Multi-Agent Planning and Negotiation in Logic Programming

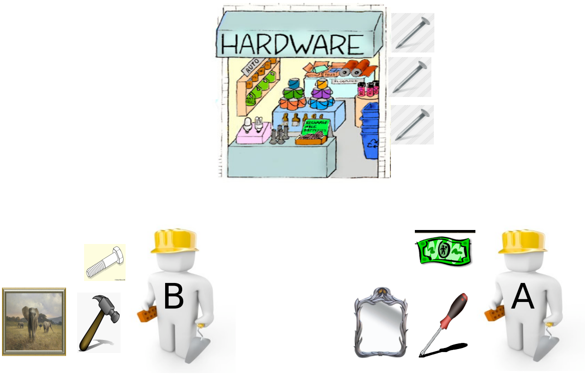
Son Cao Tran[1] Enrico Pontelli[1] Chiaki Sakama[2]

[1]Department of Computer Science
New Mexico State University
[2]Computer and Communication Sciences
Wakayama University

International Conference on Logic Programming 2009

Outline

A Simple Example



Motivations

- Multi-Agent Systems
  - agents with different capabilities
  - agents trying to achieve their own individual goals
  - use Planning to achieve a solution to their individual problems
- The Problem:
  - Individual agents may be unable to separately achieve their goals
    - E.g., missing resources, missing knowledge, . . .
  - Agents can obtain missing resources/knowledge/etc. through mutual exchanges
- Our proposal: Integrate Negotiation in the context of Planning

Objectives

1. Development of a Generic Model of Negotiation
   1. Agents in Dynamic Environments
   2. Negotiations as "actions" to contribute to achieve goals
2. Instantiate Negotiation Model in Multi-Agent Planning
   1. Search for joint plans to achieve all agents' goals
   2. First approach: complete negotiation as single plan steps
   3. Second approach: interleaving steps of planning and step of negotiation
3. Modular encoding in logic programming

Answer Set Planning: $\mathcal{A}$

- Language signature $\langle \mathcal{F}_i, \mathcal{A}_i \rangle$

$$a \textbf{ causes } \ell \textbf{ if } \varphi \qquad \textit{(Dynamic Causal Law)}$$
$$a \textbf{ executable } \varphi \qquad \textit{(Executability Law)}$$

- Semantics:
  - State: complete and consistent set of fluent literals
  - Effects:
$$e_i(a,s) = \{\ell \mid (a \textbf{ causes } \ell \textbf{ if } \varphi), s \models \varphi\}$$
  - Transition:
$$\Phi_i(a,s) = (s \cup e(a,s)) \setminus \overline{e(a,s)}$$
  - Projection:
$$\hat{\Phi}_i([a_1;\ldots;a_n], s) = \Phi(a_n, \hat{\Phi}([a_1;\ldots;a_{n-1}], s))$$
  - Planning Problem: $\langle D_i, I_i, O_i \rangle$

$\mathcal{A}$ and Logic Programming

- From problem $\mathcal{P}_i = \langle D_i, I_i, O_i \rangle$ to $\Pi^n(\mathcal{P}_i)$
- Predicates: $h(i, \ell, t)$ $occ(i, a, t)$ $poss(i, a, t)$

$$
\begin{array}{lcr}
a \textbf{ executable } \varphi & \Rightarrow & poss(i,a,T) \leftarrow h(i,\varphi,T) \\
 & & \leftarrow occ(i,a,T), not\ \ poss(i,a,T) \\
a \textbf{ causes } \ell \textbf{ if } \varphi & \Rightarrow & h(i,\ell,T+1) \leftarrow h(i,\varphi,T), occ(i,a,T) \\
\ell \in I_i & \Rightarrow & h(i,\ell,0) \\
\ell \in O_i & \Rightarrow & \leftarrow not\ \ h(i,\ell,n) \\
(Inertia) & \Rightarrow & h(i,\ell,T+1) \leftarrow h(i,\ell,T), not\ \ h(i,\bar{\ell},T+1) \\
(ActionOccurrence) & \Rightarrow & 1\{occ(i,A,T) : action(i,A)\}1 \leftarrow T < n
\end{array}
$$

Multi-Agent Planning

- Collection of named agents $\mathcal{AG} = \{i_1, \ldots, i_k\}$
- Collection of planning problems $\langle D_{i_1}, I_{i_1}, O_{i_1} \rangle, \cdots, \langle D_{i_k}, I_{i_k}, O_{i_k} \rangle$
  - Tagged fluents/formulae $f[i]$ for $f \in \mathcal{F}_i$
- Collection of tagged formulae $\mathcal{CON}$ (Constraints)
- Collection $\mathcal{NC}$ of sets of pairs (i, a_i) (Non-concurrent actions)
- Collection $\mathcal{C}$ of sets of pairs (i, a_i) (Concurrent actions)
- Multi-state: $\langle s^i \rangle_{i \in \mathcal{AG}}$
- Joint Action Sequence: $\langle \alpha_j \rangle_{j \in \mathcal{AG}}$

$$\left[\begin{array}{lllllll} i_1 \in \mathcal{AG} & \langle & a_0^{i_1} & a_1^{i_1} & \cdots & a_n^{i_1} & \rangle \\ & & & \cdots & & & \\ i_k \in \mathcal{AG} & \langle & a_0^{i_k} & a_1^{i_k} & \cdots & a_n^{i_k} & \rangle \\ & & \langle s_0^i \rangle_{i \in \mathcal{AG}} & \langle s_1^i \rangle_{i \in \mathcal{AG}} & \cdots & \langle s_n^i \rangle_{i \in \mathcal{AG}} & \end{array} \right]$$

Multi-Agent Planning

- Encoding in Logic Programming: all rules from each $\Pi^n(\langle D_i, I_i, O_i\rangle)$ plus

$$
\begin{array}{lcr}
\varphi \in \mathcal{CON} & \Rightarrow & \leftarrow not \;\; h(tagged, \varphi, T) \\
\{(i_1, a_1), \ldots, (i_r, a_r)\} \in \mathcal{NC} & \Rightarrow & \leftarrow occ(i_1, a_1, T), \ldots, occ(i_r, a_r, T) \\
\{(i_1, a_1), \ldots, (i_r, a_r)\} \in \mathcal{C} & \Rightarrow & \leftarrow 1\{occ(i_1, a_1, T), \ldots, occ(i_r, a_r, T)\}r - 1
\end{array}
$$

A Model of Negotiation

- Negotiation as exchanges between two agents – exchange = formulae
- Successful negotiation will affect the state of the two agents
- Agent i has
  1. its own representation language $\mathcal{L}_i$
  2. a collection $\mathcal{W}_i$ of legal states it could be in
- Compatible States: for each pair of agents i, j: $\mathcal{R}_{i,j} \subseteq \mathcal{W}_i \times \mathcal{W}_j$
  - if B has the screw, then A cannot have the screw
- Language Mapping: $\rho_{i,j} : \mathcal{L}_i \rightarrow \mathcal{L}_j$ which preserves equivalences

A Model of Negotiation

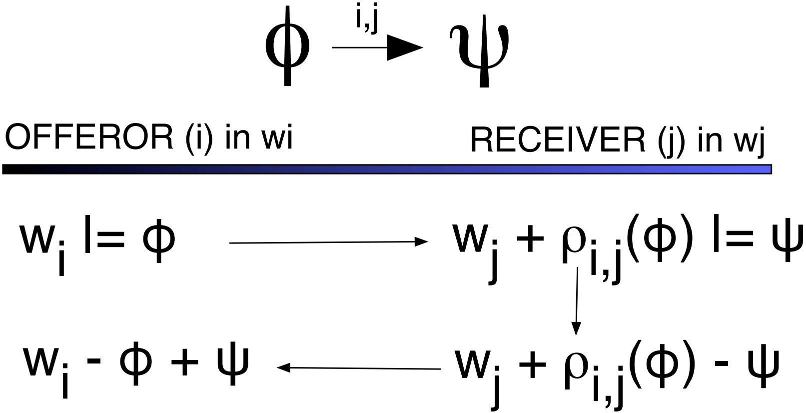
- Proposal: $\varphi \overset{i,j}{\Rightarrow} \psi$
  "If I make φ true for you, would you make ψ true for me?"
- Negotiation is a sequence of proposals; how are proposals evaluated/assimilated?
  - $RPre_j$ conditions to ensure feasible acceptance of a proposal by j

$$RPre_j(w, \varphi \overset{i,j}{\Rightarrow} \psi) \subseteq \mathcal{W}_j$$
$$\textit{if} \;\; w' \in RPre_j(w, \varphi \overset{i,j}{\Rightarrow} \psi) \;\; \textit{then} \;\; w' \models \rho_{i,j}(\psi)$$

  - $RPost_j$ consequences of accepting a proposal; $RPost_j(w, \varphi i, j \Rightarrow \psi) \subseteq \mathcal{W}_j$
  - $OPost_i$ states reached by i if its proposal is accepted

$$OPost_i(w, \varphi \overset{i,j}{\Rightarrow} \psi) \subseteq \mathcal{W}_i$$
$$\textit{if} \;\; w' \in OPost_i(w, \varphi \overset{i,j}{\Rightarrow} \psi) \;\; \textit{then} \;\; w' \models \psi$$

A Model of Negotiation



A Model of Negotiation

- Acceptable Negotiation: i is in state w_i and j in state w_j; offer $\varphi \stackrel{i,j}{\Rightarrow} \psi$
  - Offeror side: $w_i \models \varphi$ and $OPost_i(w_i, \varphi \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset$
  - Receiver side: $RPre_j(w_j, \varphi \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset$ and $RPost_j(w_j, \varphi \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset$

A Model of Negotiation

- Receiver refines the offer [R-negotiable]: from $\varphi \stackrel{i,j}{\Rightarrow} \psi$ to $\rho_{j,i}(\eta) \stackrel{i,j}{\Rightarrow} \psi$

  $RPre_j(w_j, \rho_{j,i}(\eta) \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset \qquad RPost_j(w_j, \rho_{j,i}(\eta) \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset$
- Offeror refines the offer [O-negotiable]: from $\varphi \stackrel{i,j}{\Rightarrow} \psi$ to $\eta \stackrel{i,j}{\Rightarrow} \psi$

  $w_i \models \eta \qquad OPost_i(w_i, \eta \stackrel{i,j}{\Rightarrow} \psi) \neq \emptyset$
- (i, j)-negotiation for ψ: sequence of formulae $m_0, m_1, m_2, \ldots$
  - for each even k, $m_k \stackrel{i,j}{\Rightarrow} \psi$ is O-negotiable
  - for each odd k, $m_k \stackrel{i,j}{\Rightarrow} \psi$ is R-negotiable
  - if last m_n is accept, then $m_{n-1} \stackrel{i,j}{\Rightarrow} \psi$ is acceptable
- Result of a (i, j)-negotiation of ψ are two new states w_i' and w_j', where
  - $(w_i', w_j') \in \mathcal{R}_{i,j}$
  - $w_i' \in OPost_i(w_i, m_{n-1} \stackrel{i,j}{\Rightarrow} \psi)$
  - $w_j' \in RPost_j(w_j, m_{n-1} \stackrel{i,j}{\Rightarrow} \psi)$

Integrating Negotiation in Multi-Agent Planning

- $\mathcal{W}_i$ corresponds to the possible states of agent i
- $\varphi \stackrel{i,j}{\Rightarrow} \psi$ leads
  - agent i moves to a state s_i' such that $s_i' \models \psi$ and $s_i' \models \overline{\varphi}$
  - agent j moves to a state s_j' such that $s_j' \models \varphi$ and $s_j' \models \overline{\psi}$

OFFEROR i

$s_i \models \varphi$

$$OPost_i(s_i, \varphi \stackrel{i,j}{\Rightarrow} \psi) = \begin{cases} \{s_i \cup e \setminus \overline{e}\} & \textit{if } s_i \models \varphi \\ \emptyset & \textit{otherwise} \end{cases} \qquad e = \psi \cup \overline{\varphi}$$

RECEIVER j

$$RPre_j(s_j, \varphi \stackrel{i,j}{\Rightarrow} \psi) = \begin{cases} \{s_j\} & \textit{if } s_j \models \rho_{i,j}(\psi) \\ \emptyset & \textit{otherwise} \end{cases}$$

$$RPost_j(s_j, \varphi \stackrel{i,j}{\Rightarrow} \psi) = \begin{cases} \{s_j \cup e' \setminus \overline{e'}\} & \textit{if } s_j \models \rho_{i,j}(\psi) \\ \emptyset & \textit{otherwise} \end{cases} \qquad e' = \rho_{i,j}(\varphi) \cup \overline{\rho_{i,j}(\psi)}$$

Planning with Non-Interleaved Negotiation

- Complete negotations are viewed as actions
- $N_{i,j}$: all finite length negotiations of the type $\_\_\_ \stackrel{i,j}{\Rightarrow} \_\_\_$
- Joint-action sequence: $\langle a^i_0, a^i_1, \cdots, a^i_k \rangle_{i \in \mathcal{AG}}$ where
  - a^i_j legal action for agent i or
  - $a^i_j \in N_{i,t} \cup N_{t,i}$ for some $t \in \mathcal{AG}$, and $a^t_j = a^i_j$
- If $N \in N_{i,j}$ with outcome $\varphi \stackrel{i,j}{\Rightarrow} \psi$ then

$$\begin{aligned} \Phi_i(N, s_i) &= OPost_i(s_i, \varphi \stackrel{i,j}{\Rightarrow} \psi) \\ \Phi_j(N, s_j) &= RPost_j(s_j, \varphi \stackrel{i,j}{\Rightarrow} \psi) \end{aligned}$$

Planning with Interleaved Negotiation

- More interesting

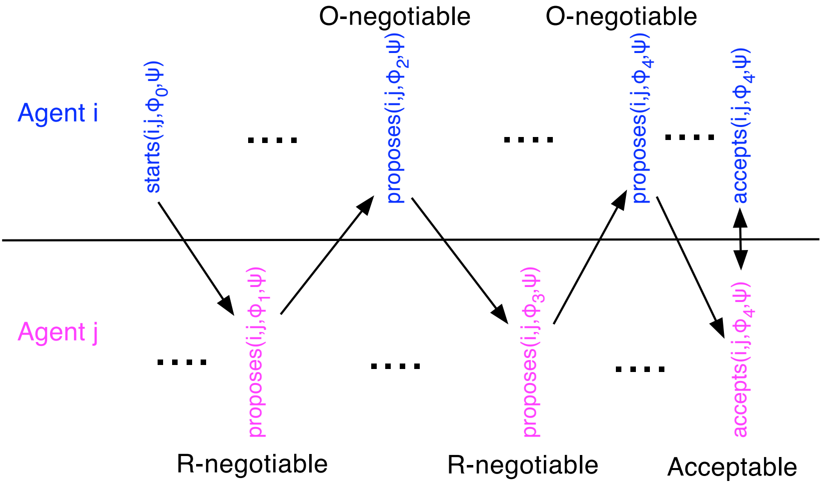
  For agent B to obtain a nail in exchange for a screw from A, it requires A to first buy the nail.

- Enable steps of negotiation to be interleaved with execution of regular actions (Negotiation Actions)
  For simplicity, each agent can participate in only one negotiation at a time.
  - $starts(i, j, \varphi, \psi)$ — generate proposal $\varphi \overset{i,j}{\Rightarrow} \psi$
  - $proposes(i, j, \varphi, \psi)$ — φ is a new negotiation step
  - $accepts(i, j, \varphi, \psi)$ — acceptable negotiation with outcome $\varphi \overset{i,j}{\Rightarrow} \psi$

Planning with Interleaved Negotiation

$$
\begin{aligned}
\Phi_i(starts(i,j,\varphi,\psi),s_i) &= s_i \\
\Phi_j(starts(i,j,\varphi,\psi),s_j) &= fails \\
\Phi_i(proposes(i,j,\varphi,\psi),s_i) &= s_i \\
\Phi_j(proposes(i,j,\varphi,\psi),s_j) &= s_j \\
\Phi_i(accepts(i,j,\varphi,\psi),s_i) &= OPost_i(s_i,\varphi \stackrel{i,j}{\Rightarrow} \psi) \\
\Phi_j(accepts(i,j,\varphi,\psi),s_j) &= RPost_i(s_j,\varphi \stackrel{i,j}{\Rightarrow} \psi)
\end{aligned}
$$

Planning with Interleaved Negotiation



A Logic Programming Encoding

Briefly....

- na(i, a): a is a negotiation action for agent i
- $1\{occ(i, A, T) : action(i, A), occ(i, A, T) : na(i, a)\}1 \leftarrow T < n, agent(i)$
- wait(i, j, φ, ψ): fluent denoting whose turn is next in the negotiation
- hyp_h$(i, \varphi, \psi, \ell, T)$: same as $h(i, \ell, T)$ but assuming φ is lost and ψ gained

$$bad(i, \varphi, \psi, T) \leftarrow hyp\_h(i, \varphi, \psi, p), hyp\_h(i, \varphi, \psi, neg(p))$$

- Acceptable proposal (i case):

$$\begin{array}{lcl} acceptable(i, j, T) & \leftarrow & h(i, wait(i, j, \varphi, \psi), T), h(i, \varphi, T), \\ & & not\ \ bad(i, \varphi, \psi, T) \end{array}$$

A Logic Programming Encoding

- Valid proposal (*i* case):

$$valid\_proposal(i, j, \varphi', T) \leftarrow h(i, wait(i, j, \varphi, \psi), T), h(i, \varphi', T), \text{not } bad(i, \varphi', \psi, T)$$

- Constraints to enforce protocol, e.g.,

$$
\begin{array}{l}
\%\%\ Cannot\ execute\ out\ of\ turn \\
\quad \leftarrow \quad occ(i, proposes(i, j, \varphi, \psi), T), h(i, neg(wait(i, j, \varphi', \psi')), T) \\
\qquad \qquad \ldots \\
\%\%\ Negotiation\ Actions\ should\ be\ correct \\
\quad \leftarrow \quad occ(i, starts(i, j, \varphi, \psi), T), \text{not } h(i, \varphi, T) \\
\quad \leftarrow \quad occ(i, accepts(i, j, \varphi, \psi), T), \text{not } acceptable(i, j, T) \\
\qquad \qquad \ldots
\end{array}
$$

- Final effect of negotiation (*i* case)

$$
\begin{array}{lcl}
h(i, \ell, T+1) & \leftarrow & occ(i, accepts(i, j, \varphi, \psi), T), in\_formula(\ell, \psi) \\
h(i, \bar{\ell}, T+1) & \leftarrow & occ(i, accepts(i, j, \varphi, \psi), T), in\_formula(\ell, \varphi)
\end{array}
$$

Conclusion

- Asymmetric model of negotiation
- Integration of negotiation in a multi-agent planning framework
- Logic Programming validation and feasibility analysis

Future Work

- More complex language matching functions in the context of planning
- Alternative definitions of RPre, RPost, OPost for non-consumable resources
- Agent use offered resources before deciding to accept/reject a proposal
- Negotiation among groups of agents

Thank You

Questions?