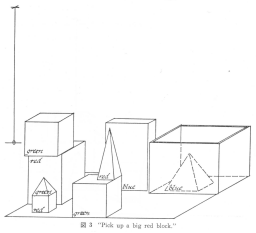
Design of Physically Grounded Communication System
実世界指向コミュニケーション特論

Michita Imai
今井 倫太

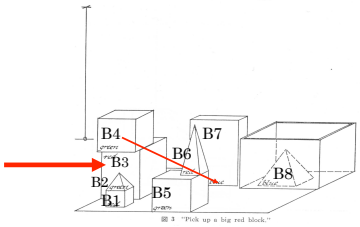
SHRDLU

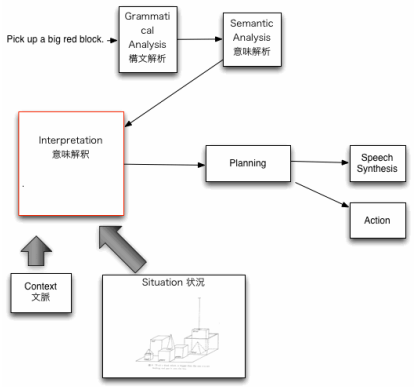
- 1968, T. Winograd developed it.
- Dialogue system on a virtual world (block world)

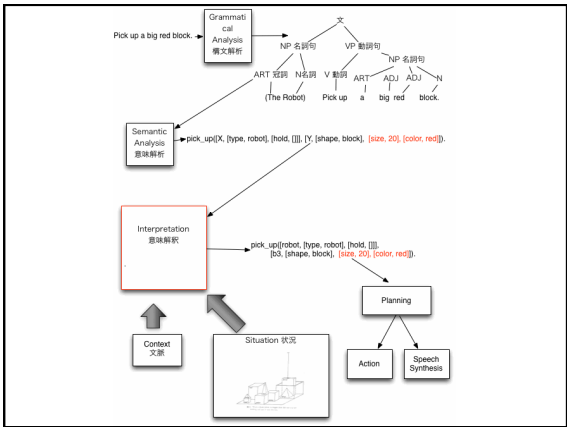


SHRDLU

- pick up a big red block.
  - OK





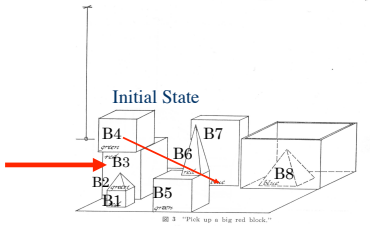


Planning Algorithm

- Generate a sequence of actions to achieve a given goal.
- Action plan from a initial state to a goal state
- A system (ie. a robot) executes the actions after planning.
  - SHRDLU

Plan

- Goal：Pickup B3 (pick up a big red block.)



Plan
Pickup B4

Put B4 Table

Pickup B3

Planning Algorithm

- STRIPS
  - 1971: Developed at Stanford Univ.
  - STRIPS: Stanford Research Institute Problem Solver
  - Simple and basic planning algorithm
  - There is a situation that it is not good at dealing with real world problems.

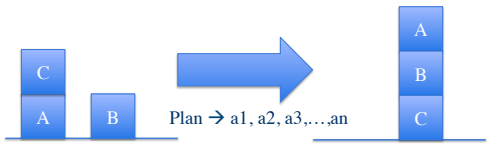
STRIPS

- Set of States
- Initial State
  - Current environmental state
- Goal State
- A set of actions

Initial State and Goal State

Initial State I:
{Clear(B), Clear(C), On(C, A), HandEmpty, OnTable(A), OnTable(B)}

Goal State G：
{On(A,B), On(B,C)}



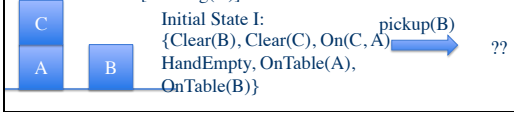
Action Rule

- Definition of an action
  - Condition List：Conditions to execute an action
  - Delete List：a literal/fact which does not exist after executing the action.
  - Add List：a literal/fact which appears after executing the action.
- pickup(X): Pickup a block on a table
  - Condtion List：[OnTable(X), Clear(X), HandEmpty]
  - Delete List：[OnTable(X), Clear(X), HandEmpty]
  - Add List：[Holding(X)]

Action Rule

- Literals/facts which an action does not change are Kept still in DB.
  - An action rule has descriptions which change related to the action.

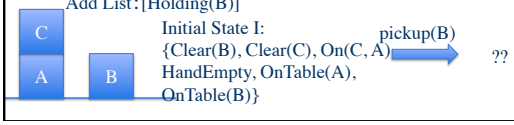
pickup(X): Pickup a block on a table
Condtion List：[OnTable(X), Clear(X), HandEmpty]
Delete List：[OnTable(X), Clear(X), HandEmpty]
Add List：[Holding(X)]



Action Rule

- Literals/facts which an action does not change are Kept still in DB.
  - Matching

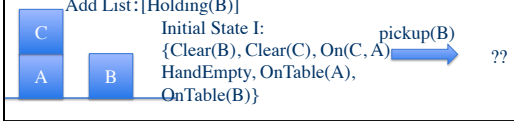
pickup(B): Pickup a block on a table
Condtion List:[OnTable(B), Clear(B), HandEmpty]
Delete List:[OnTable(B), Clear(B), HandEmpty]
Add List:[Holding(B)]



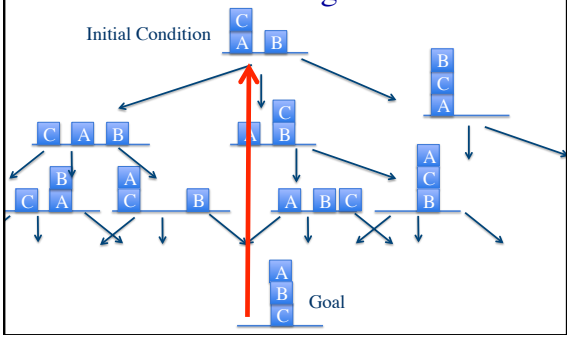
Action Rule

- Literals/facts which an action does not change are Kept still in DB.
  - Execute dele lists -> Execute add lists.

pickup(B): Pickup a block on a table
Condtion List:[OnTable(B), Clear(B), HandEmpty]
Delete List:[OnTable(B), Clear(B), HandEmpty]
Add List:[Holding(B)]



Planing



STRIPS Algorithm

STRIPS(I,G) : I (List of a initial state)、G (List of a goal state)

Step1. Calc the difference D between I and G. D is a set of G's literals which are not true in I.

Step2. If D is empty, STRIPS(I, G) return an empty list.

Step3. Select one of G's iterals, that is a sub-goal, and select an action O to achieve it.

Step4. Call STRIPS(I, P) recursively. Here, P is a conditional list of O.

Step5. Add O into the last part of the list given by STRIPS(I, P). This means that it adds O into an action sequence achieveing P.

Step6. Get a new state Q which is the result of applying an action sequence gotten at Step 5.

Step7.Call STRIPS(Q, G) recursively.

Step8.Add an action sequence given by Step 7 into the last part of the list of Step 5.

Step9.Return the value of Step 8. Finish.

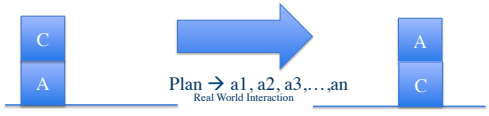
Action Rules

- **pickup(x):** Pickup x on a table
  Conditon:OnTable(x)∧Clear(x)∧HandEmpty
  Delete List: OnTable(x), Clear(x), HandEmpty
  Add List:Holding(x)
- **putdown(x):** Put down X which a robot holds on a table.
  Condition: Holding(x)
  Delete List: Holding(x)
  Add List:OnTable(x), Clear(x), HandEmpty
- **stack(x, y):** Put X which a robot holds on Y
  Condtion:Holding(x) ∧Clear(y)
  Delete List: Holding(x), Clear(y)
  Add List: HandEmpty, On(x,y), Clear(x)
- **unstack(x, y):** Hold x which is on Y.
  Condtion: HandEmpty ∧ On(x,y)∧Clear(x)
  Delete List: HandEmpty, On(x,y), Clear(x)
  Add List: Holding(x), Clear(y)

Planning

- Fill the Gap between Initial Condition and Goal

Diff. On(A,C)

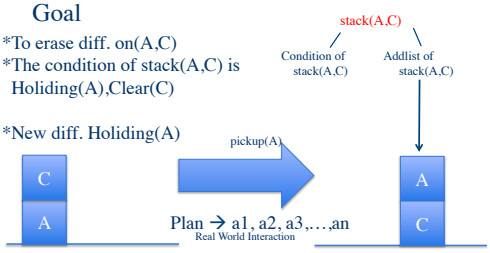


Planning

- Fill the Gap between Initial Condition and Goal

*To erase diff. on(A,C)
*The condition of stack(A,C) is Holding(A),Clear(C)

*New diff. Holding(A)

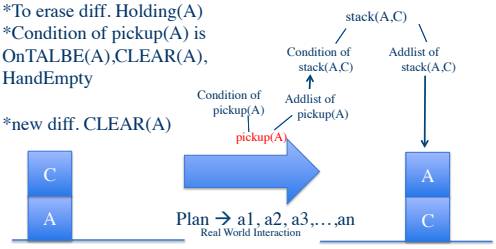


Planning

- Fill the Gap between Initial Condition and Goal

*To erase diff. Holding(A)
*Condition of pickup(A) is OnTALBE(A),CLEAR(A), HandEmpty

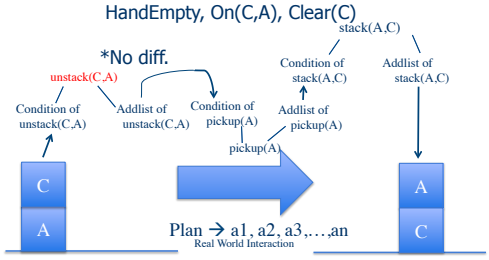
*new diff. CLEAR(A)



Planning

- Fill the Gap between Initial Condition and Goal

*To erase diff. CLEAR(A)
*Condition of unstack(C,A) is HandEmpty, On(C,A), Clear(C)



Planning

- Fill the Gap between Initial Condition and Goal

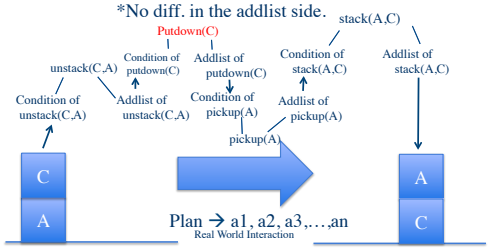
*Apply unstack(C,A) to Initial condition.

*A new diff. appears.
HandEmpy

Planning

- Fill the Gap between Initial Condition and Goal

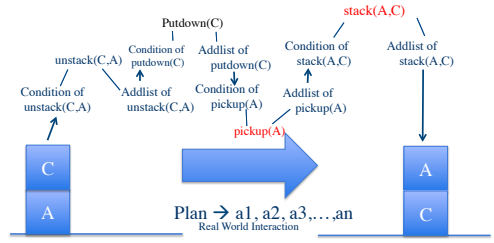
*To erase diff. HandEmpty
*No diff. in the condition side.
*No diff. in the addlist side.



Planning

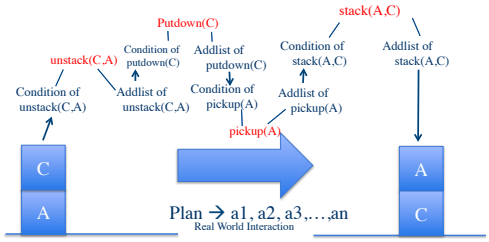
- Fill the Gap between Initial Condition and Goal

*No diff. in the addlist sides of pickup(A) and stack(A,C).



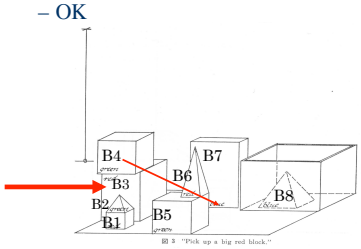
Planning

- Get a plan



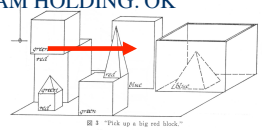
SHRDLU

- pick up a big red block.
  - OK



SHRDLU: Example with Block world knowledge

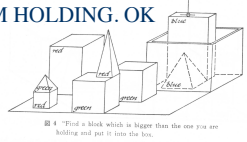
- find a block which is taller than the one you are holding and put it into the box.
  - But "IT," I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING. OK



Real World Interaction

SHRDLU: Example with Current Situation

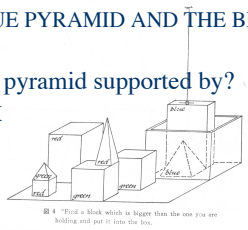
- find a block which is taller than the one you are holding and put it into the box.
  - But "IT," I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING. OK



Real World Interaction

SHRDLU: Example with Context

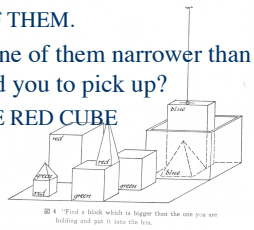
- What does the box contain?
  - THE BLUE PYRAMID AND THE BLUE BLOCK.
- What is the pyramid supported by?
  - THE BOX



Real World Interaction

SHRDLU: Example with Lexical knowledge

- How many blocks are not in the box?
  - FOUR OF THEM.
- Is at least one of them narrower than the one which I told you to pick up?
  - YES, THE RED CUBE



Real World Interaction

Intelligent system in virtual world

- System dealing with toy world
  - Extension of thought experiment
  - Static world
  - Lack of effect of time
  - Sufficient to solve a problem logically
  - Interaction depends on discrete turn-taking.
  - It cannot deal with dynamic changes.

Toward real world application

- There are vast (infinite) numbers of items or events in the real world.
- A part of them relates to the meaning of conversation. All items are not used.
- The items are already selected in the virtual world.
- A system must selects the items related to communication or interaction in real world.