Constructing Consensus Logic Programs

Chiaki Sakama
Wakayama University

Katsumi Inoue
National Institute of Informatics

Motivations

- Logic Programming provides a formal language for representing knowledge and belief of an agent.
- The declarative semantics of a program is given by a set of canonical models.
- What are the suitable conclusions drawn from a collection of programs?
- How to synthesize a program having such a collective semantics?

Multi-Agent Systems

- Those problems are especially important when there exist more than one agent in multi-agent environments.
- In MAS, multiple agents have different beliefs and intensions.
- To make decision and act as a whole community, agents seek consensus which is acceptable to every individual agent.

MAS in LP

- We suppose MAS in which each agent has a knowledge base represented by a logic program under the answer set semantics.
- Answer sets represent sets of beliefs of an agent on the basis of a program.
- Different agents have different collections of answer sets in general.
- We capture building consensus among multiple agents as the problem of finding a new program which has consequences common to all programs.

Example

- John and Mary are a couple.
- John wants to buy a new PC. He has two options to save money: stop bar-hopping or giving up a family trip.
- Mary wants to buy new dress. She has also two options to save money: giving up a family trip or going no restaurant. She usually does not go to restaurant. If the family gives up a trip, however, she wants to have a dinner at a restaurant, instead.

Example – cont.

- John's belief is represented as:

  P1: ← *not* pc, pc ← money,

  money ← ¬ bhop, money ← ¬ trip,

  bhop ; ¬ bhop ← , trip ; ¬ trip ←

- P1 has three answer sets:

  S1={ pc, money, ¬ bhop, trip }

  S2={ pc, money, bhop, ¬ trip }

  S3={ pc, money, ¬ bhop, ¬ trip }.

Example – cont.

- Mary's belief is represented as:

  P2: ← *not* dress, dress ← money,

  money ← ¬ rest, money ← ¬ trip,

  ¬ rest ← *not* rest, rest ← ¬ trip,

  trip ; ¬ trip ←

- P2 has two answer sets:

  T1={ dress, money, ¬ rest, trip }

  T2={ dress, money, rest, ¬ trip }

Example – cont.

Question: Which conclusions should be drawn as consensus of the couple?

- {money} is included in every answer set of two programs, so it is certainly a result of consensus.
- {money, trip} is a subset of both S1 and T1, and {money, ¬trip} is a subset of S2,S3,and T2. So these two sets are admissible as results of consensus.

Observation

- There are two different types of consensus.
- The first one collects minimal sets of beliefs that are included in an answer set of every program.
- The second one collects maximal sets of beliefs that are included in an answer set of every program.
- The purpose of this study is to develop a theory of such consensus among multiple logic programs.

Extended Disjunctive Program (EDP)

- A program P 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.
- The declarative semantics of EDP is given by the answer set semantics.
- The set of all answer sets of P is written as AS(P).
- AS(P) is an anti-chain set, i.e., no element $S \in AS(P)$ is a proper subset of another element $T \in AS(P)$.

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 P^S. Here, P^S contains the rule $L_1 ; \ldots ; L_l \leftarrow L_{l+1}, \ldots, L_m$ if $\{ L_{m+1}, \ldots, L_n \} \cap S = \Phi$ for any ground rule $L_1 ; \ldots ; L_l \leftarrow L_{l+1}, \ldots, L_m , not\ L_{m+1}, \ldots, not\ L_n$ in P.

Definitions

- A program P is consistent if it has a consistent answer set. An inconsistent program has either the answer set Lit or no answer set.
- A literal L is a consequence of skeptical reasoning in P if L is included in every answer set of P (written $L \in skp(P)$).
- A literal L is a consequence of credulous reasoning in P if L is included in some answer set of P (written $L \in crd(P)$).

Consensus Logic Programs

- Let P_1 and P_2 be two programs. Then, define $cons(P_1, P_2) = \{ S \cap T \mid S \in AS(P_1)$ and $T \in AS(P_2)\}$.
- In particular, $cons(P_1, P_2) = \phi$ if $AS(P_1) = \phi$ or $AS(P_2) = \phi$.

Consensus Logic Programs

- Let P_1 and P_2 be two programs. A program Q is called minimal consensus (between P_1 and P_2) if

  $AS(Q) = min(cons(P_1, P_2))$.
- A program R is called maximal consensus if

  $AS(R) = max(cons(P_1, P_2))$.
- Q and R are also called consensus programs.
- Each element in AS(Q) / AS(R) is called a result of minimal/maximal consensus.

Example

For $AS(P_1) = \{ \{ p, s \} , \{ q \} \}$ and
$AS(P_2) = \{ \{ p, t \}, \{ r \} \}$,

it becomes $cons(P_1, P_2) = \{ \phi , \{p\} \}$.

Then, the result of minimal consensus is ϕ,
while the result of maximal consensus is $\{p\}$.

Properties

- Let P_1 and P_2 be two programs, Q a minimal consensus, and R a maximal consensus.

  1. Q and R are consistent iff both P_1 and P_2 are consistent, or one is consistent and the other has the contradictory answer set Lit.
  2. If $AS(P_1)=\{Lit\}$, $AS(Q)=AS(R)=AS(P_2)$.

†When one of two programs is inconsistent,
the results of consensus are trivial.
So we consider consistent programs hereafter.

Properties

- $\forall U \in AS(Q)$, $\exists S \in AS(P_1)$ and $\exists T \in AS(P_2)$ such that $U \subseteq S$ and $U \subseteq T$.
  $\forall V \in AS(R)$, $\exists S \in AS(P_1)$ and $\exists T \in AS(P_2)$ such that $V \subseteq S$ and $V \subseteq T$.

† A result of minimal/maximal consensus reflects a part of beliefs included in an answer set of every program.

- $\forall S \in AS(P_1)$ and $\forall T \in AS(P_2)$, $\exists U \in AS(Q)$ such that $U \subseteq S$ and $U \subseteq T$.

† Conversely, beliefs in an answer set of every program are partly reflected as a result of minimal consensus.

Properties

- When $AS(Q)=AS(P_1)$ (or $AS(R)=AS(P_1)$), P_1 dominates P_2 under minimal/maximal consensus.

† When P_1 dominates P_2 under minimal/maximal consensus, we can easily have a consensus program as P_1 .

- If $S \subseteq T$ for any $S \in AS(P_1)$ and for any $T \in AS(P_2)$, P_1 dominates P_2 under minimal consensus.
- If $S \subseteq T$ for any $S \in AS(P_1)$ and for some $T \in AS(P_2)$, P_1 dominates P_2 under maximal consensus.

Properties

- Let P_1 and P_2 be two consistent programs, Q a minimal consensus, and R a maximal consensus.
  1. $skp(Q) = skp(P_1) \cap skp(P_2)$,
  2. $skp(Q) \subseteq skp(R)$,
  3. $crd(R) = crd(P_1) \cap crd(P_2)$,
  4. $crd(Q) \subseteq crd(R)$.

† Minimal consensus extracts skeptical consequences that are common between P_1 and P_2 .

Maximal consensus extracts credulous consequences that are common between P_1 and P_2 .

Computing Consensus Programs

- We assume finite ground programs.
- A program P is transformed to its kernel form *ker*(P) which consists of rules of the form:

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

by iterative applications of partial evaluation, and elimination of tautologies, non-minimal rules, and unfired rules.

Computing Consunsus Programs

P_1 , P_2 : programs, $\Sigma \subseteq 2^{Lit}$: an anti-chain over Lit.

1. Compute kernel forms $ker(P_1)$ and $ker(P_2)$.
2. Let $S \in \Sigma$. For any rule $r \in ker(P_1) \cup ker(P_2)$ satisfying $head(r) \cap S \neq \phi$, construct a rule r* s.t.
   - $head(r^*) = head(r) \cap S$,
   - $body(r^*) = body(r) \cup \{ not\ L \mid L \in head(r) \setminus S \} \cup \{ not\ M \mid M \in T \setminus S$ for any $T \in \Sigma \}$.

   Put $R(S)=\{r^*\}$ as the set of all such rules.
3. For any $S \in \Sigma$, collect R(S) as $\cup_{S \in \Sigma} R(S)$.

Define $P_1 \diamond_\Sigma P_2 = \cup_{S \in \Sigma} R(S)$.

Example (1)

P_1: $p \leftarrow q$, $q \leftarrow$ *not* r, $r \leftarrow$ *not* q

P_2: $p \leftarrow r$, $q ; r \leftarrow$

and $\Sigma = \{ \{p\}, \{q\}, \{r\} \}$.

First, ker(P_1) and ker(P_2) become

ker(P_1): $p \leftarrow$ *not* r, $q \leftarrow$ *not* r, $r \leftarrow$ *not* q

ker(P_2): $p ; q \leftarrow$, $q ; r \leftarrow$

Example (2)

For $\{p\} \in \Sigma$, $p \leftarrow$ *not* r in $\ker(P_1)$ becomes

R(p): $p \leftarrow$ *not* q, *not* r,

where *not* q is added to the body as
$q \in \{q\} \setminus \{p\}$ for $\{q\} \in \Sigma$.

Likewise, $p ; q \leftarrow$ in $\ker(P_2)$ becomes

R(p): $p \leftarrow$ *not* q, *not* r

where q is shifted to the body as *not* q,
and *not* r is added to the body as
$r \in \{r\} \setminus \{p\}$ for $\{r\} \in \Sigma$.

Example (3)

As a result, $P_1 \diamond_\Sigma P_2$ becomes

p ← *not* q, *not* r,

q ← *not* p, *not* r,

r ← *not* p, *not* q,

where $AS(P_1 \diamond_\Sigma P_2) = \{ \{p\}, \{q\}, \{r\} \}$.

Properties

- The operation $\diamond_{\Sigma}$ is commutative and associative.
- Let P_1 and P_2 be two consistent programs, $\Sigma \subseteq cons(P_1, P_2)$ an anti-chain over 2^{Lit}. Then, $AS(P_1 \diamond_{\Sigma} P_2) = \Sigma$.

Main Result

- Let P_1 and P_2 be two consistent programs. Then, a minimal consensus Q is given as the program $P_1 \diamond_{AS(Q)} P_2$; and
  a maximal consensus R is given as the program $P_1 \diamond_{AS(R)} P_2$.

Application to Multi-Agent Consensus

- Agents have their own knowledge bases and build consensus through communication.
- Every agent can share the result of consensus.
- A consensus program serves as a social knowledge base which best reflects belief of individual agents.

Example (1)

Three agents -- John, Mary and Susie , cook dinner together. Each agent has different preference:

1. John wants either meat or fish. If he eats meat, he wants salad. Else if he eats fish, he wants soup. He prefers red wine in case of meat, and white wine in case of fish.
2. Mary is vegetarian, so she eats neither meat nor fish. Instead, she wants to have both salad and soup. She likes wine, but no preference between red and white.
3. Susie likes meat and wants to take either salad or soup. She usually drinks beer, but she will give up beer if other two agents agree with drinking red wine. She do not want white wine.

Example (2)

P_j: meat ← *not* fish, fish ← *not* meat,
salad ← meat, soup ← fish,
red ← meat, white ← fish ;

P_m: salad ← , soup ← , red ; while ←,
← meat, ← fish ;

P_s: meat ← , salad ; soup← , beer ← *not* ¬ beer,
¬ beer ← red, ← white .

$AS(P_j)$ = { {meat, salad, red}, {fish, soup, white} };
$AS(P_m)$ = { {salad, soup, red}, {salad, soup, white} };
$AS(P_s)$ = { {meat, salad, beer}, {meat, soup, beer} } .

In this case, the result of minimal consensus: $AS(Q)=\{\phi\}$.
The result of maximal consensus: AS(R)={{salad},{soup}}.

Example (3)

- Now every agent agrees with cooking either salad or soup. The maximal consensus program $P_j \diamond_{AS(R)} P_m \diamond_{AS(R)} P_s$ becomes

  salad ← *not* soup, soup ← *not* salad.
- Three agents notice that there is no consensus about drink. However, Susie can change her preference if the other two agents agree with drinking red wine.
- Then, she asks John and Mary to let her know their consensus about drinking.

Example (4)

- In response to this, John and Mary construct a consensus program wrt drinking. The maximal consensus program wrt drinking becomes

  R: red ← *not* soup, *not* white,

  white ← *not* salad, *not* red.

- As a result, John and Mary inform Susie of their consensus program wrt drinking. Susie updates her program as

  $P_s^+ = P_s \cup R$, which has the single answer set

  { meat, salad, ¬beer, red } .

- The result of maximal consensus among P_j , P_m , and P_s^+ then becomes { salad, red }. Three agents now agree with preparing salad and red wine.

Note

- Why consensus programs are needed?

  red ← *not* soup, *not* white,

  white ← *not* salad, *not* red.

- Because, a simple result of consensus { red } or {white} does not bring information on which the consensus is ground.
- If Susie took soup, the agreement (drinking red wine) could not be reached.

Final Remarks

- [Sakama and Inoue, CLIMA-V, 2005] introduces a framework of coordination. Given two programs P_1 and P_2 , they construct programs Q and R satisfying $AS(Q)=AS(P_1) \cup AS(P_2)$ and $AS(R)= AS(P_1) \cap AS(P_2)$.
- [Sakama and Inoue, CLIMA-VI, 2006] introduces a framework of composition. Given two programs P_1 and P_2 , they construct a program Q satisfying $AS(Q)=\min\{ S \cup T \mid S \in AS(P_1) \text{ and } T \in AS(P_2) \}$.
- Coordination, composition, and consensus formalize different types of social behaviors of multiple agents in logic programming.
- [Inoue and Sakama, ICLP, 2006] reveals that those theories have close relation to a theory of generalization in answer set programming.