ASP-Prolog for Negotiation Among Dishonest Agents

Ngoc-Hieu Nguyen[1], Tran Cao Son[1],
Enrico Pontelli[1], Chiaki Sakama[2]
[1]New Mexico State University
[2]Wakayama University

Outline

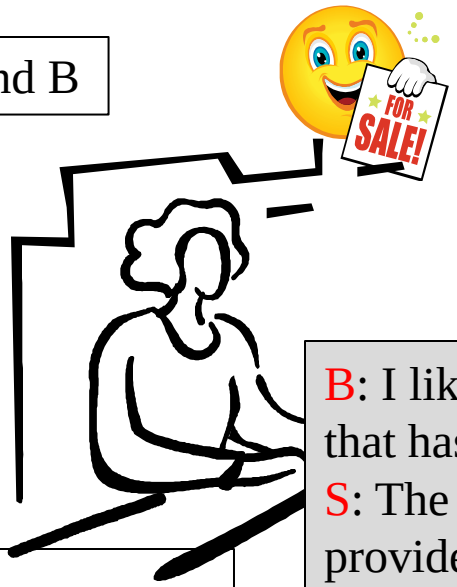
- Motivation
- Formalism
- Implementation
- Conclusions and future work

Aim

- Building automated negotiation agents
  - Methodologies for the design and development of negotiations agents who employ a knowledge base (in some form) in their negotiation
  - Environment for the development of negotiation agents

Example

Products: A and B



Not a student, pay cash
Look for good quality product
Hate advertisement

A: made by C
A: discount for students
B: made by D
B: ¬good
B: lowest price for joining the mailing list

B: I like a digital camera by the maker C. I want to get one that has good quality at a discount price.
S: The product A is made by C and has good quality. We provide a discounted price to students. **[MISLEADING]**
B: I am not a student.
S: The product B by the maker D is on bargain sale. It has good quality and is provided at a discount price for every customer paying in cash. **[LYING]**
B: I do not want products by the maker D at the price.
S: If you join our mailing list, we can provide the product at the lowest price.
B: I'd like to join the list and buy it at the price. **[LYING]**

Agents in Negotiation

- Preference:
  - Seller: higher price
  - Buyer: lower price, certain specification
- Incomplete information (assumptions):
  - Seller does not have the information about buyer's preference about product, financial status, etc. and makes assumptions about it
  - Buyer does not know about the available products, their quality
- Disinformation: seller/buyer misleads/lies
- Goal change: both can change goals

Results

- Formalizing negotiation among dishonest agents (IJCAI, 2011)
- Implementation of a platform for negotiation (a prototype, this paper)

Formalizing Negotiation Among Dishonest Agents

Dishonest agents = Agents who use disinformation

Negotiation among dishonest agents

= Negotiation + Disinformation

Formalizing Negotiation Among Dishonest Agents

- Representation language
  - abductive logic program with preferences (assumptions for incomplete information and preference)
  - extending to deal with disinformation
    - misleading: stating a fact is true while its truth value is unknown in every belief set
    - lying: stating a fact is true while it is believed to be false
- Knowledge based agents
  - each agent is encoded with an abductive logic program with disinformation, preferences, and goals

Seller Negotiation Knowledge Base

¬qualityB :- productB.
makerC :- productA.
makerD :- productB.
bargain :- productB.
sale :- productA, high_pr.
sale :- productA, low_pr.
sale :- productB, high_pr.
sale :- productB, low_pr.
sale :- productB, lowest_pr.
productA. productB.
senior_custome :- age ≥ 65.
student_customer :- student.
:- high_pr, low_pr.
:- high_pr, lowest_pr.
:- low_pr, lowest_pr.
:- not sale.
prefer(n1, n2). …

n1: high_pr.
n2: low_pr :- senior_customer.
n3: low_pr :- student_customer.
n4: low_pr :- bargain, cash.
n5: lowest_pr :- mail_list, cash.

student. cash. mail_list. age ≥ 65.

qualityB. (lies)

qualityA. (bullshit)

high_pr > low_pr > lowest_pr. (goal)

Dealing with Disinformation

¬qualityB :- productB.
makerC :- productA.
makerD :- productB.
bargain :- productB.
sale :- productA, high_pr.
sale :- productA, low_pr.
sale :- productB, high_pr.
sale :- productB, low_pr.
sale :- productB, lowest_pr.
productA. productB.
senior_custome :- age ≥ 65.
student_customer :- student.
:- high_pr, low_pr.
:- high_pr, lowest_pr.
:- low_pr, lowest_pr.
:- not sale.
prefer(n1, n2). …

n1: high_pr.
n2: low_pr :- senior_customer.
n3: low_pr :- student_customer.
n4: low_pr :- bargain, cash.
n5: lowest_pr :- mail_list, cash.

student. cash. mail_list. age ≥ 65.

If qualityB is used, some rules from KB must be removed

qualityB.

qualityA. (bullshit)

high_pr > low_pr > lowest_pr. (goal)

Formalizing Negotiation Among Dishonest Agents

- Negotiation specific: exchanges between agents
  - what is a proposal?
  - when is a proposal acceptable, rejectable, or negotiable?
  - what is the best proposal for a give negotiation goal?

Proposal

¬qualityB :- productB.
makerC :- productA.
makerD :- productB.
bargain :- productB.
sale :- productA, high_pr.
sale :- productA, low_pr.
sale :- productB, high_pr.
sale :- productB, low_pr.
sale :- productB, lowest_pr.
productA. productB.
senior_custome :- age ≥ 65.
student_customer :- student.
:- high_pr, low_pr.
:- high_pr, lowest_pr.
:- low_pr, lowest_pr.
:- not sale.
prefer(n1, n2). …

n1: high_pr.
n2: low_pr :- senior_customer.
n3: low_pr :- student_customer.
n4: low_pr :- bargain, cash.
n5: lowest_pr :- mail_list, cash.

student. cash. mail_list. age ≥ 65.

qualityB. (lies)

qualityA. (bullshit)

an utterance consisting of the goal,
the assumptions about opponent, and additional conditions
e.g <{high_pr}, ∅, {productA}>
or <{lowest_pr},{cash, mail_list},{productB, makerD,qualityB}>

Classification of Proposals

¬qualityB :- productB.
makerC :- productA.
makerD :- productB.
bargain :- productB.
sale :- productA, high_pr.
sale :- productA, low_pr.
sale :- productB, high_pr.
sale :- productB, low_pr.
sale :- productB, lowest_pr.
productA. productB.
senior_custome :- age ≥ 65.
student_customer :- student.
:- high_pr, low_pr.
:- high_pr, lowest_pr.
:- low_pr, lowest_pr.
:- not sale.
prefer(n1, n2). …

n1: high_pr.
n2: low_pr :- senior_customer.
n3: low_pr :- student_customer.
n4: low_pr :- bargain, cash.
n5: lowest_pr :- mail_list, cash.

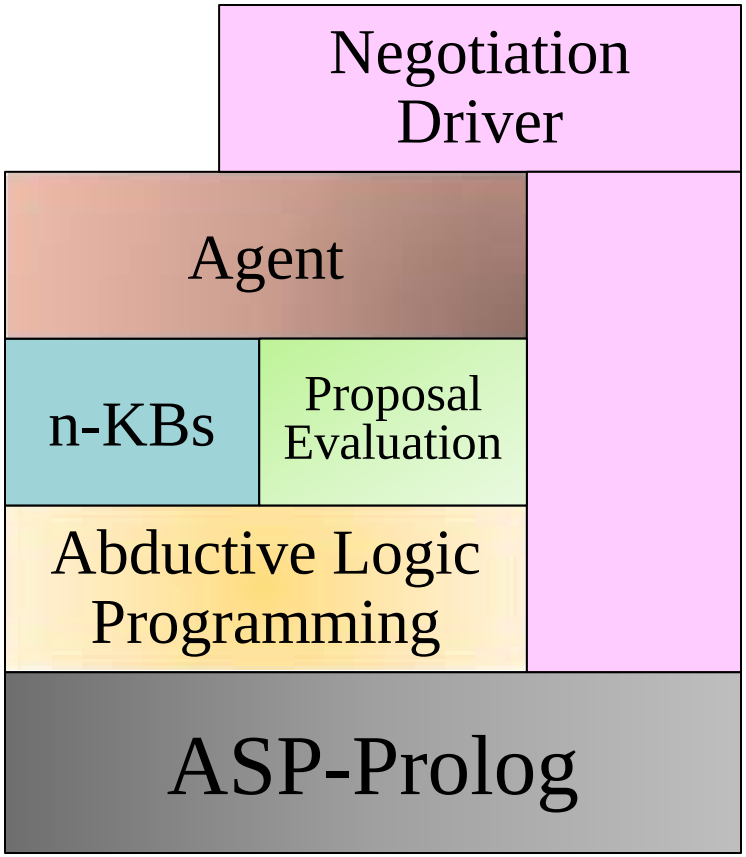
student. cash. mail_list. age ≥ 65.

qualityB. (lies)

qualityA. (bullshit)

consistency of receiving information with respect to the KB
<{high_pr}, {productA}, ∅ > acceptable
<{low_pr}, {productA, qualityA, makerC}, ∅> negotiable
<{lowest_pr}, {productA, qualityA, makerC}, ∅> rejectable

Implementation Overview



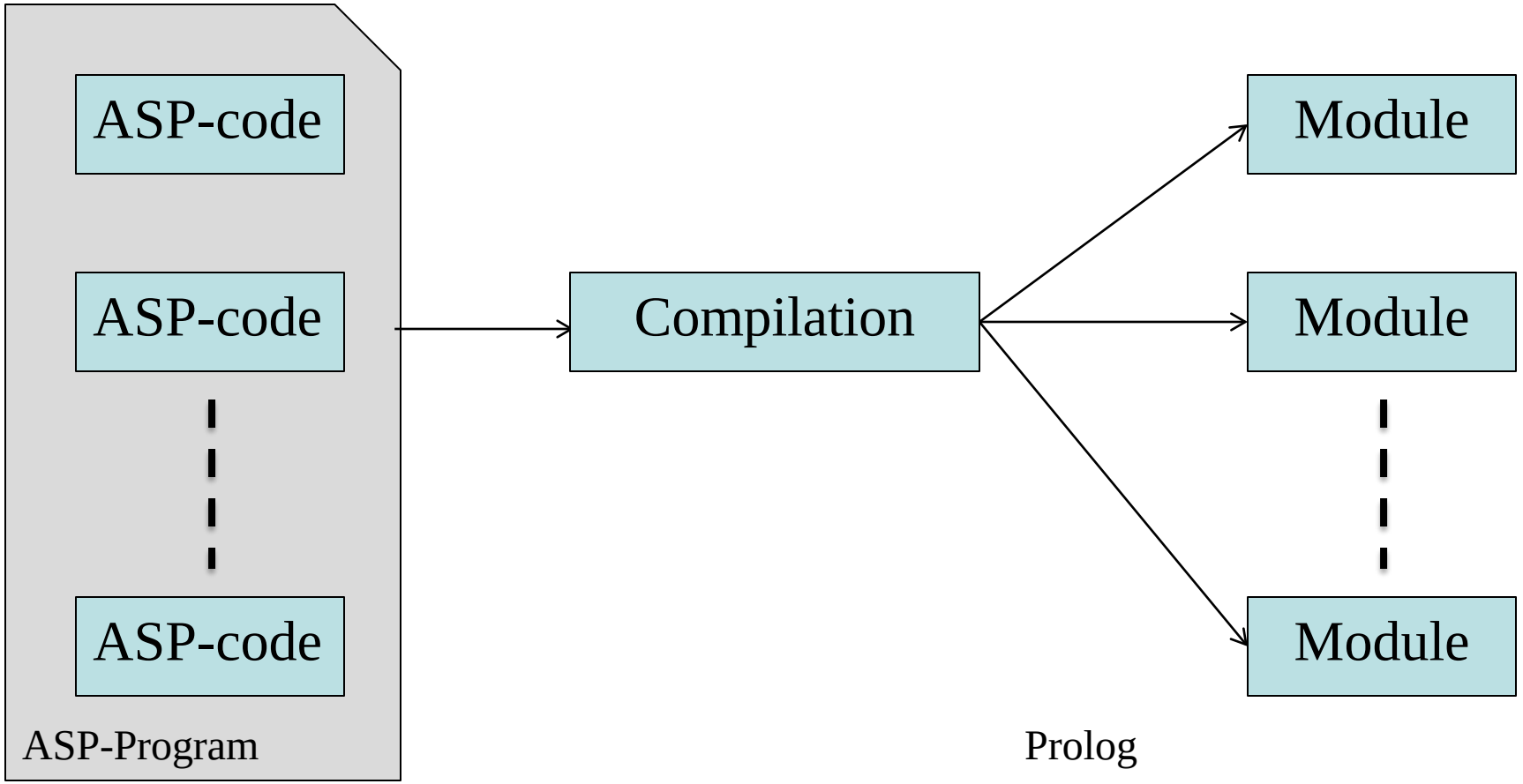
This paper

Previously developed

ASP-Prolog

- Extending Sicstus-Prolog module system with ASP-modules
- Ability to access information from other modules via the intended models
  - Prolog: least Herbrand model
  - ASP: answer set semantics
- Ability to access content of modules

Internal Implementation



ASP-Prolog

q1 (ASP)

:- asp_module.

r :- not s.

h.

q1
:- module q11

q11
h.

q2 (ASP)

:- asp_module.

:- use_module(q1).

a :- not s.

s :- not a.

q :- q1:h

q2
:- … q21
: …. q22

q21
s.
q.

q22
a.
q.

p1 (Prolog)

:-use_module(q1).

:-use_module(q2).

p :- q1:r.

s :- q2:a.

t :- q2:model(X), X:a.

v :- q1:assert(s), q1:r.

:- p1:t? yes

:- p1:q? yes

:- p1:v? no

ALP with Preferences

- Extending ASP-Prolog to consider ALP: compiling abducible rules (assumptions) to normal rules with a new predicate ok(r) and use choice rules to activate/deactivate the rules
- Computing most preferred belief sets: providing predicates for accessing most preferred beliefs sets

Negotiation Agents

- Each agent specification consists of 5 parts:
  - regular program
  - assumptions
  - lies
  - bullshits
  - goals and preferences
- Each agent is compiled into an ALP module
  - each belief set of the new ALP module could be used to generate several proposals

Seller Program

¬qualityB :- productB.
makerC :- productA.
makerD :- productB.
bargain :- productB.
sale :- productA, high_pr.
sale :- productA, low_pr.
sale :- productB, high_pr.
sale :- productB, low_pr.
sale :- productB, lowest_pr.
productA. productB.
senior_custome :- age ≥ 65.
student_customer :- student.
:- high_pr, low_pr.
:- high_pr, lowest_pr.
:- low_pr, lowest_pr.
:- not sale.
prefer(n1, n2). …

n1: high_pr.
n2: low_pr :- senior_customer.
n3: low_pr :- student_customer.
n4: low_pr :- bargain, cash.
n5: lowest_pr :- mail_list, cash.

student. cash. mail_list. age ≥ 65.

qualityB. (lies)

qualityA. (bullshit)

high_pr > low_pr > lowest_pr. (goal)

Negotiation Engine

- Implementation of predicates for
  - computing proposals
  - selecting proposals
  - evaluating proposals
- Implementation of strategies for selecting proposals
  - non-repeated (an agent can not repeat a proposal)
  - honest > bullshits > lies

Conclusions and Future Works

- Prototype of a negotiation platform
  - design
  - implementation on top of ASP-Prolog
- Future work
  - strategies
  - negotiation server: distributed, message exchanges
  - real-world applications: multi-agent planning