A Knowledge-Level Approach to Dialogue Planning

Ron Petrick
(Joint work with Mark Steedman)

Institute for Communicating and Collaborative Systems
School of Informatics
University of Edinburgh

Presented as part of the ICCS Seminar Series, University of Edinburgh
12 October 2007
Motivation

• A simple scenario: contacting a person whose telephone number is not known
  1. Read the number from a piece of paper (Action Planning)
  2. Ask another agent for the number (Dialogue Management)

• In general, an agent operating in a dynamic world must do so with incomplete information about its environment, by
  – Making decisions based on what it knows or believes
  – Reasoning about the effects of its actions
  – Sensing the world to gather information

⇒ Planning with incomplete information and sensing
  – “Reading” and “asking/telling” are both instances of sensing or knowledge-producing actions
  – Action planning and dialogue management problems similar

• Planning dialogue actions is an instance of the general problem of planning with incomplete information and sensing
Planning and dialogue planning

• **Planning problem**
  – A model of the world and/or the agent’s knowledge
  – A description of the initial world/knowledge state
  – A set of state transforming actions
  – A set of goal conditions to be achieved

• **Plan**
  – A sequence of actions that when applied to the initial state transforms the state in such a way that the resulting state satisfies the goal conditions

• **Planning dialogue actions**
  – Model includes references to multiple agents (participants)
  – Actions correspond to dialogue acts, e.g., *ask*, *tell*, *say*, . . .
  – Plans specify mixed-initiative discourse among participants
Previous approaches

• Treating dialogue management as planning is not a new idea
  – Early BDI-based approaches, e.g., Litman & Allen (1987),
    Cohen & Levesque (1990), Grosz & Sidner (1990), . . .

• Recent work has tended to separate action planning and
  dialogue planning and focused on specialized approaches
  – Finite state-transition machines
  – Information state
  – Rule-based approaches to speech act theories, dialogue
    games, textual coherence, etc.

• Planning is often rejected on complexity grounds
This approach

- Adapt recent techniques from the knowledge representation and planning communities to mixed-initiative dialogue
- Extend the **Linear Dynamic Event Calculus** (LDEC) for modelling dialogue domains with knowledge and sensing (Steedman 1997, 2002)
- Apply intuitions from the **PKS planner** (Planning with Knowledge and Sensing) (Petrick & Bacchus 2002, 2004)
- **Goal**: PKS as a target platform for generating dialogue plans
Outline of talk

1. Motivation and overview
2. Linear Dynamic Event Calculus (LDEC)
3. PKS: Planning with Knowledge and Sensing
4. Extending LDEC for planning dialogue actions
5. Example
6. Next step: automated dialogue planning with PKS
7. Conclusions and future work
• A logical language for reasoning about actions and change (Steedman 1997, 2002)

• Related to
  – Situation Calculus (McCarthy & Hayes 1969)
  – Event Calculus (Kowalski & Sergot 1986)
  – STRIPS (Fikes & Nilsson 1971)
  – Dynamic logic (Harel 1984)
  – Linear logic (Girard 1987)
  – ...

• The dynamics of the world are modelled axiomatically
Linear Dynamic Event Calculus (LDEC)...

- World properties are modelled by logical fluents:
  
  \[ \text{temperature, doorOpen, objLoc(x), \ldots} \]

- Actions provide the sole means of change in the world and are defined by a deterministic necessity modality \([\alpha]\)

  \[ P(a) \Rightarrow [\alpha]Q(a, b) \]

- A sequence operator “;” chains actions together

  \[ [\alpha_1; \alpha_2; \ldots; \alpha_n] F_1 \land \neg F_2 \]

- A special fluent \textit{affords}(\alpha) denotes an action \(\alpha\) as being “possible” or “executable” in a situation
LDEC domains

- **LDEC domains** are formally described by 3 sets of axioms

1. Action precondition axioms

   \[ F_1 \land F_2 \land \ldots \land F_k \Rightarrow \text{affords}(\alpha) \]

2. Effect axioms

   \[ \{\text{affords}(\alpha)\} \land F_1 \land F_2 \land \ldots \land F_m \longrightarrow [\alpha] F_1' \land F_2' \land \ldots \land F_n' \]

3. Initial situation axioms

   \[ F_1 \land F_2 \land \ldots \land F_l \]
Action precondition axioms

\[ F_1 \land F_2 \land \ldots \land F_k \Rightarrow \text{affords}(\alpha) \]

- Action precondition axioms specify the fluent conditions \((F_1, F_2, \ldots, F_k)\) that must hold for an action \(\alpha\) to be performed

Example

\[ \text{handempty} \land \text{ontable}(x) \Rightarrow \text{affords}(\text{pickup}(x)) \]
Effect axioms

\[
\{\text{affords}(\alpha)\} \land F_1 \land F_2 \land \ldots \land F_m \rightarrow [\alpha] F'_1 \land F'_2 \land \ldots \land F'_n
\]

- Effect axioms encode the changes (and non-changes) to fluents that result from performing actions
- Linear implication \( \rightarrow \) provides STRIPS-style fluent updates
  - Performing \( \alpha \) makes the \( F_i \)'s \text{false} and the \( F'_j \)'s \text{true}
  - All other fluents are \text{unchanged} by \( \alpha \)
- A solution to the \textbf{frame problem} (McCarthy & Hayes 1969)

**Example**

\[
\{\text{affords}(\text{pickup}(x))\} \land \text{handempty} \land \text{ontable}(x) \rightarrow [\text{pickup}(x)]\text{holding}(x)
\]
Initial situation axioms

\[ F_1 \land F_2 \land \ldots \land F_l \]

- Initial situation axioms describe the state of world fluents before any actions have been applied

Example

\[ \text{handempty} \land \text{ontable(cup)} \land \text{box(b)} \land \text{empty(b)} \]
A plan in LDEC is an action sequence $\alpha$ afforded by the initial situation that satisfies a goal formula $\phi_{goal}$ in a domain $\Sigma$, i.e.,

$$\Sigma \models \text{affords}(\alpha) \land [\alpha] \phi_{goal}$$

Easy to automate process using forward-chaining search
- Action precondition axioms determine the set of actions that are possible at a given situation
- Effect axioms update the fluent states (used to verify action preconditions and goals) based on action choice
Example: LDEC planning by proof

Axioms

A1. \( \text{handempty} \land \text{ontable}(x) \Rightarrow \text{affords}(\text{pickup}(x)) \)

A2. \( \text{holding}(x) \land \text{box}(y) \Rightarrow \text{affords}(\text{dropInBox}(x, y)) \)

A3. \( \{\text{affords}(\text{pickup}(x))\} \land \text{handempty} \land \text{ontable}(x) \)

\( \quad \rightarrow_0 [\text{pickup}(x)] \text{holding}(x) \)

A4. \( \{\text{affords}(\text{dropInBox}(x, y))\} \land \text{holding}(x) \land \text{empty}(y) \)

\( \quad \rightarrow_0 [\text{dropInBox}(x, y)] \text{inBox}(x, y) \land \text{handempty} \)

A5. \( \text{handempty} \land \text{ontable}(\text{cup}) \land \text{box}(b) \land \text{empty}(b) \)

Goal: \( \text{affords}(\alpha) \land [\alpha] \text{inBox}(\text{cup}, b) \)

Plan:

\( \alpha = [\text{pickup}(\text{cup}); \text{dropInBox}(\text{cup}, b)] \)
• We require modalities for representing multi-agent dialogue

\[
\begin{align*}
[S] & \quad \text{Speaker supposition} \\
[H] & \quad \text{Hearer supposition} \\
[X], [Y], \ldots & \quad \text{Other participant/agent suppositions} \\
[C_{XY}] & \quad \text{Common ground between X and Y}
\end{align*}
\]

Examples

\[
\begin{align*}
[S]p & \quad \text{“The speaker supposes } p\text{.”} \\
[S][H]p & \quad \text{“The speaker supposes the hearer supposes } p\text{.”} \\
[H][C_{SH}][S]p & \quad \text{“The hearer supposes it’s common ground between the speaker and hearer that the speaker supposes } p\text{.”}
\end{align*}
\]
• What about *ask/tell*?  
  – Can be viewed as sensing actions  
  – Original LDEC does not model knowledge and sensing!

• Many formal accounts of knowledge and action, e.g.,  
  – Possible worlds, modal logics of knowledge, . . .

• **Trade-off**: expressiveness of representation vs. tractability/efficiency of reasoning process  
  – Practical planning requires efficient reasoning

• **Our solution**: apply the intuitions of PKS for building plans with incomplete knowledge and sensing actions
Outline of talk

1. Motivation and overview
2. Linear Dynamic Event Calculus (LDEC)
3. PKS: Planning with Knowledge and Sensing
4. Extending LDEC for planning dialogue actions
5. Example
6. Next step: automated dialogue planning with PKS
7. Conclusions and future work
• A “knowledge-level” planner that builds plans based on what an agent knows (Petrick & Bacchus 2002, 2004)
• Planner’s knowledge state is represented by 5 databases, each of which models a different type of knowledge
• Database contents have a fixed, formal translation to formulae in a modal logic of knowledge
• Representation supports non-propositional features like functions and run-time variables
• Can construct plans with conditional branches to manage indefinite information (contingent planning)
• In the middle ground of representation/efficiency trade-off
Knowledge representation in PKS databases

- $K_f$: knowledge of positive and negative facts (no CWA)
  
  \[ p \in K_f : \text{the agent knows } p \]

- $K_w$: knowledge of binary sensing effects
  
  \[ \phi \in K_w : \text{the agent knows } \phi \text{ or knows } \neg \phi \]

- $K_v$: knowledge of function values, multi-valued sensing effects
  
  \[ f \in K_v : \text{the agent knows the value of } f \]

- $K_x$: exclusive-or knowledge
  
  \[ (\ell_1|\ell_2|\ldots|\ell_n) \in K_x : \text{exactly one of the } \ell_i \text{ must be true} \]

- LCW: local closed world information (Etzioni et al. 1994)
Reasoning in PKS

- A **primitive query language** is used to pose questions about the planner’s knowledge state
  - $K_p$, is $p$ known to be true?
  - $K_v t$, is the value of $t$ known?
  - $K_v p$, is $p$ known to be true or known to be false?
  - The negation of the above queries

- A sound, but incomplete, inference procedure checks the database contents to determine the truth of a query
  - More than just single database lookup
  - Efficient reasoning (usually)

- Used to evaluate preconditions, conditional rules, and goals
### Action representation in PKS

<table>
<thead>
<tr>
<th>Action</th>
<th>Preconditions</th>
<th>Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>readPaper</td>
<td>KhavePaper</td>
<td>( \text{add}(K_v, \text{phoneNumber}) )</td>
</tr>
<tr>
<td>dial</td>
<td>K_v, phoneNumber</td>
<td>( \text{add}(K_f, \text{dialledOk}) )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{add}(K_w, \text{connected}) )</td>
</tr>
</tbody>
</table>

- **Preconditions** are represented as conjunctions of primitive queries
- **Effects** update the databases (i.e., the knowledge state), rather than the world state, in a STRIPS-like manner
- Easy to compute new knowledge states by forward chaining
  - Evaluate primitive queries against a set of databases
  - Actions update databases \( \Rightarrow \) update knowledge state
- **Planning** is a search through the space of database states
Planning by forward knowledge state progression

- **Conditional branches** are formed from $K_w$ (and $K_v$) formulae
- Actions can be parametrized with ground $K_v$ terms
- Goal conditions must be satisfied along every plan branch
Outline of talk

1. Motivation and overview
2. Linear Dynamic Event Calculus (LDEC)
3. PKS: Planning with Knowledge and Sensing
4. Extending LDEC for planning dialogue actions
5. Example
6. Next step: automated dialogue planning with PKS
7. Conclusions and future work
Extending LDEC for planning dialogue actions

• **Recall:** conversational acts like *ask/tell* can be viewed as sensing actions that require reasoning about knowledge

• Reasoning (and planning) in PKS is based on manipulating a set of simple knowledge-level assertions
  - “Know $p$”
  - “Know the value of $t$”
  - “Know whether $p$”

• $K_v$ and $K_w$ formulae in PKS act as “placeholders” for indefinite information $\Rightarrow$ information returned by sensing actions

• Can we do something similar in LDEC?
Knowledge fluents in LDEC

- Introduce PKS-style knowledge assertions into LDEC using **knowledge fluents** (Demolombe & Pozos Parra 2000)
  
  \[ Kp \quad \text{“Know } p \text{”} \]
  
  \[ Kv, t \quad \text{“Know the value of } t \text{”} \]
  
  \[ Kw,p \quad \text{“Know whether } p \text{”} \]

- Knowledge fluents are treated as ordinary fluents that have particular meanings with respect to the knowledge state.

**Examples**

\[ [X] Kv, \text{phoneNumber} \]

\[ [X] Kv, \text{phoneNumber} \Rightarrow \text{affords(dial}(X)) \]

\{\text{affords(dial}(X))\} \rightarrow_o [\text{dial}(X)] [X] Kw, \text{connected} \]
General reasoning rules for modalities

• We also require a set of general purpose rules for reasoning about speaker-hearer and common ground modalities

A1. \([X] p \Rightarrow p\) Supposition Veridicality
A2. \([X] \neg p \Rightarrow \neg [X] p\) Supposition Consistency
A3. \(\neg [X] p \Rightarrow [X] \neg [X] p\) Negative Introspection
A4. \([C] p \iff ([S] [C] p \land [H] [C] p)\) Common Ground
A5. \([X] [C] p \Rightarrow [X] p\) Common Ground Veridicality

• In general, we do not require axioms for reasoning within the scope of the knowledge fluents \(K/K_v/K_w\)

• Since knowledge fluents are ordinary fluents, plans can still be constructed using “planning by proof”
Outline of talk

1. Motivation and overview
2. Linear Dynamic Event Calculus (LDEC)
3. PKS: Planning with Knowledge and Sensing
4. Extending LDEC for planning dialogue actions
5. Example: catching a train
6. Next step: automated dialogue planning with PKS
7. Conclusions and future work
Initial facts

F1. “If I know what time it is then I know what train I will catch.”

\[ [S] K_vtime \Rightarrow [S] K_vtrain \]

F2. “I don’t know what train I will catch.”

\[ [S] \neg K_vtrain \]

F3. “I suppose you know what time it is.”

\[ [S] [H] K_vtime \]

F4. “I suppose it’s not common ground I don’t know what time it is.”

\[ [S] \neg [C_{SH}] \neg [S] K_vtime \]
Actions: $ask(X, Y, p)$ and $tell(X, Y, p)$

R1. “If $X$ doesn’t know $p$ and $X$ supposes $Y$ does, $X$ can ask $Y$ about it.”

$$\neg [X]p \land [X][Y]p \Rightarrow \text{affords}(ask(X,Y,p))$$

R2. “If $X$ asks $Y$ about $p$, it makes it common ground $X$ doesn’t know it.”

$$\{\text{affords}(ask(X,Y,p))\} \rightarrow [ask(X,Y,p)] [C_{XY}] \neg [X]p$$

R3. “If $X$ supposes $p$, and $X$ supposes $p$ is not common ground, $X$ can tell $Y$ $p$.”

$$[X]p \land [X]\neg [C_{XY}]p \Rightarrow \text{affords}(tell(X,Y,p))$$

R4. “If $X$ tells $Y$ $p$, $Y$ stops not knowing it and starts to know it.”

$$\{\text{affords}(tell(X,Y,p))\} \land \neg [Y]p \rightarrow [tell(X,Y,p)] [Y]p$$
Planning a direct speech act

Goal: I need to know which train I will catch

<table>
<thead>
<tr>
<th>Step (D1)</th>
<th>[H] $K_v$ time</th>
<th>(F3),(A1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step (D2)</td>
<td>$\neg [S] K_v$ time</td>
<td>(F2),(A2),(F1)</td>
</tr>
<tr>
<td>Step (D3)</td>
<td>$\text{affords}(\text{ask}(S, H, K_v \text{time}))$</td>
<td>(D2),(F3),(R1)</td>
</tr>
<tr>
<td>* ask($S, H, K_v$ time)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step (D4)</td>
<td>$[C_{SH}] \neg [S] K_v$ time</td>
<td>(D3),(R2)</td>
</tr>
<tr>
<td>Step (D5)</td>
<td>$\text{affords}(\text{tell}(H, S, K_v \text{time}))$</td>
<td>(D1),(D4),(A4),(A5),(R3)</td>
</tr>
<tr>
<td>* tell($H, S, K_v$ time)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step (D6)</td>
<td>$[S] K_v$ time</td>
<td>(D5),(D2),(R4)</td>
</tr>
<tr>
<td>Step (D7)</td>
<td>$[S] K_v$ train</td>
<td>(D6),(F1)</td>
</tr>
</tbody>
</table>

- After the speaker asks the hearer for the time the hearer tells the speaker the time.
Planning an indirect speech act

<table>
<thead>
<tr>
<th>Goal: I need to know which train I will catch</th>
</tr>
</thead>
<tbody>
<tr>
<td>(D1) ⇒ [S] ¬ [S] \textit{Kv}time \quad \text{(F2),(A2),(F1),(A3)}</td>
</tr>
<tr>
<td>(D2) ⇒ [S] ¬ [C_{SH}] ¬ [S] \textit{Kv}time \quad \text{(F4)}</td>
</tr>
<tr>
<td>(D3) ⇒ \textit{affords}(\textit{tell}(S, H, ¬ [S] \textit{Kv}time)) \quad \text{(D1),(D2),(R3)}</td>
</tr>
<tr>
<td>\quad \ast \textit{tell}(S, H, ¬ [S] \textit{Kv}time)</td>
</tr>
<tr>
<td>(D4) ⇒ [C_{SH}] ¬ [S] \textit{Kv}time \quad \text{(R2)}</td>
</tr>
<tr>
<td>\quad ⇒ \ldots</td>
</tr>
<tr>
<td>\quad \ast \textit{tell}(H, S, \textit{Kv}time)</td>
</tr>
<tr>
<td>\quad ⇒ \ldots</td>
</tr>
</tbody>
</table>

- After the speaker says “I don’t know what time it is” the hearer tells the speaker the time.
Observations about generated plans

- Plan generation takes place in the space of multi-agent plans
  - No reasoning is done about other participants’ goals
  - Cannot guarantee other participants’ actions
  - Planning is offline
- Actions *ask/tell* are treated as sensing actions
- Conclusions follow from direct rule applications
- Both direct and indirect speech acts result from the machinery for reasoning about knowledge and common ground
- All speech acts are indirect in the sense of involving inference
- Planning by proof can be automated ⇒ PKS as a target platform
Outline of talk

1. Motivation and overview
2. Linear Dynamic Event Calculus (LDEC)
3. PKS: Planning with Knowledge and Sensing
4. Extending LDEC for planning dialogue actions
5. Example: catching a train
6. Next step: automated dialogue planning with PKS
7. Conclusions and future work
Next step: automated dialogue planning with PKS

- There is a simple mapping between standard LDEC axioms and ordinary STRIPS actions
- Extended LDEC axioms can be syntactically compiled into PKS actions
  - LDEC knowledge fluent evaluation $\Rightarrow$ PKS database queries
  - LDEC knowledge fluent update $\Rightarrow$ PKS database updates
- Alternatively: direct encoding of dialogue problems in PKS
- Implement inference rules in PKS to support speaker/hearer modalities while maintaining tractable reasoning
- Manage common ground as an instance of local closed world (LCW) information (Etzioni et al. 1994)
  - Conclude that a formula is false if its truth can’t be established
Conclusions and future work

• Knowledge-level planning techniques are applied to the problem of mixed-initiative dialogue management
• Approach is driven by the knowledge state without specific conversational rules (except common ground consistency)
• Direct and indirect speech acts result from the same process
• Current rule set plus STRIPS-style updates are sufficient (in certain scenarios)
• This work offers a way of generating plans directly from first principles with an aim towards automation in PKS
• Future work: evaluation
  – PKS’s efficiency/effectiveness has been demonstrated on standard planning benchmarks
  – We believe performance on dialogue problems will be similar
• We are looking for interesting problems!


