What is Answer Set Programming?

Joohyung Lee (joolee@asu.edu)

Computer Science and Engineering
Arizona State University

What is Answer Set Programming (ASP)?

A new form of declarative programming oriented towards combinatorial search problems and knowledge-intensive applications.

In answer set programming we obtain the answers by declaring the properties of the answers, using logic rules. It differs from procedural languages like C/C++ and Java where we specify the sequence of operations to be executed to generate an answer.

The idea of ASP is to represent a given search problem as the problem of finding an answer set for some logic program, and then find a solution using an answer set solver.

Brief History of Answer Set Programming

1988: Definition of answer sets for Prolog-like programs.

1992: Extending the definition to more general programs.

1996: SMODELS: first answer set solver.

1999: ASP identified as a new programming paradigm.

Efficient solvers are available: SMODELS, DLV, ASSAT, CMODELS, ASPPS, ...

WASP (Working Group on Answer Set Programming) : 17 European universities in 8 countries. Funded by EU.

ASP has already found applications in many different areas, including knowledge representation, planning, diagnosis, decision support systems, model checking, production configuration, VLSI routing, the Semantic Web, computational linguistics, bioinformatics, . . .

Paradigm	SAT	ASP
Input	set of clauses	set of rules
Solutions	models	stable models (answer sets)
	monotonic	nonmonotonic
Solvers	MiniSAT, zChaff, Jerusat, Walksat, Relsat. . .	Smodels, NoMore, DLV, ASSAT, Cmodels, SAG. . .

Answer Set Semantics

$$A_0 \quad \leftarrow \quad A_1, \ldots, A_m, \textit{not}\ A_{m+1}, \ldots, \textit{not}\ A_n$$

means intuitively that

If you have generated $A_1, \ldots, A_m$, and

it is impossible to generate any of $A_{m+1}, \ldots, A_n$,

then you may derive A_0.

Answer Sets (Stable Models) vs. Models

Answer sets	Models
$p \leftarrow \textit{not}\ s$	$\neg s \rightarrow p$
$p \leftarrow r$	$r \rightarrow p$
$q \leftarrow r$	$r \rightarrow q$
$r \leftarrow p, q$	$p \wedge q \rightarrow r$
$\{p\}$	$\{p\}$, $\{s\}$, $\{p, s\}$, $\{q, s\}$, $\{p, q, r\}$, $\{p, q, r, s\}$

Answer Sets and Prolog

$$p \leftarrow \textit{not } q$$
$$q \leftarrow \textit{not } p$$

Prolog does not terminate on query `p` or `q`.

```
?- p.
ERROR: Out of local stack
   Exception: (729,178)
```

SMODELS returns

```
Answer: 1
Stable Model: p
Answer: 2
Stable Model: q
```

Finite ASP programs are guaranteed to terminate.

More General Programs

```
1 {p,q,r} 2.
:- p, q.
```

```
> lparse ex1 | smodels 0
```

```
Answer: 1
Stable Model: p
Answer: 2
Stable Model: p r
Answer: 3
Stable Model: q
Answer: 4
Stable Model: r
Answer: 5
Stable Model: q r
```

Example 1: 8-queens problem

How to place 8 queens that don't attack each other on a 8-by-8 board?

A program in answer set programming that solves this problem consists of the following rules:

"Each row has exactly one queen"

$$1 \leq \{q_{i,1}, \ldots, q_{i,8}\} \leq 1 \quad (1 \leq i \leq 8). \tag{1}$$

"Two queens cannot stay on the same column"

$$\bot \leftarrow q_{i,j}, q_{i',j} \quad (1 \leq i < i' \leq 8; 1 \leq j \leq 8).$$

"Two queens cannot stay on the same diagonal"

$$\bot \leftarrow q_{i,j}, q_{i',j'} \quad (1 \leq i < i' \leq 8; 1 \leq j, j' \leq 8; i' - i = |j' - j|).$$

What is an Answer Set?

An answer set for a program is a set of predicates that satisfy the conditions imposed by the rules of the program. For instance, in the previous program for the 8-queens problem, each answer set is a set of predicates of the form $q_{i,j}$ that satisfy the above rules and hence represent a valid arrangement of the queens.

An answer set solver finds the answer sets of a logic program. The two most-known ones are SMODELS (http://www.tcs.hut.fi/Software/smodels) and DLV (http://www.dbai.tuwien.ac.at/proj/dlv).

Finding a solution for the 8-queens problem

The previous program that solves the 8-queens problem can be written in the syntax of LPARSE, a front end for SMODELS, in the following way:

```
num(1..8).
hide num(_).

1 {q(I,J): num(J)} 1 :- num(I).
:- q(I,J), q(I1,J), num(I;I1;J), I<I1.
:- q(I,J), q(I1,J1), num(I;I1;J;J1), I<I1,
                     I1-I==abs(J1-J).
```

Finding one solution for the 8-queens problem

With command line

```
% lparse 8queen |smodels
```

```
smodels version 2.26. Reading...done
Answer: 1
Stable Model: q(1,4) q(2,6) q(3,1) q(4,5) q(5,2)
q(6,8) q(7,3) q(8,7)
True
Duration 0.020
Number of choice points: 3
Number of wrong choices: 0
Number of atoms: 89
Number of rules: 552
Number of picked atoms: 240
Number of forced atoms: 5
Number of truth assignments: 2535
Size of searchspace (removed): 64 (0)
```

Finding all solutions for the 8-queens problem

With the same program, but with the following command line

```
% lparse 8queen |smodels 0
```

SMODELS computes and shows all 92 valid queen arrangements. For instance, the last one is

```
Answer: 92
Stable Model: q(1,7) q(2,2) q(3,4) q(4,1) q(5,8)
q(6,5) q(7,3) q(8,6)
```

Example 2: Estimating Schur numbers

A set S of integers is called *sum-free* if there are no numbers x, y in S such that $x + y$ is in S. For instance, $\{1, 3, 5\}$ is sum free; $\{2, 3, 5\}$ and $\{2, 4\}$ are not.

What is the largest value of n such that $\{1, \ldots, n\}$ can be partitioned into 3 sum-free subsets?

Example 2: Estimating Schur numbers

A set S of integers is called *sum-free* if there are no numbers x, y in S such that $x + y$ is in S. For instance, $\{1, 3, 5\}$ is sum free; $\{2, 3, 5\}$ and $\{2, 4\}$ are not.

What is the largest value of n such that $\{1, \ldots, n\}$ can be partitioned into 3 sum-free subsets?

The Schur number $S(k)$ is the largest integer n for which the interval $\{1, \ldots, n\}$ can be partitioned into k sum-free sets.

Solution for $n = 13$:

$$\{2, 3, 11, 12\}, \{5, 6, 7, 8, 9\}, \{1, 4, 10, 13\}.$$

The Schur number $S(3)$ is 13.

Input:

```
    subset(1..k).
    number(1..n).
    #domain number(X;Y).

    1{in(X,I) : subset(I)}1.
    :- in(X,I), in(Y,I), in(X+Y,I), subset(I), X+Y<=n.
```

```
 > lparse -c k=3 -c n=13 -d none schur | smodels
```

Output:

```
   Stable Model: in(1,3) in(2,1) in(3,1) in(4,3) in(5,2) in(6,2)
   in(7,2) in(8,2) in(9,2) in(10,3) in(11,1) in(12,1) in(13,3)
```

which represents a partition of $\{1, \ldots, 13\}$ into 3 sum-free sets:

$$\{2, 3, 11, 12\} \cup \{5, 6, 7, 8, 9\} \cup \{1, 4, 10, 13\}.$$

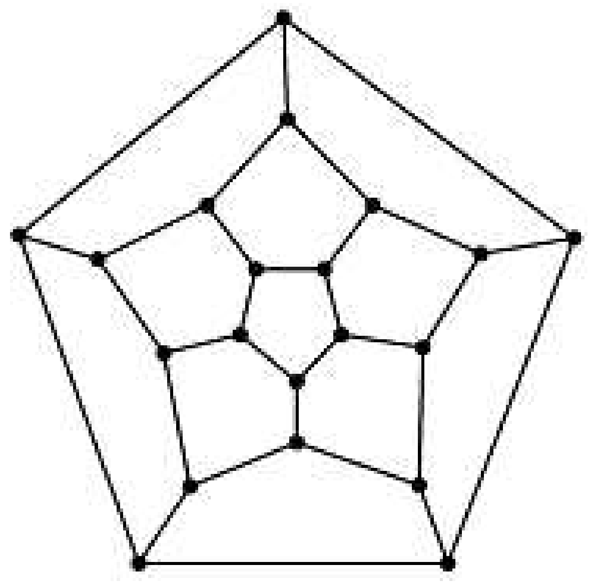
Example 3: Sudoku

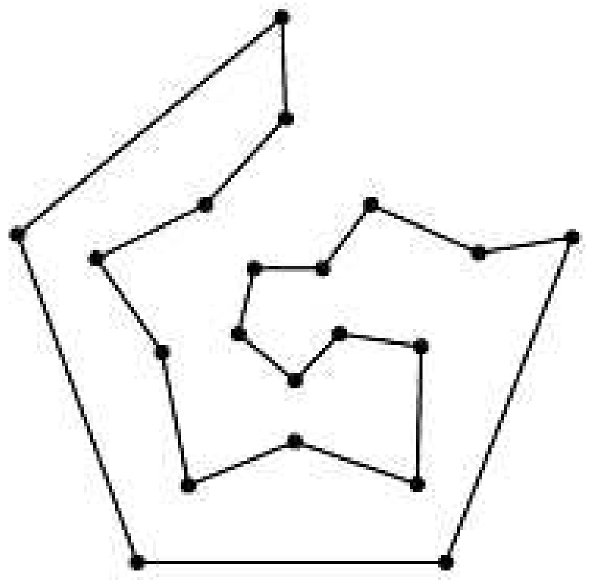
	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

Sudoku in ASP

```
row(1..9).            col(1..9).            num(1..9).
#domain row(R;RR).    #domain col(C;CC).    #domain num(N).

1 {a(R,C,N1) : num(N1)} 1.

1 {a(R1,C,N) : row(R1)} 1.

1 {a(R,C1,N) : col(C1)} 1.

:- a(R,C,N), a(RR,CC,N), R!=RR, C!=CC,
   (((R-1)/3)*3 + (C-1)/3) == (((RR-1)/3)*3 + (CC-1)/3).
```

Example 4: Hamiltonian Cycles





Hamiltonian Cycles in ASP

```
{in(U,V)} :- edge(U,V).

:- in(U,V), in(U,W), V!=W.
:- in(U,W), in(V,W), U!=V.

reachable(U) :- in(v0,U).
reachable(V) :- reachable(U), in(U,V).

:- not reachable(U), vertex(U).
```

Applications

These are some fields where answer set programming has been used:

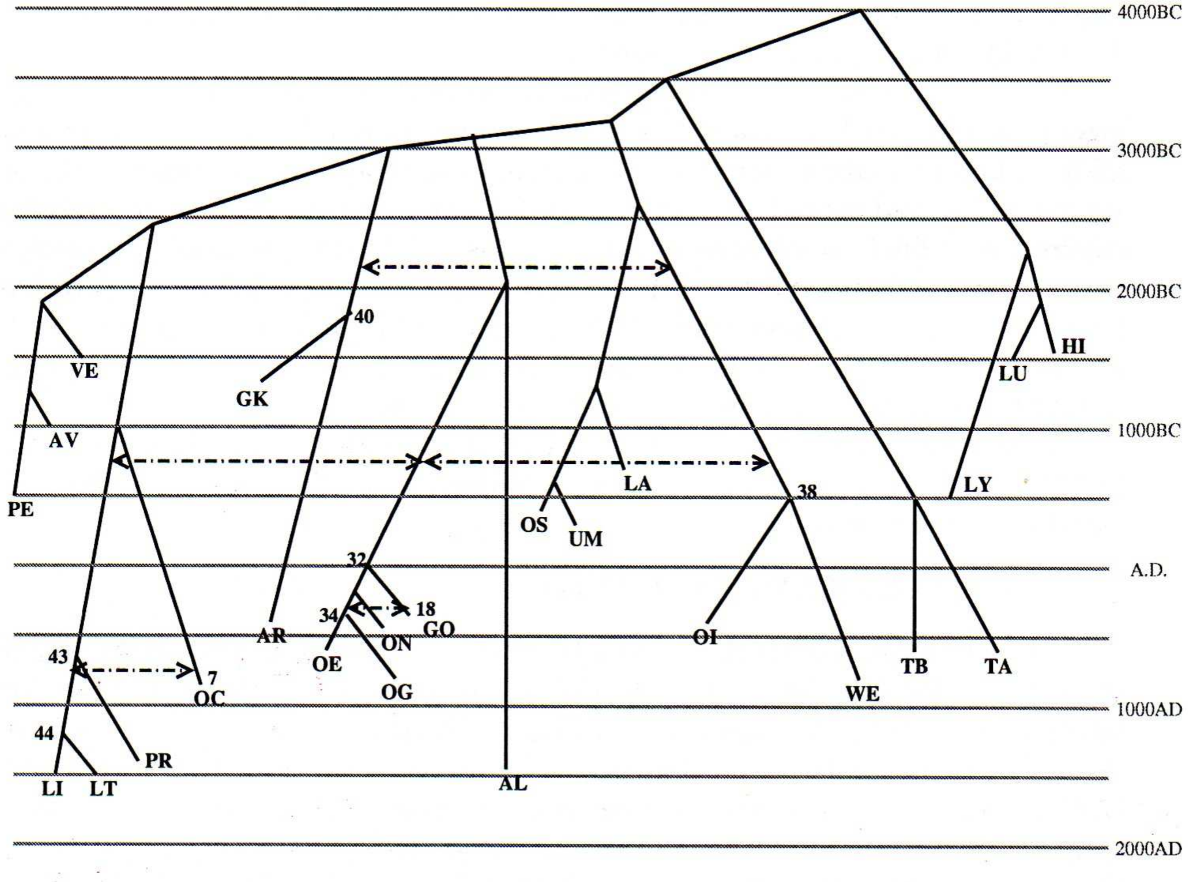
- planning
- commonsense reasoning
- model checking
- VLSI (wire routing)
- historical linguistic
- phylogeny reconstruction
- verifying security protocols

Some Applications

- *Answer Set Programming and Plan Generation.*
- *A Logic Programming Approach to Knowledge-state Planning.*
- *Reconstructing the Evolutionary History of Indo-European Languages Using Answer Set Programming.*
- *Character-based Cladistics and Answer Set Programming.*
- *Rectilinear Steiner Tree Construction Using Answer Set Programming.*

- *An A-Prolog Decision Support System for the Space Shuttle* (RCS/USA-Advisor : decision support system for shuttle controllers).
- *Developing a Declarative Rule Language for Applications in Product Configuration* (variantum : spin-off).
- *Bounded LTL Model Checking with Stable Models.*
- *Using Logic Programs with Stable Model Semantics to Solve Deadlock and Reachability Problems for 1-Safe Petri Nets.*
- *Data Integration: A Challenging ASP Application.*
- *Efficient Haplotype Inference with Answer Set Programming*

Reconstructing the Evolutionary History of Indo-European Languages Using Answer Set Programming, (Erdem, Lifschitz, Nakhleh, Ringe, PADL-03)



How to Turn ASP into classical logic?

Classical logic is monotonic: If $\Gamma \vdash A$, then $\Gamma \cup \Delta \vdash A$

ASP language is nonmonotonic: $\Gamma \vdash A$, but $\Gamma \cup \Delta \not\vdash A$ for some Δ.

Theorem on loop formulas The answer sets of a program are exactly the models of the program and loop formulas.

Π_1	Π_1	$\cup\ LF(\Pi_1)$
$p \leftarrow q$	$q \supset p$	$p \supset q$
$q \leftarrow p$	$p \supset q$	$q \supset p$
$r \leftarrow \textit{not}\ p$	$\neg p \supset r$	$r \supset \neg p$
		$p \wedge q \supset \bot$

Why ASP is a Good Formalism for Knowledge Representation?

Any equivalent translation from logic programs to propositional formulas involves a significant increase in size assuming a plausible conjecture ($P \not\subseteq NC^1/poly$) [Lifschitz and Razborov, "Why are there so many loop formulas?", ACM TOCL, 2006].

> *How succinctly can the formalism express the set of models that it can? . . . [W]e consider formalism A to be stronger than formalism B if and only if any knowledge base in B has an equivalent knowledge base in A that is only polynomially longer, while there is a knowledge base in A that can be translated to B only with an exponential blowup.* [Gogic, Kautz, Papadimitriou, Selman, IJCAI-95]

Conclusion

ASP is a new form of declarative programming oriented towards combinatorial search problems and knowledge intensive applications.