The Jobs Puzzle
A Challenge for Logical Expressibility and Automated Reasoning

Stuart C. Shapiro

Department of Computer Science and Engineering
and Center for Cognitive Science
and Center for Multisource Information Fusion
University at Buffalo, The State University of New York
Buffalo, NY 14260-2000
shapiro@buffalo.edu

22 March 2011

Why I'm here

1984 Wos, Overbeek, Lusk, & Boyle publish the Jobs Puzzle.

ca. 1984–93 I start including the Jobs Puzzle
as a standard SNePS demonstration.

Early 2010 I start preparing the Jobs Puzzle
as a demo for my KR-2010 poster,
and realize that it's very difficult for resolution reasoners.

Fall 2010 I decide to discuss the situation
with the commonsense reasoning community.

Dec. 2010 The Commonsense-2011 PC agrees.

The Jobs Puzzle

1. There are four people: Roberta, Thelma, Steve, and Pete.
2. Among them, they hold eight different jobs.
3. Each holds exactly two jobs.
4. The jobs are: chef, guard, nurse, telephone operator, police officer (gender not implied), teacher, actor, and boxer.
5. The job of nurse is held by a male.
6. The husband of the chef is the telephone operator.
7. Roberta is not a boxer.
8. Pete has no education past the ninth grade.
9. Roberta, the chef, and the police officer went golfing together.

Question: Who holds which jobs?

The Solution by Person or Persons Unknown

	Roberta	*Thelma*	*Steve*	*Pete*
chef	x	x	x	x
guard	x	x	x	x
nurse	x	x	x	x
operator	x	x	x	x
police	x	x	x	x
teacher	x	x	x	x
actor	x	x	x	x
boxer	no	x	x	x

“x” = possibility still in doubt

[based on Wos et al. p 56]

The Solution by Program or Programs Known (OTTER)

Example clauses from

`http://www.mcs.anl.gov/~wos/mathproblems/jobs.txt`

(accessed 3/2/2011)

```
POSSJOBS(l(pj(Roberta,chef),
           l(pj(Roberta,guard),
             l(pj(Roberta,nurse),
               l(pj(Roberta,clerk),
                 l(pj(Roberta,police),
                   l(pj(Roberta,teacher),
                     l(pj(Roberta,actor),
                       l(pj(Roberta,boxer),
                         end))))))))).
-POSSJOBS(l(pj(x,y),l(pj(x,z),end)))
   |  EQUALP(x,w)
   | EQUAL(pj(w,y),crossed).
```

Wos et al. Assessment

“Make no mistake, the representation of the problem to an automated reasoning program is sometimes difficult and sometimes tedious.”
[p.63]

The Challenge

- Represent the Jobs Puzzle
- to an automated reasoning program,
  suitable for general-purpose commonsense reasoning,
- in a non-difficult, non-tedious way,
- by a series of logical formulae
  that adhere closely to the English statements of the puzzle
  and the allowed immediate inferences,
- and have that automated reasoning program
  solve the puzzle quickly.

Solutions by:

TPTP Participants (formalized by them)

SNePS (formalized by me)

Lparse/Smodels (formalized by me)

TPTP Overview

- PUZ019-1 in version 5.1.0 of TPTP (Thousands of Problems for Theorem Provers)
- 64 Clauses
  - 4 Non-Horn clauses
- Solved by 20 of 29 recorded attempts
- Will show here as standard FOL

SNePS Overview

- SNePS 2.7.1
- SNePSLOG
- Natural deduction
- Sound, not complete
- No modus tollens

Lparse/Smodels Overview

- Lparse front-end
- Extended logic programming syntax
- Smodels: Stable model semantics
- Finds satisfying models of ground clauses

Unique Names

$\forall(x)(equal\_people(x,x) \wedge equal\_jobs(x,x))$
$\forall(x,y)(equal\_people(x,y) \Rightarrow equal\_people(y,x))$
$\forall(x,y)(equal\_jobs(x,y) \Rightarrow equal\_jobs(y,x))$

$\neg equal\_people(roberta, thelma)$ ... (6 clauses)
$\neg equal\_jobs(chef, guard)$ ... (28 clauses)

Built in in SNePS

Built in in Lparse/Smodels

Set/Conjunctive Arguments

SNePS:

$P(\{a_1, \ldots, a_n\})$ $\vdash P(a_i)$, $1 <= i <= n$

Lparse/Smodels:

$P(a_1; \ldots; a_n)$

abbreviates conjunction of $P(a_1)$, and ..., and $P(a_n)$

Counting Propositions & Instances

SNePS:

`nexists(`i, j, k`) (`x`) (`$P(x)$`: `$Q(x)$`))`

k individuals satisfy $P(x)$,
and, of them,
at least i and at most j also satisfy $Q(x)$

Lparse/Smodels:

i { $R(x, y)$`[:`$P(x)$`], `$Q(z)$ } j

The number of literals that satisfy $R(x, y)$ plus those that satisfy $Q(z)$
[(assuming that the first argument of each R satisfies $P(x)$]
is between i and j inclusive.

1. *jp: There are four people: Roberta, Thelma, Steve, and Pete.*

$$\forall x(has\_job(roberta, x) \lor has\_job(thelma, x) \\ \lor has\_job(pete, x) \lor has\_job(steve, x))$$

```
Person({Roberta, Thelma, Steve, Pete}).
```

```
person(roberta;thelma;steve;pete).
```

inf: "if the four names did not clearly imply the sex of the people, [the puzzle] would be impossible to solve."

$\forall x((male(x) \lor female(x)) \land \neg(male(x) \land female(x)))$

```
:- person(X), male(X), female(X).
```

$female(roberta) \land female(thelma)$
$male(steve) \land male(pete)$

```
Female({Roberta, Thelma}).
Male({Steve, Pete}).
```

```
female(roberta; thelma).
male(steve; pete).
```

2. *jp: Among [the people], they hold eight different jobs.*
4. *jp: The jobs are: chef, guard, nurse, telephone operator, police officer (gender not implied), teacher, actor, and boxer.*

$$\begin{aligned}&\forall x(has\_job(x, chef) \vee has\_job(x, guard)\\&\quad \vee has\_job(x, nurse) \vee has\_job(x, operator)\\&\quad \vee has\_job(x, police) \vee has\_job(x, teacher)\\&\quad \vee has\_job(x, actor) \vee has\_job(x, boxer))\end{aligned}$$

3. *jp: Each holds exactly two jobs.*

$$\begin{aligned}&\forall(x, y, z, u)(has\_job(z, y) \wedge has\_job(z, x) \wedge has\_job(z, u)\\&\qquad \Rightarrow equal\_jobs(x, y) \vee equal\_jobs(u, y) \vee equal\_jobs(u, x)\end{aligned}$$

2. *jp: Among [the people], they hold eight different jobs.*
3. *jp: Each holds exactly two jobs.*

```
all(p)(Person(p)
       => nexists(2,2,8)(j)(Job(j):  hasJob(p,j)).
2 {hasJob(X,Y): job(Y)} 2 :- person(X).
```

4. *jp: The jobs are: chef, guard, nurse, telephone operator, police officer (gender not implied), teacher, actor, and boxer.*

```
Job({chef, guard, nurse, operator, police, teacher,
     actor, boxer}).
job(chef; guard; nurse; operator; police; teacher;
    actor; boxer).
```

inf: "No job is held by more than one person."

$$\forall(x, y, z)(has\_job(x, z) \land has\_job(y, z) \Rightarrow equal\_people(x, y))$$

```
all(j)(Job(j)
       => nexists(1,1,4)(p)(Person(p):hasJob(p,j))).
```

```
1 {hasJob(X,Y): person(X)} 1 :- job(Y).
```

5. *jp: The job of nurse is held by a male.*

$\forall x(has\_job(x, nurse) \Rightarrow male(x))$

```
all(x)(Female(x)=>~hasJob(x,nurse)).
```

```
male(X) :- person(X), hasJob(X,nurse).
```

6. *jp: The husband of the chef is the telephone operator.*

$$\forall x(has\_job(x, chef) \Rightarrow \forall y(husband(x, y) \Leftrightarrow has\_job(y, operator)))$$

```
hasJob(X,operator) :- person(X;Y),
                      hasJob(Y,chef),
                      hasHusband(Y,X).*
hasHusband(Y,X) :- person(X;Y),
                   hasJob(Y,chef),
                   hasJob(X,operator).
```

*Not needed for solution

6. *jp: The husband of the chef is the telephone operator.*
inf: "the implicit fact that husbands are male"
inf: since the chef has a husband, she must be female.

$\forall(x, y)(husband(x, y) \Rightarrow female(x) \wedge male(y))$

```
all(w)(Female(w)=>~hasJob(w,operator)).
all(m)(Male(m)=>~hasJob(m,chef)).
```

```
2 {female(X), male(Y)} 2 :- person(X;Y),
                            hasHusband(X,Y).
```

7. *jp: Roberta is not a boxer.*

$\neg has\_job(roberta, boxer)$

```
~hasJob(Roberta,boxer).
```

```
:- hasJob(roberta, boxer).
```

8. *jp: Pete has no education past the ninth grade.*

$\neg educated(pete)$

```
~educated(Pete).
```

```
:- educated(pete).
```

8. *inf: "the jobs of nurse, police officer, and teacher each require more than a ninth-grade education."*

$$\forall x(\mathit{has\_job}(x, \mathit{nurse}) \lor \mathit{has\_job}(x, \mathit{police}) \lor \mathit{has\_job}(x, \mathit{teacher}) \Rightarrow \mathit{educated}(x))$$

```
all(x)(~educated(x)
       => nor{hasJob(x, nurse),
              hasJob(x, police),
              hasJob(x, teacher)}).
```

```
educated(X) :-
       1 {hasJob(X,nurse),
          hasJob(X,police), hasJob(X,teacher)} 2,
       person(X).
```

9. *jp: Roberta, the chef, and the police officer went golfing together. inf: "Thus, we know that Roberta is neither the chef nor the police officer."*

$\neg(\mathit{has\_job}(\mathit{roberta}, \mathit{chef}) \vee \mathit{has\_job}(\mathit{roberta}, \mathit{police}))$

```
nor{hasJob(Roberta, chef), hasJob(Roberta, police)}.
```

```
0 {hasJob(roberta, chef), hasJob(roberta, police)} 0.
```

inf: "Since they went golfing together, the chef and the police officer are not the same person."

$\forall x \neg(has\_job(x, chef) \wedge has\_job(x, police))$

```
all(p)(Person(p)
       => nand{hasJob(p,chef), hasJob(p,police)}).
```

```
0{hasJob(X,chef), hasJob(X,police)}1 :- person(X).
```

jp: Question: Who holds which jobs?

$$\exists(x1, x2, x3, x4, x5, x6, x7, x8)(has\_job(x1, chef) \wedge has\_job(x2, guard) \wedge has\_job(x3, nurse) \wedge has\_job(x4, operator) \wedge has\_job(x5, police) \wedge has\_job(x6, teacher) \wedge has\_job(x7, actor) \wedge has\_job(x8, boxer))$$

```
ask hasJob(?p, ?j)?
```

```
#hide.
#show hasJob(X,Y).
```

The answers:

SZS answers short [[thelma, roberta, steve, pete,
steve, roberta, pete, thelma]]
0.182411 seconds of total run time

(by SNARK)

```
wff111!:  hasJob(Thelma,boxer)
wff101!:  hasJob(Pete,operator)
wff99!:  hasJob(Pete,actor)
wff87!:  hasJob(Steve,nurse)
wff85!:  hasJob(Roberta,guard)
wff83!:  hasJob(Roberta,teacher)
wff28!:  hasJob(Thelma,chef)
wff24!:  hasJob(Steve,police)

CPU time :  0.19
```

```
Answer: 1
Stable Model: hasJob(pete,operator)
              hasJob(pete,actor)
              hasJob(steve,nurse)
              hasJob(steve,police)
              hasJob(thelma,chef)
              hasJob(thelma,boxer)
              hasJob(roberta,guard)
              hasJob(roberta,teacher)

Duration: 0.000
```

TPTP clause version

- Still somewhat tedious
- Some formalizations more clever than direct translations
- Uses non-Horn clauses
- 9 of 29 recorded attempts failed
- Success required careful choice of strategies

SNePS and Lparse/Smodels versions benefit from

- Unique Names Assumption
- Set/Conjunctive arguments
- Numerical Quantifier/Cardinality constraints

SNePS version

- Natural Deduction and incompleteness provided focus
- Contrapositives occasionally required
- Quite close translation

Lparse/Smodels version

- Constraint-satisfaction model-finding
- Limited to ground predicate logic
- Very close translation

Try your favorite system!

Thanks to:

- Mark Stickel
- Raphael Finkel
- SNePS Research Group, especially
  - William J. Rapaport
  - Jonathan P. Bona
  - Michael Kandefer
  - Michael Prentice
  - A. Patrice Seyed
- US Army Research Office (ARO)