From 3-valued semantics to supported model computation for logic programs in vector spaces

Feb. 22, 2020

Taisuke Sato, AIST Japan

(joint work with Chiaki Sakama and Katsumi Inoue)

Why supported models?

- Combinatorial problems:
  - SAT, TSP, planning, scheduling, bioinformatics....
  - Problem: to find a discrete finite solution satisfying constraints
- Solve by:
  - constraint programming (CP), integer linear programming (ILP), SAT, answer set programing (ASP),...
- ASP:
  - problem = logic program DB
  - solution = stable model of DB
  - BTW searching for supported models looks a good substitute

    stable models $\subseteq$ supported models

An overview

- Example program DB

  DB = { single(X)←man(X) & not(husband(X)),

  husband(X)←man(X) & not(single(X)) }

  iff(DB) = { s(X) ⇔ m(X) & not(h(X)),

  h(X) ⇔ m(X) & not(s(X)) }

  $\mathbf{M}_{DB}$ is a supported model iff $\mathbf{M}_{DB} \models$ iff(DB)

- Compute a supported model $\mathbf{M}_{DB}$ by

  Step 1: Deterministically compute three-valued model $\mathbf{M}_{DB}^3$

  Step 2: Assign {**t**, **f**} to undefined atoms in $\mathbf{M}_{DB}^3$ appropriately

  – while conducting **Step 1** and **Step 2** in a vector space by matrix operation for efficiency & scalability

AIST

Logic programming semantics in vector spaces: simple case

- DB is a Horn program:
  - taking a transitive closure r2 of r1

    r1(a,b), r1(b,c),

    r2(X,Z) ← r1(X,Z), r2(X,Z) ← r1(X,Y) & r2(Y,Z)
  - the least model $\mathbf{M}_{DB}$ = { a: ground atom | DB ⊢ a }
- We embed the whole task in a vector space
  - DB^g = grounding of DB
  - encode DB^g by binary matrix Q and threshold vector θ
  - represent $\mathbf{M}_{DB}$ as binary vector $\mathbf{u}$ (true(1),false(0))
  - compute $\mathbf{u}$ s.t. $\mathbf{u} = (Q\mathbf{u})_{\geq\theta}$ where $(x)_{\geq\theta} = (x \geq \theta\ ?\ 1 : 0)$

Computing matricized $\mathbf{M}_{\mathrm{DB}}$

$DB_1 = \{ p \leftarrow q \& r,\ q \leftarrow r \& s,\ r \leftarrow q \vee s,\ s \leftarrow \}$

$$
\begin{array}{ccccc|c|l}
 & p & q & r & s & \theta & \text{\% threshold} \\
\hline
Q = [& 0 & 1 & 1 & 0 & 2 & \text{\% p: AND goal} \\
 & 0 & 0 & 1 & 1 & 2 & \text{\% q: AND goal} \\
 & 0 & 1 & 0 & 1 & 1 & \text{\% r: OR goal} \\
 & 0 & 0 & 0 & 1\ ] & 1 & \text{\% s: fact as } s \leftarrow s
\end{array}
$$

program matrix

$\mathbf{u}_0 = [\ 0 \quad 0 \quad 0 \quad 1\ ]^T$ % $\mathbf{u}_0(\text{fact})=1$

$\mathbf{u}_{n+1} = (Q\mathbf{u}_n) \geq \theta$ % $(x) \geq \theta = 1$ if $x \geq \theta$, $=0$ o.w.

$\mathbf{u}_\infty = [\ 1 \quad 1 \quad 1 \quad 1\ ]^T$ % all atoms are true in $\mathbf{M}_{\mathrm{DB}}$

% $O(N^3)$ where $N = |HB|$

Hard case: computing supported models

- Supported models of DB with negation
  - may not exist
  - existence of supported model ➔ NP-complete
- Naïve strategy
  - non-deterministically assign {**t**,**f**} to atoms to make a supported model

AIST

Our strategy

- a ∈ **SS** iff :-a succeeds by fair SLD refutation
  - a is true in all supported models of DB
- a ∈ **FF** iff :-a finitely fails by fair SLD refutation
  - a is false in all supported models of DB
- Search for supported models by

Step 1: deterministically compute the least 3-valued model

$\mathbf{M}_{DB}^{3}$ = (**SS**,**FF**) by DB^d (*dualized DB*)

➔ remove nondeterminacy

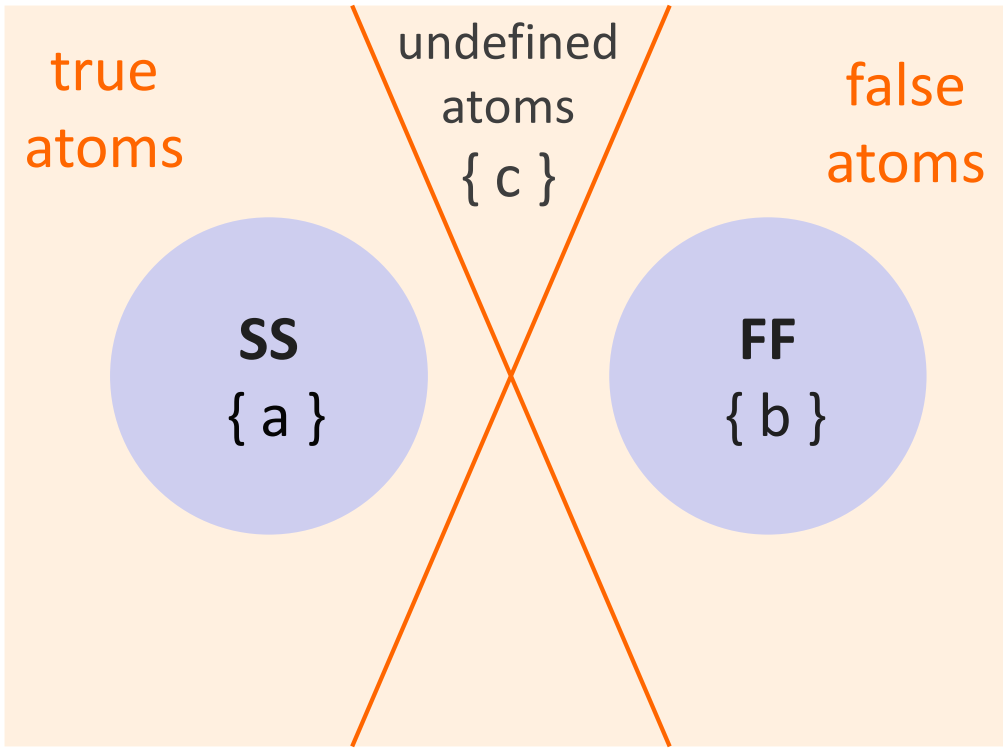
Step 2: non-deterministically assign {**t**,**f**} to undefined atoms

(∉ **SS** ∪ **FF**) to make a supported model

Two supported models

DB:
$a \leftarrow \mathrm{not}(b)$
$c \leftarrow a \,\&\, c$

iff(DB):
$a \Leftrightarrow \mathrm{not}(b)$
$b \Leftrightarrow \mathrm{false}$
$c \Leftrightarrow a \,\&\, c$



SS
:- a succeeds w.r.t. DB

FF
:- b finitely fails w.r.t. DB

$\{a, \mathrm{not}(b), c\} \vDash \mathrm{iff}(DB)$
$\{a, \mathrm{not}(b), \mathrm{not}(c)\} \vDash \mathrm{iff}(DB) \leftarrow$ stable model

Computing SS and FF by DB^d in a vector space

DB:

$a \leftarrow \text{not}(b)$

$c \leftarrow a \;\&\; c$

DB^d:

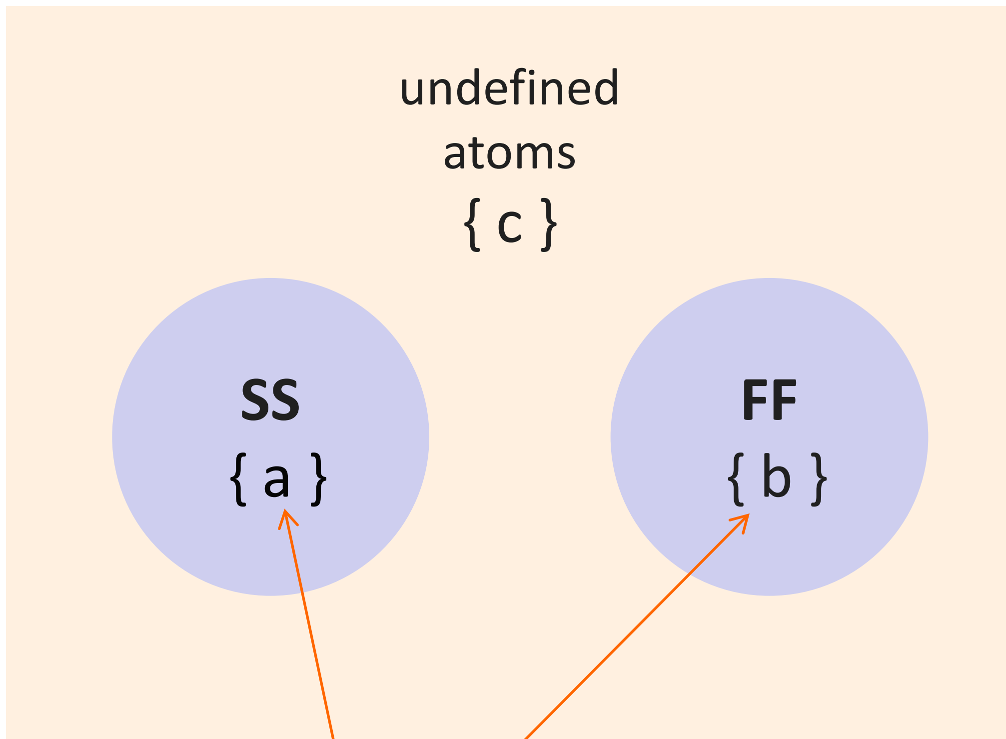
$a \leftarrow nb$

$c \leftarrow a \;\&\; c$

$na \leftarrow b$

$nc \leftarrow na \vee nc$

$nb \leftarrow$



$\mathbf{u}^d = (Q^d \mathbf{u}^d) \geq \theta \Rightarrow \mathbf{M}_{DB^d} = \{a, nb\}$

DB$^d \vdash a, nb$

Compute all supported models of DB

Step-1:

Compute $\mathbf{u}^d_0$, $\mathbf{u}^d_1$... by $\mathbf{u}^d_i = (Q^d \mathbf{u}^d_{i-1})_{\geq\theta}$ and obtain the least fixed point $\mathbf{u}^d = (Q^d \mathbf{u}^d)_{\geq\theta}$ of DB^d (or use (DG'84) algorithm)

Step-2:

Put *udef* = { a | $\mathbf{u}^d(a) = \mathbf{u}^d(na) = 0$ } = undefined atoms in $\mathbf{M}_{DB}{}^3$

Find possible assignments of truth values {**t**(1),**f**(0)}

to atoms in *udef* s.t. the resulting interpretation $\mathbf{u} \models$ iff(DB)

Experiment

- program DB: randomly generated 100 clause in $\{a_1,\dots,a_{100}\}$
- ini_determined_atom Δ_{ini}: fact(true) $\cup$ no_callee_atom(false)
- new_determined_atom Δ_{new} : $(\mathbf{SS} \cup \mathbf{FF}) \setminus \Delta_{ini}$ where $(\mathbf{SS},\mathbf{FF}) = \mathbf{M}_{DB}{}^3$
- reduction_rate = $\#\Delta_{new}/100$

ave. on 10 trials

base atoms $\{a_1,\dots,a_{10}\}$	used as fact $a \leftarrow$	used as tautology $a \leftarrow a$
$\#(\mathbf{SS} \cup \mathbf{FF})$	96.5	57.5
#no_callee_atoms	3.6	12.4
$\#\Delta_{new}$	96.5 – (3.6+10) = 83.9	57.5 - 12.4 = 45.1
reduction_rate	83.9%	45.1%

83.9% of atoms are removed from search space by the introduction of **Step 1** that computes $(\mathbf{SS},\mathbf{FF}) = \mathbf{M}_{DB}{}^3$

Conclusion

- We logically formulate combinatorial problems in such a way that a solution is a supported model of a logic program DB
- However computing supported models is NP-hard
- We proposed to reduce the search space by computing the deterministic part (**SS**,**FF**) separately in **Step 1** via DB^d in a vector space
- Our linear algebraic approach is particularly amenable to parallelism supported by GPU and many cores