Combining Answer Sets of Nonmonotonic Logic Programs

Chiaki Sakama
Wakayama University

Katsumi Inoue
National Institute of Informatics

Compositionality of Logic Programs

- A desirable feature for declarative knowledge representation languages is compositionality in its semantics.
- A semantics is compositional if the meaning of a program can be obtained from the meaning of its components.

Compositionality of Logic Programs

- Semantics of LPs is not compositional wrt the union of programs even for definite programs.
- For instance, two programs $P_1 = \{ p \leftarrow q \}$ and $P_2 = \{ q \leftarrow \}$ have the least models Φ and $\{q\}$, respectively. But the least model of $P_1 \cup P_2$ is not obtained by the composition of Φ and $\{q\}$.
- To solve the problem, a number of different compositional semantics for definite programs are proposed.

Combining Knowledge in Multi-Agent Systems

- In MAS different knowledge/belief of agents are combined/coordinated to solve problems cooperatively/collaboratively.
- Individual agents in MAS have incomplete information, so combining multiple knowledge is formulated as the problem of composing different nonmonotonic theories.

Difficulty of Composing Nonmonotonic Theories

- “Nonmonotonic reasoning and compositionality are intuitively orthogonal issues that do not seem easy to be reconciled. Indeed the semantics for extended logic programs are typically non-compositional w.r.t. program union” [Brogi, 2004].

Example

- There is a trouble in a system which consists of three components c1, c2, and c3.
- After some diagnoses, an expert E1 concludes that the trouble would be caused by either c1 or c2. Another expert E2 concludes that it would be caused by either c2 or c3.
- E1 has no knowledge on the component c3, and E2 has no knowledge on c1.

Example – cont.

- Two experts' diagnoses are encoded as:

  E1: c1 ; c2 ←

  E2: c2 ; c3 ←

- Merging these programs, E1 ∪ E2 has two answer sets: { c2 } and { c1, c3 }.
- The first one is the common solution, while the second one is cooperative. Two solutions have different grounds and would be acceptable to each expert.

Example – cont.

- E1 knows that c1 is older than c2, so c1 is more likely to disorder. On the other hand, E2 knows that c2 is more fragile than c3 and is more likely to cause the trouble. Two experts then modify their diagnoses as:

  E1': $c1 \leftarrow$ *not* $c2$, $c2 \leftarrow \neg\ c1$

  E2': $c2 \leftarrow$ *not* $c3$, $c3 \leftarrow \neg\ c2$

- Merging two programs, E1' $\cup$ E2' has the single answer set: { c2 }, which reflects the result of diagnoses of E2' but does not reflect E1'.

Problem

- E1’ puts weight on c1 relative to c2, and E2’ puts weights on c2 relative to c3.
- Simple merging has the effect of preferring c2 to c1 as c2 is included in a relatively lower stratum than c1.
- However, there is no reason to conclude c2 as the plausible solution. Because the local preference in E1’ or E2’ does not necessarily imply the global preference in E1’ ∪ E2’.

Purpose

- Composition of nonmonotonic theories is not achieved by simple program union.
- The problem is then how to build a compositional semantics of NM theories.
- In this study we consider composition of extended disjunctive programs (EDP) under the answer set semantics.

Extended Disjunctive Program

- A program consists of rules of the form:

  $L_1 ; \ldots ; L_l \leftarrow L_{l+1}, \ldots, L_m, \mathit{not}\ L_{m+1}, \ldots, \mathit{not}\ L_n$

  where L_i is a literal and *not* represents NAF.
  A program is NAF-free if it contains no NAF.
- For each rule r of the above form,
  $\mathit{head}(r) = \{ L_1, \ldots, L_l \}$, $\mathit{body}^+(r) = \{ L_{l+1}, \ldots, L_m \}$, and $\mathit{body}^-(r) = \{ L_{m+1}, \ldots, L_n \}$.

Answer Sets

- For an NAF-free EDP P, a set S is an answer set of P if it is a minimal set satisfying every rule in P and is logically closed (i.e., S=Lit if S is contradictory).
- For any EDP P, a set S is an answer set of P if S is an answer set of the reduct ^{S}P. Here, the rule $head(r) \cap S \leftarrow body^{+}(r)$ is included in ^{S}P if $body^{+}(r) \subseteq S$ and $body^{-}(r) \cap S = \phi$ for any rule r in the ground instantiation of P.

Remark

- The definition of reduct is different from the original one in [Gelfond&Lifschitz, 1991]. In GL-reduction, the rule head(r) $\leftarrow$ body+(r) is included in the reduct P^S if body-(r) $\cap S = \phi$.
- Two reducts produce the same answer sets, i.e., for any EDP P, S is an answer set of ${}^SP$ iff S is an answer set of P^S.

Example

P: $p\ ;\ q \leftarrow,\quad q \leftarrow p,\quad r \leftarrow \mathit{not}\ p\ .$

For S={ q, r }, P^S becomes

P^S : $p\ ;q \leftarrow,\quad q \leftarrow p,\quad r \leftarrow,$

while ${}^SP$ becomes

${}^SP$: $q \leftarrow,\quad r \leftarrow.$

Two reducts produce the same answer set S.

Combining Answer Sets

- Let S and T be two sets of literals. Then, define

$$S \uplus T = \begin{cases} S \cup T, & \text{if } S \cup T \text{ is consistent;} \\ Lit, & \text{otherwise.} \end{cases}$$

- Let AS(P) be the set of answer sets of P. Then, define

$$AS(P_1) \uplus AS(P_2) = \{ S \uplus T \mid S \in AS(P_1) \text{ and } T \in AS(P_2) \}.$$

Compositional Semantics

- Given two consistent programs P_1 and P_2 , the program Q satisfying

  $$AS(Q) = \min(AS(P_1) \uplus AS(P_2))$$

  is called a composition of P_1 and P_2.
- The set AS(Q) is called the compositional semantics of P_1 and P_2 .

Example

For $AS(P_1) = \{ \{p\} , \{q\} \}$ and
$AS(P_2) = \{ \{p\}, \{r\} \}$,
the compositional semantics becomes
$AS(Q) = \{\{p\}, \{q, r\}\}$.

Properties

- Let P_1 and P_2 be two consistent programs, and Q a result of composition. Then, for any $S \in AS(Q)$, there is $T \in AS(Pi)$ for i=1,2 such that $T \subseteq S$.

† Every answer set in the compositional semantics extends some answer sets of the original programs.

Properties

Def. Let P_1 and P_2 be two consistent programs, and Q a result of composition. When $AS(Q)=AS(P_1)$, P_1 absorbs P_2.

† When one program absorbs another program, the compositional semantics coincides with one of the original programs.

- P_1 absorbs P_2 iff for any $S \in AS(P_1)$ there is $T \in AS(P_2)$ such that $T \subseteq S$.

Properties

Def. A literal L is a consequence of credulous/skeptical reasoning in P (written as $L \in crd(P)$ / $L \in skp(P)$) if L is included in some/every answer set of P.

- Let P_1 and P_2 be two consistent programs. When a result Q of composition is consistent,
  1. $crd(Q) = crd(P_1) \cup crd(P_2)$;
  2. $skp(Q) = skp(P_1) \cup skp(P_2)$.

† A consistent compositional semantics combines skeptical consequences of P_1 and P_2 , and any information included in an answer set of Q has its origin in an answer set of either P_1 or P_2 .

Properties

† Composition of consistent programs may become inconsistent.

ex) Composing $AS(P_1)=\{\{p\}\}$ and $AS(P_2)=\{\{\neg p\}\}$ becomes $AS(Q)=\{$ Lit $\}$.

- Let P_1 and P_2 be consistent programs, and Q a result of composition. Then, Q is consistent iff there are $S \in AS(P_1)$ and $T \in AS(P_2)$ such that $S \cup T$ is consistent.

Composing Programs

- Given programs P_1 ,..., P_k , define
  $P_1 ; \cdots ; P_k = \{$
  $head(r_1) ; \cdots ; head(r_k) \leftarrow body(r_1),...,body(r_k)$
  $\mid r_i \in P_i \quad (1 \leq i \leq k) \}$.
- Let P_1 and P_2 be two consistent programs s.t. $AS(P_1)=\{ S_1,...,S_m \}$ and $AS(P_2)=\{ T_1,...,T_n \}$. Then, define
  $P_1 \circledcirc P_2 = R(S_1,T_1); \cdots ; R(S_m,T_n)$
  where $R(S,T)={}^{S}P_1 \cup {}^{T}P_2$ and $R(S_1,T_1),...,R(S_m,T_n)$ is any enumeration of the $R(S_i,T_j)$'s for $S_i \in AS(P_1)$ $(i=1,...,m)$ and $T_j \in AS(P_2)$ $(j=1,...,n)$.

Example (1)

P_1: $p \leftarrow not\ q$, $q \leftarrow not\ p$, $s \leftarrow p$
P_2: $p \leftarrow not\ r$, $r \leftarrow not\ p$
where $AS(P_1) = \{ \{p,s\}, \{q\} \}$ and
$AS(P_2) = \{ \{p\}, \{r\} \}$.
There are four R(S,T)'s such that

R({p,s},{p}): $p \leftarrow$, $s \leftarrow p$
R({p,s},{r}): $p \leftarrow$, $s \leftarrow p$, $r \leftarrow$
R({q},{p}): $q \leftarrow$, $p \leftarrow$
R({q},{r}): $q \leftarrow$, $r \leftarrow$

Example (2)

Then, $P_1 ◎ P_2$ contains

$p;q \leftarrow$, $p;r \leftarrow$, $p;q;r \leftarrow$, $q;s \leftarrow p$,
$q;r;s \leftarrow p$, $p;q;s \leftarrow p$, $p;r;s \leftarrow p$.

Among them, yellow rules are redundant and eliminated, the result then becomes

$p;q \leftarrow$, $p;r \leftarrow$, $q;s \leftarrow p$.

Properties

- The operation ◎ is commutative and associative.
- For two consistent programs P_1 and P_2 ,

$$AS(P_1 \circledcirc P_2) = \min(AS(P_1) \uplus AS(P_2)).$$

Composition vs. Merging

- For two consistent NAF-free EDPs P_1 and P_2 , if $P_1 \cup P_2$ is consistent, $P_1 \circledcirc P_2$ is consistent.
- For two consistent NAF-free ELPs P_1 and P_2 , $P_1 \circledcirc P_2 \subseteq P_1 \cup P_2$.
- For two consistent NAF-free ELPs P_1 and P_2 , $U \subseteq V$ holds for the answer set U of $P_1 \circledcirc P_2$ and the answer set V of $P_1 \cup P_2$.

Compositional Semantics for Multi-Agent Coordination.

Let P_1 and P_2 be two consistent programs, and Q a result of composition. Then, any answer set $S \in AS(Q)$ is conservative if it satisfies every rule in $P_1 \cup P_2$.

Example

P_1: $p \leftarrow not\ q$, $q \leftarrow not\ p$, $s \leftarrow p$
P_2: $p \leftarrow not\ r$, $r \leftarrow not\ p$
where $AS(P_1) = \{ \{p,s\}, \{q\} \}$ and
$AS(P_2) = \{ \{p\}, \{r\} \}$.
The compositional semantics is
$AS(Q)=\{\{p,q\}, \{p,s\}, \{q,r\}\}$.
Among them, $\{p,s\}$ and $\{q,r\}$ satisfy every rule in $P_1 \cup P_2$, so they are conservative.
Note: $\{p,q\}$ does not satisfy $s \leftarrow p$ in P_1.

Notes

- Conservative answer sets are acceptable to each agent because they satisfy the original programs.
- Conservative answer sets do not always exist in compositional semantics.
- We introduce a permissible version of compositional semantics that retains persistent beliefs of each agent in coordination.

Persistent Beliefs

- Persistent Beliefs in a program P are distinguished as PB $\subseteq$ P where PB is the set of rules that should be satisfied by the compositional semantics.

Permissible Composition

- Let P_1 and P_2 be two consistent programs, and PB_1 and PB_2 their persistent beliefs, respectively. A program Ω is called permissible composition of P_1 and P_2 if it satisfies the condition:

$$AS(\Omega) = \{ S \mid S \in \min(AS(P_1) \uplus AS(P_2)) \text{ and } S \text{ satisfies } PB_1 \cup PB_2 \}.$$

- The set $AS(\Omega)$ is called the permissible compositional semantics of P_1 and P_2.
- Any answer set in $AS(\Omega)$ is called a permissible answer set.

Properties

- The permissible compositional semantics reduces to the compositional semantics when $PB_1 \cup PB_2 = \emptyset$.
- Conservative answer sets are permissible answer sets with $PB_1 \cup PB_2 = P_1 \cup P_2$.
- Every permissible answer set satisfies persistent beliefs of each agent, and extends some answer sets of an agent by additional information of another agent.

Program Composition for Permissible Semantics

- Let P_1 and P_2 be two consistent programs, and Ω a result of permissible composition. Then, $AS(\Omega)$
  $= AS((P_1 \circledcirc P_2) \cup IC(PB_1) \cup IC(PB_2))$,
  where IC(PB)={$\leftarrow$ *body(r), not_head(r)*
  | *head(r)* $\leftarrow$ *body(r)* $\in$ PB }
  and *not_head(r)* = { *not* L_1 ,..., *not* L_l } for *head(r)*={ L_1 ,..., L_l }.

Example

P_1: $p \leftarrow not\ q$, $q \leftarrow not\ p$, $s \leftarrow p$,
P_2: $p \leftarrow not\ r$, $r \leftarrow not\ p$.
Let PB1={ $s \leftarrow p$ } and PB2= Φ . Then,
$(P_1 \circledcirc P_2) \cup IC(PB_1) \cup IC(PB_2)$ becomes
$p;q \leftarrow$, $p;r \leftarrow$, $q; s \leftarrow p$, $\leftarrow p, not\ s$,
which has two permissible answer sets
$\{p,s\}$ and $\{q,r\}$.

Final Remarks

- Simple union of different programs does not reflect the meaning of individual programs.
- We then took an approach of retaining belief of each agent and combine answer sets of different programs.
- Program composition should be distinguished from revision or update, where one of the two information sources is known more reliable.

Final Remarks

- From the viewpoint of answer set programming, program composition is considered a program development under a specification that requests a program reflecting the meanings of two or more programs.
- Future work includes investigation of other types of program composition for multi-agent coordination, and their characterization in computational logic.