Multi-agent actions under uncertainty: situation calculus, discrete time, plans and policies

David Poole

University of British Columbia

The problem and Solution

Problem: determine what an agent should do based on background knowledge, preferences and what it observes.

Basis for preferences and uncertainty: Bayesian decision theory. Alternatives: goals, satisficing.

Problem representation: independent choice logic.
Alternatives: Bayesian networks, MDPs, FOPC, …

Action representation: situation calculus.
Alternatives: discrete or continuous time.

Agent function represented as conditional plan.
Alternative: policies.

Logic and Uncertainty

Choice:

- Rich logic including all of first-order predicate logic (e.g., Bacchus) — use both probability and disjunction to represent uncertainty.
- Weaker logic where all uncertainty is handled by Bayesian decision theory. The underlying logic contains no uncertainty — uncertainty is in terms of probabilities, decisions and utilities.

Independent Choice Logic

independent choices + acyclic logic program to give consequence of choices.

extension of pure Prolog with negation as failure; rules have their normal logical meaning.

all numbers can be consistently interpreted as probabilities.

extension of Bayesian networks: same notion of 'causation'; can express structured probability tables, logical variables.

independent hypotheses: if there is a dependence we invent a new hypothesis to explain the dependence.

Independent choice logic

An **independent choice logic theory** is built from:

C_0 ‘**nature’s choice space**’ is a set of alternatives.

An **alternative** is a set of atomic choices.

An **atomic choice** is a ground atomic formula.

F the **facts** is an acyclic logic program such that no atomic choice unifies with the head of any rule. Can include negation as failure.

Semantics

A **total choice** is a set containing exactly one element of each alternative in $\mathbf{C}_0$.

For each total choice τ there is a possible world w_τ.

Formula f is true in w_τ (written $w_\tau \models f$) if f is true in the (unique) stable model of $\mathbf{F} \cup \tau$.

Independent choice logic

An **independent choice logic theory** can also contain:

A called the **action space**, is a set of primitive actions that the agent can perform.

O the **observables** is a set of terms.

P_0 is a function $\cup \mathbf{C}_0 \rightarrow [0,1]$.
Probability distribution over alternatives:
$\forall \chi \in \mathbf{C}_0, \sum_{\alpha \in \chi} P_0(\alpha) = 1.$

Temporal models in ICL

ICL is independent of any model of time. E.g.,

- Time implicit: action chosen depends on history:

  $do(A) \leftarrow do\_choice(A, C) \wedge history(C)$

  $\forall C\ \{do\_choice(A, C) : A \text{ possible action}\} \in \mathbf{C}_\alpha$

- Explicit time: discrete Markovian

  $do(A, T) \leftarrow do\_choice(A, S) \wedge state(S, T)$

  $state(S', T + 1) \leftarrow state\_trans(S, S') \wedge state(S, T)$

  $\forall S\ \{do\_choice(A, S) : A \text{ possible action}\} \in \mathbf{C}_\alpha$

  $\forall S\ \{state\_trans(S, S') : S' \text{ state}\} \in \mathbf{C}_0$

- Situation-based time, actions specified in program.

Situation calculus and Uncertainty

s_0 is a situation and $do(A, S)$ is a situation if A is an action and S is a situation.

Deterministic case: the trajectory of actions by the (single) agent determines what is true — situation=state.

With uncertainty, the trajectory of an agent's actions does not determine what is true.

Choice:

- keep the semantic conception of situation=state,
- or keep the syntactic form, so situation≠state, but situations have simple form.

In general there will be a probability distribution over states for a situation.

The agent's actions are treated very differently from exogenous actions.

Situation Calculus in ICL

A possible world is temporally extended — specifies a truth value for every fluent in every situation.

Use standard situation calculus rules to specify what is true after an action — body of rules can include atomic choices.

Robot programs select situations in possible worlds.

Programs can be contingent on observations: the robot will observe different things in different possible worlds.

Actions have no preconditions — they can always be attempted.

Situation Calculus in ICL: Example

$$
\begin{aligned}
& carrying(key, do(pickup(key), S)) \leftarrow \\
& \quad at(robot, Pos, S) \wedge \\
& \quad at(key, Pos, S) \wedge \\
& \quad pickup\_succeeds(S).
\end{aligned}
$$

$$
\begin{aligned}
& carrying(key, do(A, S)) \leftarrow \\
& \quad carrying(key, S) \wedge \\
& \quad A \neq putdown(key) \wedge \\
& \quad A \neq pickup(key) \wedge \\
& \quad keeps\_carrying(key, S).
\end{aligned}
$$

Alternatives

$$\forall S \ \{pickup\_succeeds(S), pickup\_fails(S)\} \in \mathbf{C}_0$$

$P_0(pickup\_succeeds(S))$ is the probability the robot is carrying the key after the $pickup(key)$ action when it was at the same position as the key, and wasn't carrying the key.

$$\forall S \ \{keeps\_carrying(key, S), drops(key, S)\} \in \mathbf{C}_0$$

Utility Axioms

Utility complete if $\forall w_\tau \forall S$, exists unique U such that $w_\tau \models utility(U, S)$

$$
\begin{aligned}
& utility(R + P, S) \leftarrow \\
& \qquad prize(P, S) \wedge \\
& \qquad resources(R, S).
\end{aligned}
$$

$$prize(-1000, S) \leftarrow crashed(S).$$

$$prize(1000, S) \leftarrow in\_lab(S) \wedge \sim crashed(S).$$

$$prize(0, S) \leftarrow \sim in\_lab(S) \wedge \sim crashed(S).$$

$$resources(200, s_0).$$

$$resources(R - Cost, do(goto(To, Route), S)) \leftarrow$$
$$\quad at(robot, From, S) \land$$
$$\quad pathcost(From, To, Route, Cost) \land$$
$$\quad resources(R, S).$$

$$resources(R - 10, do(A, S)) \leftarrow$$
$$\quad \sim gotoaction(A) \land$$
$$\quad resources(R, S).$$

$$gotoaction(goto(A, S)).$$

Imperfect Sensors

A sensor is symptomatic of what is true in the world.

$$
\begin{aligned}
& sense(at\_key, S) \leftarrow \\
& \quad at(robot, P, S) \wedge \\
& \quad at(key, P, S) \wedge \\
& \quad sensor\_true\_pos(S). \\
& sense(at\_key, S) \leftarrow \\
& \quad at(robot, P_1, S) \wedge \\
& \quad at(key, P_2, S) \wedge \\
& \quad P_1 \neq P_2 \wedge \\
& \quad sensor\_false\_pos(S).
\end{aligned}
$$

Conditional Plans

A **conditional plan** can use sequential composition and conditionals.

Plans select situations in worlds. The plan:

$$a;\ \text{if } c \text{ then } b \text{ else } d;\ e \text{ endIf};\ g$$

selects situation $\boldsymbol{do(g, do(b, do(a, s_0)))}$ in w_τ

if $sense(c, do(a, s_0))$ is true in w_τ

selects situation $\boldsymbol{do(g, do(e, do(d, do(a, s_0))))}$ in w_τ

if $sense(c, do(a, s_0))$ is false in w_τ.

Plans select situations in worlds

We can recursively define $do(P, W, S_1, S_2)$ which is true if doing plan P in world W takes us from situation S_1 to S_2.

... in pseudo Prolog:

$$do(skip, W, S, S).$$

$$do(A, W, S, do(A, S)) \leftarrow$$
$$\quad primitive(A).$$

$$do((P; Q), W, S_1, S_3) \leftarrow$$
$$\quad do(P, W, S_1, S_2) \wedge$$
$$\quad do(Q, W, S_2, S_3).$$

$$
\begin{aligned}
&do((\text{if } C \text{ then } P \text{ else } Q \text{ endIf}), W, S_1, S_2) \leftarrow \\
&\qquad W \models sense(C, S_1) \wedge \\
&\qquad do(P, W, S_1, S_2). \\
&do((\text{if } C \text{ then } P \text{ else } Q \text{ endIf}), W, S_1, S_2) \leftarrow \\
&\qquad W \not\models sense(C, S_1) \wedge \\
&\qquad do(Q, W, S_1, S_2).
\end{aligned}
$$

Expected Utility of Plans

The **expected utility** of plan P is

$$\varepsilon(P) = \sum_{\tau} p(w_\tau) \times u(w_\tau, P)$$

where $u(W, P)$ is the utility of plan P in world W:

$$u(W, P) = U \text{ if } W \models utility(U, S)$$

$$\text{where } do(P, W, s_0, S)$$

$p(w_\tau)$ is the probability of world w_τ:

$$p(w_\tau) = \prod_{\chi_0 \in \tau} P_0(\chi_0)$$

Other Work

Exponentially more compact than probabilistic STRIPS: E.g., each predicate p_i persists stochastically and independently through a wait:

$$p_i(do(wait, S)) \leftarrow persists\_p_i(S) \wedge p_i(S) \in \mathbf{F} \text{ for each } p_i$$

$$\{persists\_p_i(S), stops\_p_i\} \in \mathbf{C}_0 \text{ for each } p_i$$

Similar to action networks [Boutilier et al. 95] but doesn't need #*actions* $\times$ #*predicates* space — this the frame problem!

Plans correspond to policy trees of finite stage POMDPs [Kaelbling et al. 96].

Conditional plans are like Levesque[AAAI-96]'s robot plans.

Policies

Can axiomatize change using temporal model
(e.g., event calculus).

Reactive Policy:

$$
\begin{aligned}
& do(pickup(key), T) \leftarrow \\
& \quad sense(at\_key, T) \land \\
& \quad recall(want\_key, T).
\end{aligned}
$$

Conclusion

- Combine situation calculus and Bayesian decision theory.
- Allow conditional plans and noisy sensors and effectors.
- Notion of goal expanded to utilities.
- Plans or policies have expected values.
- Planning: finding (approximately) optimal plan/policy.
- Paper explores muti-agents and reactive policies vs plans.