Research Review

ADL

The Action Description Language (ADL) language is a richer domain specific language used for planning and agent modelling. It relaxed some of the restrictions in the STRIPS language and made it possible to encode more realistic problems. The ADL allows the effects of an operator to be conditional as its basic proposal. The language itself can automatically generate a chain of action to satisfied the restriction and lead to the final result. The difference of ADL and STRIPS is it allows unknown condition of not occurred event which is set to false in STRIPS. This change makes the language more applicable to the real-world problems.

```
Action (
  Load (c: Freight, p: Airplane, A: Airport)
  Precondition: At(c, A) ^ At(p, A)
  Effect: ¬At(c, A) ^ In(c, p)
)

Action (
  Unload (c: Freight, p: Airplane, A: Airport)
  Precondition: In(c, p) ^ At(p, A)
  Effect: At(c, A) ^ ¬In(c, p)
)

Action (
  Fly (p: Airplane, from: Airport, to: Airport)
  Precondition: At(p, from)
  Effect: ¬At(p, from) ^ At(p, to)
)
```

```
(define (problem strips-gripper2)
    (:domain gripper-strips)
    (:objects rooma roomb ball1 ball2 left right)
    (:init (room rooma)
           (room roomb)
           (ball ball1)
           (ball ball2)
           (gripper left)
           (gripper right)
           (at-robby rooma)
           (free left)
           (free right)
           (at ball1 rooma)
           (at ball2 rooma))
    (:goal (at ball1 roomb)))
```

Fig.1
Left: The ADL code
Right: The PDDL code

PDDL

The Problem Domain Description Language (PDDL) was introduced as a computer-parsable, standardized syntax for representing STRIPS, ADL, and other languages. PDDL has been used as the standard language for the planning problems. PDDL itself can express all the question in STRIPS and ADL languages. PDDL itself is within development and has different extensions and successors. All these dialects aim to solve different condition of problem including ontology(OPT), probabilistic effects(PPDDL), abstract of action(APPL) and so on.

Prolog

prolog is a logic programming language based on the first-order logic. It is widely applied in artificial intelligence and computational linguistics problems. Prolog was also used to solve the interleaving problem. One solution to the interleaving problem was goal regression planning, a technique in which steps in a totally ordered plan are reordered so as to avoid conflict between subgoals. This was introduced by Waldinger (1975) and also used by Warren's (1974) WARPLAN and it is one of the best examples of the remarkable economy that can sometimes be gained by using logic programming: WARPLAN is only 100 lines of code, a small fraction of the size of comparable planners of the time.

```
mother_child(trude, sally).

father_child(tom, sally).
father_child(tom, erica).
father_child(mike, tom).

sibling(X, Y)      :- parent_child(Z, X), parent_child(Z, Y).

parent_child(X, Y) :- father_child(X, Y).
parent_child(X, Y) :- mother_child(X, Y).
```

Fig.2 The Prolog code

Reference:

Winner, Elly, and Manuela M. Veloso. "Analyzing Plans with Conditional Effects." *AIPS*. 2002.
https://en.wikipedia.org/wiki/Action_description_language

Pednault, Edwin PD. "ADL: Exploring the Middle Ground Between STRIPS and the Situation Calculus." *Kr* 89 (1989): 324-332.

McDermott, Drew, et al. "PDDL-the planning domain definition language." (1998).
Waldinger, Richard. *Achieving several goals simultaneously*. Menlo Park, CA: Stanford Research Institute, 1975.

Gringarten, Alain C., and Henry J. Ramey Jr. "Unsteady-state pressure distributions created by a well with a single horizontal fracture, partial penetration, or restricted entry." *Society of petroleum engineers journal* 14.04 (1974): 413-426.
Warren, David HD. *WARPLAN: A system for generating plans*. University of Edinburgh. School of Artificial Intelligence. Department of Computational Logic, 1974.

Russell, Stuart Jonathan, et al. *Artificial intelligence: a modern approach*. Vol. 2. No. 9. Upper Saddle River: Prentice hall, 2003.