs.

Each logic program \( P \) is equivalent to a negative two-literal program under answer sets.
A random program \( \hat{P} \) is a negative two-literal program:

1. The atoms in the random program are from a set \( A_n \) of \( n \) atoms \( (n \geq 1) \).
2. Each pure rule is randomly selected with a probability \( p \), \( 0 < p < 1 \).
3. Each constraint rule is randomly selected with a probability \( d \), \( 0 \leq d \leq 1 \).
Theorem 1. \( \hat{P} \): generated under quadratic model and \( r = \frac{1-d}{1-p} \).

Then

\[
\lim_{n \to \infty} \Pr(|\mathcal{AS}(\hat{P})| > 0) = \begin{cases} 
1 & \text{if } r > 1 \\
0 & \text{if } r \leq 1
\end{cases}
\] (1)

Corollary \( P \): generated under quadratic model with \( d = 0 \). Then

\[
\lim_{n \to \infty} \Pr(|\mathcal{AS}(P)| > 0) = 1.
\]
Relation to de la Vega’s Theorem

- **Kernel**: $U$ is a *kernel* of a digraph $G = (V, E)$ iff $U \subseteq V$, no edges between vertices in $U$, and every vertex in $V \setminus U$ is connected to one vertex in $U$.

- $S$ is an answer set of negative two-literal program $P$ iff $S$ is a kernel of the dependency graph of $P$.

- **de la Vega’s Theorem**: $D(n, p)$: a random digraph with $n$ vertices and edge probability $p$ ($n > 0$, $0 \leq p \leq 1$). Then

$$\lim_{n \to \infty} Pr(D(n, p) \text{ has a kernel }) = 1.$$
Distribution of Answer Sets for Random Programs

\[ n = 200, \ p = 0.02, \ d = 0, \text{ and } 10,000 \text{ random programs:} \]

**Figure:** Answer set distribution, \( n = 200, \ p = 0.02, \text{ and samples 10000.} \)
$p = 0.25$, $d = 0$ and $n$ varies from 20 to 200.

18 testing points: $n = 20, 30, 40, ..., 200$ and for each testing point, 1,000 random programs are generated

(a) Experimental and theoretical values for $m_0$, with $p = 0.25$, $d = 0$

(b) Experimental and theoretical values for $Pr_{AS}$, with $p = 0.25$, $d = 0$
Average Number of Answer Sets for Random Programs (2)

$p = d = 0.1$ and $n$ varies from 20 to 200.
18 testing points: $n = 20, 30, 40, ..., 200$ and for each testing point, 1,000 random programs are generated.

(c) Average size $m_0$ of answer sets, $p = d = 0.1$, $n = 20 \ldots 200$

(d) $Pr_{AS}$, $p = d = 0.1$, $n = 20 \ldots 200$

Figure: Result of Experiment 3.
Phase Transition for Random Programs

\( p = 0.1, \ n = 50, 150, 300. \)

For each value of \( n \), 21 representative values of \( d \) are tested and thus correspondingly, 21 values for the ratio \( r = \frac{1-d}{1-p}. \)
A random program generated by quadratic model has a high density (it has $p \times n \times (n - 1)$ rules).

Rule-based systems in the semantic web and ontologies usually have a low density of rules.

**Linear Model:** A random program $P$ on $A_n$ is a two-literal program generated as follows:

- $P$ consists of only rules of the form $a_1 \leftarrow \text{not } a_2$ where $a_1, a_2 \in A_n$ and $a_1 \neq a_2$.
- $P$ has $l = c \times n$ rules where $c$ is a constant number.
- Each rule is selected to $P$ with the probability $p$ where $p = (c \times n)/(n \times (n - 1)) = c/(n - 1)$. 
Theorem 2. \( P \): random program under linear model. Then

\[
Pr\left( E(|AS(P)| > 0) \right) \approx 1 - e^{-\frac{\alpha^2}{\sqrt{\alpha^2 + (c^2 + 2c)\alpha - c^2}}}
\]

Here \( \alpha \) is also determined by constant \( c \): \( \ln \alpha = \frac{c}{\alpha} \).

If \( c = c(n) \) and \( c(n) \to \infty \) when \( n \to \infty \),

\[
\lim_{n \to \infty} Pr\left( E(|AS(P)| > 0) \right) = 1.
\]

**de la Vega’s Conjecture:** \( D(n, p) \): a random digraph with \( n \) vertices and edge probability \( p = cn^{-1} \) where \( c \) is a constant s. t. \( c \to \infty \). Then

\[
\lim_{n \to \infty} Pr\left( D(n, p) \text{ has a kernel} \right) = 1.
\]

**de la Vega’s Conjecture** is a corollary of Theorem 2.
$c = 2.0$ and $n = 2000$

$p \approx 0.001001$ and $q \approx 0.998999$.

$\alpha \approx 2.3454$. $m_0 \approx 1147.3$, $E(|\mathcal{AS}(P,m_0)|) \approx 0.04231$, and $\sigma^2 \approx 247.259$.
Each normal program is equivalent to a negative two-literal programs under answer set semantics.

Quadratic model for random programs and a phase transition for consistency.

Linear model for random programs and a phase transition for consistency.

Future work:
- To study the phase transition for hardness.
- To extend these two models to more general classes of random programs.
THANK YOU