A Methodology for Knowledge Acquisition by Heterogeneous STRIPS-Based Planners

Sujata Ramchandran

Dept. of Electrical & Computer Engineering
University of Arizona

Outline

- Introduction
- Problem Formulation
- Definitions
- Learning Methodology
- Detailed Example
- Related Work
- Conclusion

Introduction

- We can achieve communication in a network of heterogeneous planning systems through translation
- If a problem cannot be solved,
  - it indicates a lack of knowledge within the planner
  - results in the need to send out a query to other planners
- To avoid future queries and solve other similar problems, the planner should acquire knowledge from the solutions that it receives from other planners
- The challenges
  - Many different ways to learn
  - Knowledge representation may be at different levels of abstraction
  - Preserve the correctness of the knowledge base

Components of the System

- *Planner*$_i$ contains
  - Knowledge Base - KB_i with
    - Predicates - $Pred_i$
    - Actions – $Actions_i$
      - $Pre(a_i^k)$, $Add(a_i^k)$, $Del(a_i^k)$, $Con(a_i^k)$
    - Translators – $Trans_i$
  - Domain Truth Table
    - Sets of predicates that can never exist together and those that cannot exist separately in the domain theory of *Planner*$_i$
    - Predicates that are always true or always false with respect to the common ontology
- Common Ontology - the knowledge representation which used by all planners to communicate with each other.

Communication between Planners

- Communication is through
  - Queries containing Initial Conditions, Goals and Constraints
    $Query_i = <I_i, G_i, C_i>$
  - Plans containing Actions and Orderings
    $Plan_i = <A_i, O_i>$
- Communication occurs with the following steps

  In Planner_i
  - Create $Query_i = <I_i, G_i, C_i>$
  - Translate and send as $Query_{co} = <I_{co}, G_{co}, C_{co}>$

  In Planner_j :
  - Receive $Query_{co}$ and translate to $Query_j = <I_j, G_j, C_j>$
  - Create the solution, $Plan_j = <A_j, O_j>$
  - Translate and send $Plan_{co} = <A_{co}, O_{co}>$ as the reply

  In Planner_i
  - Receive the solution and translate to $Plan_i = <A_i, O_i>$

Communication between Planners

- Translation implies that equivalent knowledge is present.
- If this is not true, the solution cannot be represented in the domain theory of $Planner_i$. Hence, the solution cannot be used.
- Objective – To present a methodology that will achieve translation through learning while preserving the correctness of knowledge in the domain theory of the planner.

Formal Objective

- Given $Plan_j = <A_j, O_j>$ in the domain theory of $Planner_j$ to add $Pred_i^{new}$, $Actions_i^{new}$ and $Trans_i^{new}$ to $KB_i = <Pred_i, Actions_i, Trans_i>$, in the domain theory of $Planner_i$, such that
  - $Plan_j = <A_j, O_j>$ can be translated to $Plan_i = <A_i, O_i>$ with the following properties
    - $Plan_i \equiv Plan_j$
    - $Plan_i$ is a correct plan
    - $Plan_i$ is a solution for $Query_i = <I_i, G_i, C_i>$
  - Constraints on this solution
    - $Actions_i^{new}$ is minimal subject to the individual actions being the most similar to $Actions_i$ when compared to any other $Actions_i^{new'}$ that could have been added to KB_i
    - $KB_i' = <Pred_i \cup Pred_i^{new}, Actions_i \cup Actions_i^{new}, Trans_i \cup Trans_i^{new}>$ is consistent and all its elements are correct

Problem Formulation

Why is this Non-Trivial?

- Creating an algorithm for a heuristic process
- Given $Plan_j$ with m actions, maximum number of plans N considered for learning is in the range

$$2^{m-1} < N < B_m$$

where B_m is a Bell number corresponding to the number of ways to partition a set in to non-empty subsets.

$$B_{m+1} = \sum_{k=0}^{m} B_k \binom{m}{k}, \text{ with } B_0 = 1$$

e.g. with m = 8 actions, 128 < N < 4140

Definitions - Equivalence of Knowledge

- The subset of predicates, $Pred_i^p$ from $Planner_i$ is equivalent to the subset of predicates, $Pred_j^q$ in $Planner_j$ if their aggregate semantics are the same. In this case we write $Pred_i^p \equiv Pred_j^q$

  $Pred_i^p$ = $Tower_i$(A,B,C)

  $Pred_j^q$ = On_j(A,B), On_j(B,C), On_j(C, Table), $Clear_j$(A)

- Two plans, $Plan_i = \langle A_i, O_i \rangle$ and $Plan_j = \langle A_j, O_j \rangle$ are equivalent ($Plan_i \equiv Plan_j$) if $Pre(Plan_i) \equiv Pre(Plan_j)$, $Add(Plan_i) \equiv Add(Plan_j)$, $Del(Plan_i) \equiv Del(Plan_j)$ and $Con(Plan_i) \equiv Con(Plan_j)$.

  $Plan_i$

  A_i = 2_Tower_to_Table$_i$(D,E), 3_Tower_to_Tower$_i$(A,B,D)

  O_i = before(2_Tower_to_Table$_i$(D,E), 3_Tower_to_Tower$_i$(A,B,D))

  $Plan_j$

  A_j = $Unstack_j$(D,E), $Putdown_j$(D), $Unstack_j$(A,B), $Stack_j$(A,D)

  O_j = before($Unstack_j$(D,E), $Putdown_j$(D)), before($Putdown_j$(D), $Unstack_j$(A,B)), before($Unstack_j$(A,B), $Stack_j$(A,D))

Methodology

Definitions - Correctness of Knowledge

- An action a_i^k is correct in $Planner_i$ if
  - $Del(a_i^k)$ is a subset of $Pre(a_i^k)$.
  - $\nexists P_i^k \subset Pre(a_i^k)$ or $\nexists P_i^k \subset Add(a_i^k)$ or $\nexists P_i^k \subset Con(a_i^k)$, such that P_i^k is listed as always false in the Domain Truth Table of $\mathrm{Planner}_i$
  - $\nexists P_i^k \subset Del(a_i^k)$, such that P_i^k is listed as always true in the Domain Truth Table of $\mathrm{Planner}_i$

- $Plan_i = <A_i, O_i>$, produced by $Planner_i$ is correct if
  - The orderings between the actions in $Plan_i$ are acyclic.
  - All actions in A_i are correct with respect to the domain theory of $Planner_i$

- $Plan_i = <A_i, O_i>$ is a solution to $Query_i = <I_i, G_i, C_i>$ if
  - $Plan_i$ is correct plan.
  - $Pre(Plan_i)$ is a subset of I_i
  - G_i is a subset of $Add(Plan_i)$
  - C_i is a subset of $Con(Plan_i)$

Definitions – Similarity of Knowledge

- Similarity Index(*SI*) - the ratio of the number of common predicates between actions to the total number of predicates in both actions. Ranges from 1(actions are equal) to 0 (actions have no common predicates)

$$SI(a_i^p, a_i^q) = \frac{|Common\_pred(a_i^p, a_i^q)|}{|Not\_common\_pred(a_i^p, a_i^q)| + |Common\_pred(a_i^p, a_i^q)|}$$

$$SI(a_i^p, a_i^q) = 0, \text{ if } Common\_pred(a_i^p, a_i^q) = \{\ \}$$

where

$p_i^k \in Common\_pred(a_i^p, a_i^q)$ if $(p_i^k \in Pre(a_i^p) \wedge p_i^k \in Pre(a_i^q))$ $\vee (p_i^k \in Add(a_i^p) \wedge p_i^k \in Add(a_i^q)) \vee (p_i^k \in Con(a_i^p) \wedge p_i^k \in Con(a_i^q))$,

otherwise $p_i^k \in Not\_common\_pred(a_i^p, a_i^q)$

- If $SI(a_i^p, a_i^k) > SI(a_i^q, a_i^k)$, then action a_i^p is more similar than action a_i^q when compared to action a_i^k.

Definitions – Similarity of Knowledge

- The Similarity Index of two sets of actions, A_i^p and A_i^q, is defined as

$$SI(A_i^p, A_i^q) = \frac{|Sum\_Similarity\_Index|}{|A_i^p|}$$

where

$Sum\_Similarity\_Index = \sum SI(a_i^r, a_i^s)$, $\forall\ a_i^r \in A_i^p$ and $a_i^s \in A_i^q$ such that,
$SI(a_i^r, a_i^s) > SI(a_i^r, a_i^m)\ \forall\ a_i^m \in A_i^q,\ m \neq s.$

- If $SI(A_i^p, A_i^k) > SI(A_i^q, A_i^k)$, then the set of actions A_i^p is more similar than the set A_i^q when compared to the set of actions A_i^k.

Methodology

Learning Algorithm

1. For $Plan_{co} = <A_{co}, O_{co}>$, generate all possible partitions of actions in A_{co} and corresponding orderings.
   $Partition_{co}^{n} = \{<A_{co}^{n-1}, O_{co}^{n-1}> <A_{co}^{n-2}, O_{co}^{n-2}> \ldots <A_{co}^{n-n}, O_{co}^{n-n}><O_{co}^{n-m}>\}$

2. Eliminate all partitions, $Partition_{co}^{k}$, that will result in plans which are not acyclic.
   $Partition_{co}^{k}$ is eliminated if, $\exists\ before(a_{co}^{p},a_{co}^{q}) \in O_{co}^{k-m}$ and $\exists\ before(a_{co}^{r},a_{co}^{s}) \in O_{co}^{k-m}, p \neq q \neq r \neq s,\ a_{co}^{p}, a_{co}^{s} \in A_{co}^{k-x}$ and $a_{co}^{q},\ a_{co}^{r} \in A_{co}^{k-y}, x \neq y$

3. Eliminate the partitions that will result in plans containing incorrect actions.

   From the remaining partitions, select those which contain the most number of elements $<A_{co}^{k-l}, O_{co}^{nk-l}>$ which can be translated to the domain theory of $Planner_{i}$.

Methodology

Learning Algorithm

4. For every $Partition_{co}^{k}$ selected above, form a set of actions A_i^k such that, $\forall <A_{co}^{k-l}, O_{co}^{k-l}> \in Partition_{co}^{k}$ that cannot be translated by $Planner_i$, $\exists\ a_i^l \in A_i^k,\ a_i^l \equiv <A_{co}^{k-l}, O_{co}^{k-l}>$.
   a. Each action a_i^l in A_i^k is formed as follows.
   b. Partition $Pre(<A_{co}^{k-l}, O_{co}^{k-l}>)$ in to $\{P_{co}^{1}, P_{co}^{2} \ldots. P_{co}^{n}, P_{co}^{m}\}$ such that, $\forall P_{co}^{k}, k=1 \ldots.n, \exists\ t_i^k \in Trans_i$ that represents the mapping

$$P_{co}^{k} \xleftrightarrow{t_i^k} P_i^k$$

   and $\forall p_{co}^{r} \in Pre(<A_{co}^{k-l}, O_{co}^{k-l}>) \wedge p_{co}^{r} \notin P_{co}^{k}, k=1 \ldots.n, p_{co}^{r} \in P_{co}^{m}$
   Similarly, partition $Add(<A_{co}^{k-l}, O_{co}^{k-l}>)$, $Del(<A_{co}^{k-l}, O_{co}^{k-l}>)$ and $Con(<A_{co}^{k-l}, O_{co}^{k-l}>)$.
   c. Form action a_i^l with $Pre(a_i^l) = P_i^1 \cup P_i^2 \cup \ldots. \cup P_i^n \cup P_{co}^{m}$ where P_i^k is the translation of P_{co}^{k}, i.e.

$$P_{co}^{k} \xleftrightarrow{t_i^k} P_i^k$$

   In a similar manner, $Add(a_i^l)$, $Del(a_i^l)$ and $Con(a_i^l)$ are obtained.
   d. Replace each constant in $Pre(a_i^l)$, $Add(a_i^l)$, $Del(a_i^l)$ and $Con(_i^l)$ with a unique variable, to form the final action, a_i^l.

Learning Algorithm

5. For each set A_i^k, calculate the Similarity Index, $SI(A_i^k, Actions_i)$.

6. If $Actions_i^{new} = A_i^p$ such that $SI(A_i^p, Actions_i) > SI(A_i^q, Actions_i)$, $\forall q \neq p$. Then, $Partition_{co}^p$ is the final choice for learning.

7. $\forall a_i^k \in Actions_i^{new}$, if $\exists\, p_{co}^k \in Pre(a_i^k)$ or $\exists\, p_{co}^k \in Add(a_i^k)$ or $\exists\, p_{co}^k \in Con(a_i^k)$, such that

   a. $p_{co}^k \notin KB_i$, then let $p_i^k = p_{co}^k$ and $Pred_i^{new} = Pred_i^{new} \cup p_i^k$
   Formulate a translator t_i^k that represents the mapping
   $p_i^k \xleftrightarrow{t_i^k} p_{co}^k$

   b. If $p_{co}^k = p_i^k \in KB_i$, and $\nexists\, t_i^k \in Trans_i$ such that $p_i^k \xleftrightarrow{t_i^k} p_{co}^k$
   then create the translator, t_i^k.

   Let $Trans_i^{new} = Trans_i^{new} \cup t_i^k$.

Methodology

Learning Algorithm

8. $\forall a_i^k \in Actions_i^{new}$,
   1. If $a_i^k \notin Actions_i$, generate a new name that is the concatenation of the action names in the common ontology. The parameters of a_i^k consist of the list of all the variables that are a part of the action definition of a_i^k. Form the translator t_i^k which represents the mapping

      $$a_i^k \xleftrightarrow{t_i^k} < A_{co}^{p-k}, O_{co}^{p-k} >$$

      If $a_i^k = a_i^l \in Actions_i$, $l \neq k$ and $\nexists\ t_i^l \in Trans_i$ such that

      $$a_i^k \xleftrightarrow{t_i^k} < A_{co}^{p-k}, O_{co}^{p-k} >$$

      then formulate the translator.

   Let $Trans_i^{new} = Trans_i^{new} \cup t_i^l$

9. Update KB_i with the new information

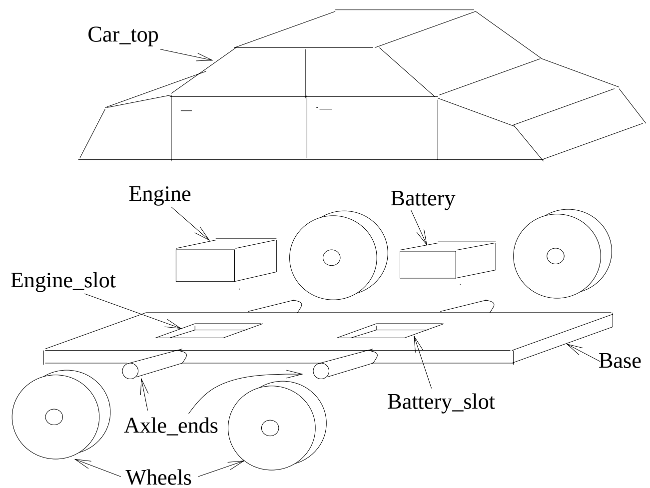
   $Pred_i = Pred_i \cup Pred_i^{new}$

   $Actions_i = Actions_i \cup Actions_i^{new}$

   $Trans_i = Trans_i \cup Trans_i^{new}$

Example

$Planner_3$ – Toy Car Assembly



I_3 = $Clear_3$(Axle_end1), $Clear_3$(Axle_end2), $Clear_3$(Axle_end3), $Clear_3$(Axle_end4), $Clear_3$(Wheel1), $Clear_3$(Wheel2), $Clear_3$(Wheel3), $Clear_3$(Wheel4), $Clear_3$(Eng), $Clear_3$(Batt), $Clear_3$(Eng_slot), $Clear_3$(Batt_slot), $Clear_3$(Top), $Clear_3$(Base)

G_3 = $Car2_3$

C_3 = $Wheel_3$(Wheel1), $Wheel_3$(Wheel2), $Wheel_3$(Wheel3), $Wheel_3$(Wheel4), Car_top_3(Top), Car_$base_3$(Base), Axle_end_3(Axle_end1), Axle_end_3(Axle_end2), Axle_end_3(Axle_end3), Axle_end_3(Axle_end4), $Engine_3$(Eng), Engine_$slot_3$(Eng_slot), $Battery_3$(Batt), Battery_$slot_3$(Batt_Slot)

Example

Planner$_3$ – Action Descriptions

Action	Pre	Add	Del	Con
Fix_together$_3$ (?x,?y)	Clear$_3$(?x) Clear$_3$(?y)	On$_3$(?x,?y) Attached$_3$(?x,?y)	Clear(?y)	
Insert_wheel$_3$(?y,?x)	Clear$_3$(?x), Clear $_3$(?y)	Inserted_wheel$_3$(?y,?x)	Clear$_3$(?y)	Wheel$_3$(?x), Axle_end$_3$(?y)
Base_assembly$_3$ (?bs,?w1,?w2,?w3, ?w4,?ae1,?ae2, ?ae3,?ae4)	Inserted_wheel$_3$ (?ae1,?w1) Inserted_wheel$_3$ (?ae2,?w2) Inserted_wheel$_3$ (?ae3,?w3) Inserted_wheel$_3$ (?ae4,?w4)	Assembled_base_car1$_3$		Wheel$_3$(?w1),Wheel$_3$(?w2) Wheel$_3$(?w3),Wheel$_3$(?w4) Axle_end$_3$(?ae1), Axle_end$_3$(?ae2) Axle_end$_3$(?ae3) Axle_end$_3$(?ae4) Car_base$_3$(?bs)
Body_assembly$_3$ (?tp,?bs)	Fastened$_3$(?tp,?bs) On$_3$(?tp,?bs) Assembled_base_3	Assembled_body$_3$		Car_top$_3$(?tp) Car_base$_3$
Make_car1$_3$	Assembled_base_car1$_3$ Assembled_body$_3$	Car1$_3$		
Make_car2$_3$	Assembled_base_car2$_3$ Assembled_body$_3$	Car2$_3$		

Example

$Planner_3$ – Mappings

Action Mappings in KB_3

$Planner_3$	**Common Ontology**
$Fix\_together_3(?x, ?y)$	$Pickup_{co}(?x)$ > $Align_{co}(?x,?y)$ > $Place_{co}\_\{co\}(?x,?y)$ > $Attach\_on_{co}(?x,?y)$
$Insert\_wheel_3(?x, ?y)$	$Pickup_{co}\_\{co\}(?x)$ > $Align_{co}(?x,?y)$ > $Insert\_wheel_{co}(?x,?y)$
$Base\_assembly\_car1_3(?bs)$	$Wheel\_assembly_{co}$ > $Base\_assembly_{co}(?bs,Car1)$
$Body\_assembly_3(?cr,?tp,?bs)$	$Body\_assembly_{co}(?cr,?tp,?bs)$
$Make\_car1_3$	$Assemble_{co}(Car1)$
$Make\_car2_3$	$Assemble_{co}(Car2)$

Domain Truth Table of $Planner_3$

Always True	**Always False**
$Handempty_{co}$	$Holding_{co}$

Predicate Mappings in KB_3

$Planner_3$	**Common Ontology**
$On_3 (?x,?y)$	$On_{co}(?x,?y)$
$Inserted\_wheel_3(?x,?y)$	$In\_wheel_{co}(?x,?y)$
$Clear_3(?x)$	$Clear_{co}(?x)$
$Fastened_3(?x,?y)$	$Attached_{co}(?x,?y)$
$Connected_3(?x,?y)$	$Wired_{co}(?x,?y)$
$Wheel_3(?x)$	$Wheel_{co}(?x)$
$Axle\_end_3(?x)$	$Axle\_end_{co} (?x)$
$Engine_3(?x)$	$Engine_{co}(?x)$
$Battery_3(?x)$	$Battery_{co}(?x)$
$Engine\_slot_3(?x)$	$Engine\_slot_{co}(?x)$
$Battery\_slot_3(?x)$	$Battery\_slot_{co}(?x)$
$Car\_top_3(?x)$	$Top_{co}(?x)$
$Car\_base_3(?x)$	$Base_{co}(?x)$
$Car_3(?x)$	$Car_{co}(?x)$
$Assembled\_eng\_batt_3$	$Assembled\_eng\_batt_{co}$
$Assembled\_base\_car1_3$	$Assembled\_base_{co}(Car1)$
$Assembled\_base\_car2_3$	$Assembled\_base_{co}(Car2)$
Assembled_body_3	$Assembled\_body_{co}$
$Car1_3$	$Complete_{co}(Car1)$
$Car2_3$	$Complete_{co}(Car2)$

Example

Planner_3 - Query

Query_3

I_3 = Clear_3(Axle_end1), Clear_3(Axle_end2), Clear_3(Axle_end3), Clear_3(Axle_end4), Clear_3(Wheel1), Clear_3(Wheel2), Clear_3(Wheel3), Clear_3(Wheel4), Clear_3(Eng), Clear_3(Batt), Clear_3(Eng_slot), Clear_3(Batt_slot), Clear_3(Top), Clear_3(Base),

G_3= $\text{Assembled\_base\_car2}_3$

C_3= Wheel_3(Wheel1), Wheel_3(Wheel2), Wheel_3(Wheel3), Wheel_3(Wheel4), Car_top_3(Top), Car_base_3(Base), Axle_end_3(Axle_end1), Axle_end_3(Axle_end2), Axle_end_3(Axle_end3), Axle_end_3(Axle_end4), Engine_3(Eng), Engine_slot_3(Eng_slot), Battery_3(Batt), Battery_slot_3(Batt_Slot)

Query_{co}

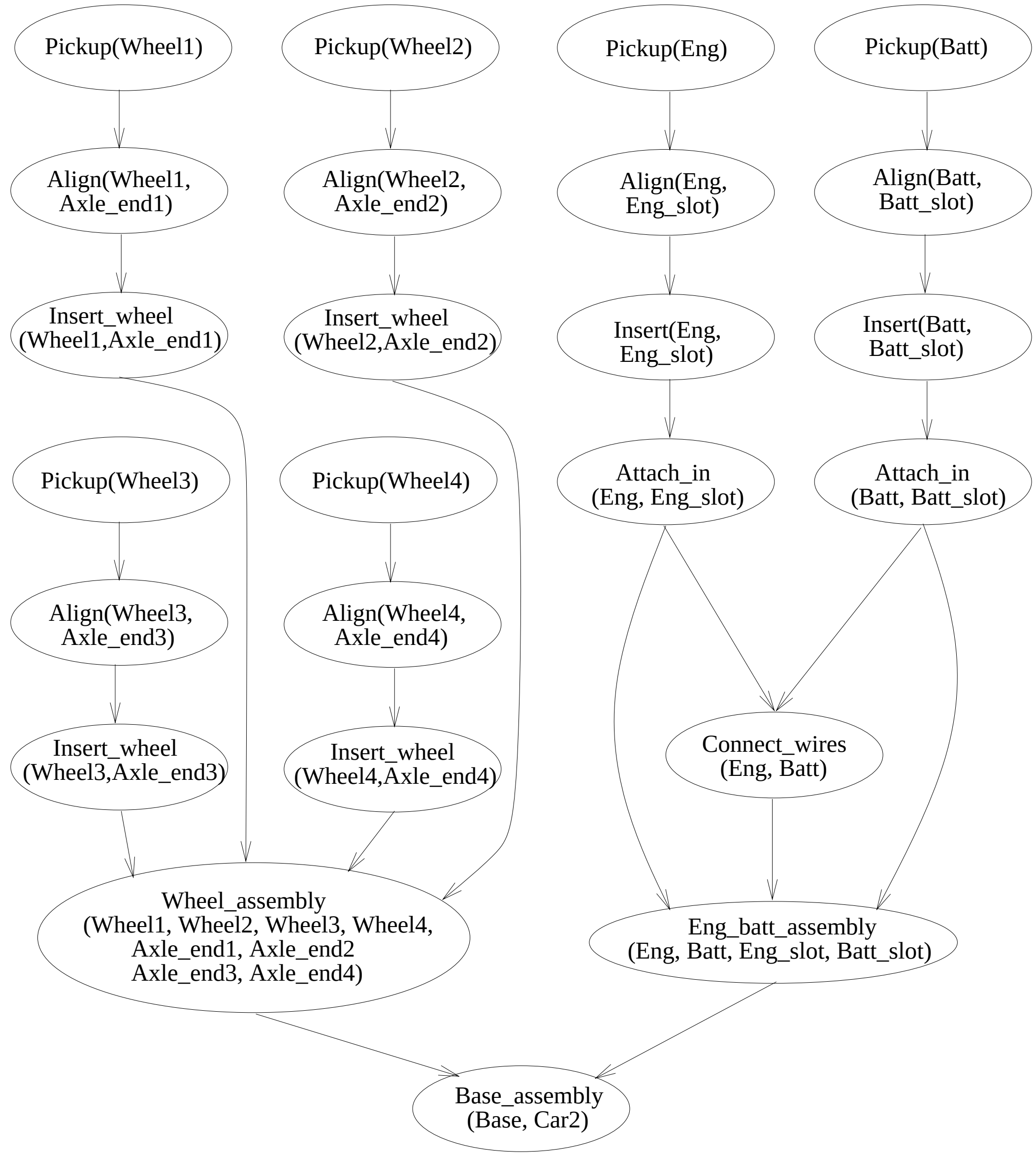
I_{co} = Clear_{co}(Axle_end1), Clear_{co}(Axle_end2), Clear_{co}(Axle_end3), Clear_{co}(Axle_end4), Clear_{co}(Wheel1), Clear_{co}(Wheel2), Clear_{co}(Wheel3), Clear_{co}(Wheel4), Clear_{co}(Eng), Clear_{co}(Batt), Clear_{co}(Eng_slot), Clear_{co}(Batt_slot), Clear_{co}(Top), Clear_{co}}(Base)

G_{co} = $\text{Assembled\_base}_{co}$(Car2)

C_{co} = Wheel_{co}(Wheel1), Wheel_{co}(Wheel2), Wheel_{co}(Wheel3), Wheel_{co}(Wheel4), Top_{co}(Top), Base_{co}(Base), Axle_end_{co}(Axle_end1), Axle_end_{co}(Axle_end2), Axle_end_{co}(Axle_end3), Axle_end_{co}(Axle_end4), Engine_{co}(Eng), Engine_slot_{co}(Eng_slot), Battery_{co}(Batt), Battery_slot_{co}(Batt_Slot), Handempty_{co}

Example

Common Ontology – Solution Plan



Example

Common Ontology – Solution Plan

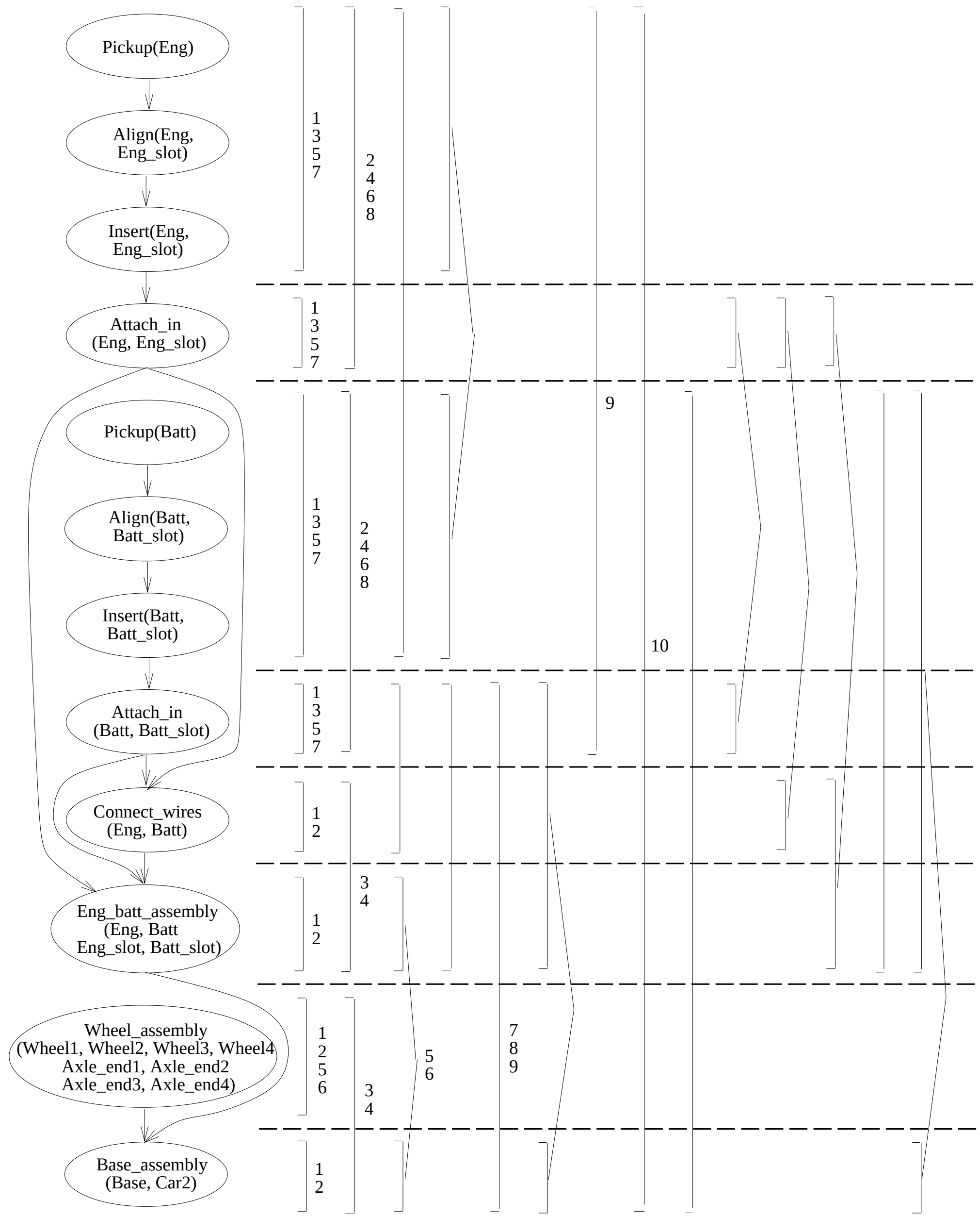
A_{co} = a_{co}^{1} = $Pickup_{co}$(Wheel1),
a_{co}^{2} = $Align_{co}$(Wheel1,Axle_end1),
a_{co}^{3} = $Insert\_wheel_{co}$(Wheel1,Axle_end1),
a_{co}^{4} = $Pickup_{co}$(Wheel2),
a_{co}^{5} = $Align_{co}$(Wheel,Axle_end2),
a_{co}^{6} = $Insert\_wheel_{co}$(Wheel2,Axle_end2),
a_{co}^{7} = $Pickup_{co}$(Wheel3),
a_{co}^{8} = $Align_{co}$(Wheel3,Axle_end3),
a_{co}^{9} = $Insert\_wheel_{co}$(Wheel3,Axle_end3),
a_{co}^{10} = $Pickup_{co}$(Wheel4),
a_{co}^{11} = $Align_{co}$(Wheel4,Axle_end4),
a_{co}^{12} = $Insert\_wheel_{co}$(Wheel4,Axle_end4),
a_{co}^{13} = Wheel_assembly(Wheel1, Wheel2, Wheel3, Wheel4, Axle_end1,Axle_end2, Axle_end3,Axle_end4)

a_{co}^{14} = $Pickup_{co}$(Eng),
a_{co}^{15} = $Align_{co}$(Eng,Eng_slot),
a_{co}^{16} = $Insert_{co}$(Eng,Eng_slot),
a_{co}^{17} = $Attach\_in_{co}$(Eng,Eng_slot),
a_{co}^{18} = $Pickup_{co}$(Batt),
a_{co}^{19} = $Align_{co}$(Batt,Batt_slot),
a_{co}^{20} = $Insert_{co}$(Batt,Batt_slot),
a_{co}^{21} = $Attach\_in_{co}$(Batt,Batt_slot),
a_{co}^{22} = $Connect\_wires_{co}$(Eng, Batt,Eng_slot,Batt_slot),
a_{co}^{23} = $Eng\_batt\_assembly_{co}$(Eng, Batt,Eng_slot,Batt_slot),
a_{co}^{24} = $Base\_assembly_{co}$(Base,Car2)

O_{co} = before(a_{co}^{1},a_{co}^{2}), before(a_{co}^{2},a_{co}^{3}), before(a_{co}^{4},a_{co}^{5}), before(a_{co}^{7},a_{co}^{6}), before(a_{co}^{7},a_{co}^{8}), before(a_{co}^{8},a_{co}^{9}),before(a_{co}^{10},a_{co}^{11}), before(a_{co}^{11},a_{co}^{12}), before(a_{co}^{3},a_{co}^{13}), before(a_{co}^{6},a_{co}^{13}), before(a_{co}^{9},a_{co}^{13}), before(a_{co}^{12},a_{co}^{13}), before(a_{co}^{14},a_{co}^{15}), before(a_{co}^{15},a_{co}^{16}), before(a_{co}^{16},a_{co}^{17}), before(a_{co}^{18},a_{co}^{19}), before(a_{co}^{19},a_{co}^{20}), before(a_{co}^{20},a_{co}^{21}), before(a_{co}^{17},a_{co}^{22}), before(a_{co}^{21},a_{co}^{22}}), before(a_{co}^{17},a_{co}^{23}), before(a_{co}^{21},a_{co}^{23}), before(a_{co}^{22},a_{co}^{23}), before(a_{co}^{13},a_{co}^{24}}), before(a_{co}^{23},a_{co}^{24})

Example

Solution Plan - Partitions



Example

Solution Plan - Partition$_{co}^{1}$

$A_{co}^{1-1} = \{a_{co}^{1}, a_{co}^{2}, a_{co}^{3}\}$

$A_{co}^{1-2} = \{a_{co}^{4}, a_{co}^{5}, a_{co}^{6}\}$

$A_{co}^{1-3} = \{a_{co}^{7}, a_{co}^{8}, a_{co}^{9}\}$

$A_{co}^{1-4} = \{a_{co}^{10}, a_{co}^{11}\ a_{co}^{12}\}$

$A_{co}^{1-5} = \{a_{co}^{14}, a_{co}^{15}, a_{co}^{16}\}$

$A_{co}^{1-6} = \{a_{co}^{17}\}$

$A_{co}^{1-7} = \{a_{co}^{18}, a_{co}^{19}, a_{co}^{20}\}$

$A_{co}^{1-8} = \{a_{co}^{21}\}$

$A_{co}^{1-9} = \{a_{co}^{22}\}$

$A_{co}^{1-10} = \{a_{co}^{23}\}$

$A_{co}^{1-11} = \{_{co}^{13}\}$

$A_{co}^{1-12} = \{a_{co}^{24}\}$

$O_{co}^{1-1} = \{(a_{co}^{1} > a_{co}^{2}), (a_{co}^{2} > a_{co}^{3})\},$

$O_{co}^{1-2} = \{(a_{co}^{4} > a_{co}^{5}), (a_{co}^{5} > a_{co}^{6})\},$

$O_{co}^{1-3} = \{(a_{co}^{7} > a_{co}^{8}), (a_{co}^{8} > a_{co}^{9})\},$

$O_{co}^{1-4} = \{(a_{co}^{10} > a_{co}^{11}), (a_{co}^{11} > a_{co}^{12})\},$

$O_{co}^{1-5} = \{(a_{co}^{14} > a_{co}^{15}), (a_{co}^{15} > a_{co}^{16})\},$

$O_{co}^{1-6} = \{\ \}$

$O_{co}^{1-7} = \{(a_{co}^{18} > a_{co}^{19}), (a_{co}^{19} > a_{co}^{20})\},$

$O_{co}^{1-8} = \{\ \}$

$O_{co}^{1-9} = \{\ \}$

$O_{co}^{1-10} = \{\ \}$

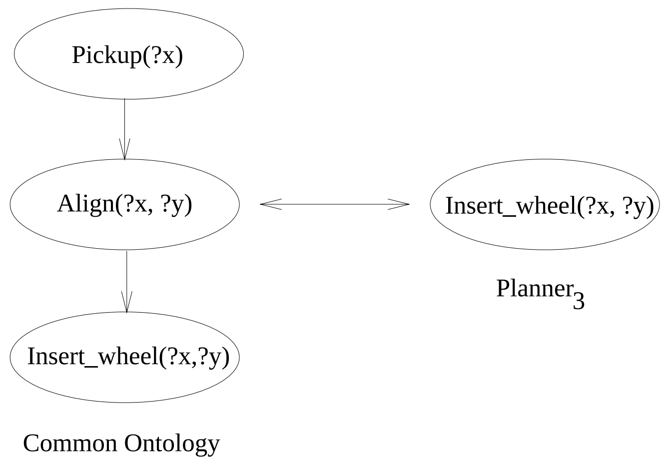
$O_{co}^{1-11} = \{\ \}$

$O_{co}^{1-12} = \{\ \}$

$O_{co}^{1-13} = \{(a_{co}^{16} > a_{co}^{17}), (a_{co}^{17} > a_{co}^{22}), (a_{co}^{17} > a_{co}^{23}), (a_{co}^{20} > a_{co}^{21}), (a_{co}^{21} > a_{co}^{22}),$
$(a_{co}^{22} > a_{co}^{23}), (a_{co}^{22} > a_{co}^{23}), (a_{co}^{23} > a_{co}^{24}), (a_{co}^{13} > a_{co}^{24})\}$

Example

Solution Plan - $\text{Partition}_{co}^{1}$



- Groups

  $<A_{co}^{1\text{-}1}, O_{co}^{1\text{-}1}>$, $<A_{co}^{1\text{-}2}, O_{co}^{1\text{-}2}>$,

  $<A_{co}^{1\text{-}3}, O_{co}^{1\text{-}3}>$, $<A_{co}^{1\text{-}4}, O_{co}^{1\text{-}4}>$

  all translate to the action

  $\text{Insert\_wheel}_3(?x,?y)$

Example

$\text{Partition}_{co}{}^{1}$ – New Actions

- Group $<A_{co}^{1\text{-}5},O_{co}^{1\text{-}5}>$ cannot be translated
- It has the following Preconditions, Add list, Delete List & Constraints

  Pre ($<A_{co}^{1\text{-}5},O_{co}^{1\text{-}5}>$) = Clear_{co}(Eng), Clear_{co}(Eng_slot)

  Add($<A_{co}^{1\text{-}5},O_{co}^{1\text{-}5}>$) = In_{co} (Eng, Eng_slot)

  Del($<A_{co}^{1\text{-}5},O_{co}^{1\text{-}5}>$) = Clear_{co} (Eng_slot)

  Con($<A_{co}^{1\text{-}5},O_{co}^{1\text{-}5}>$) = --
- The new action this would form is

  a_3^1(?x,?y)

  Pre(a_3^1) = Clear_3(?x), Clear_3(?y)

  Add(a_3^1) = **In_{co}(?x,?y)**

  Del(a_3^1) = Clear_3(?y)

  Con(a_3^1) = -
- The similarity index is SI(a_3^1,Fix_together_3) = 0.4
- The element $<A_{co}^{1\text{-}7},O_{co}^{1\text{-}7}>$ forms a new action, a_3^3, that is equal to a_3^1. Therefore, $a_3^3 = a_3^1$ and SI(a_3^3,Fix_together_3) = 0.4

Example

$\text{Partition}_{co}^{1}$ – New Actions

- Similarly, <$A_{co}^{1\text{-}6}$,$O_{co}^{1\text{-}6}$> and <$A_{co}^{1\text{-}8}$,$O_{co}^{1\text{-}8}$> form actions a_3^2 and a_3^4 which are equal and their similarity indices are

  $SI(a_3^2, Fix\_together_3) = 0.2$

  $SI(a_3^4, Fix\_together_3) = 0.2$

- Finally, <$A_{co}^{1\text{-}9}$,$O_{co}^{1\text{-}9}$>, <$A_{co}^{1\text{-}10}$,$O_{co}^{1\text{-}10}$>, < $A_{co}^{1\text{-}11}$,$O_{co}^{1\text{-}11}$}> and <$A_{co}^{1\text{-}12}$,$O_{co}^{1\text{-}12}$> form the new actions a_3^5(?w,?x,?y,?z), a_3^6(?en,?bt,?es,?bs), a_3^7(?en,?bt,?es,?bs) and a_3^8(?w,?x,?y,?z) whose similarity indices are as below.

  $SI(a_3^5, Fix\_together_3) = 0.125$

  $SI(a_3^6, Body\_assembly_3) = 0.067$

  $SI(a_3^7, Base\_assembly_3) = 0.75$

  $SI(a_3^8, Base\_assembly_3) = 0.105$

- The similarity index of the entire set is

  $SI(A_3^1, Actions_3) = 0.281$

Example

Solution Plan – Calculation Results

- Below are a few of the highest similarity indices for the remaining partitions.

  $SI(A_3^2, Actions_3) = 0.374$

  $SI(A_3^3, Actions_3) = 0.348$

  $SI(A_3^4, Actions_3) = 0.523$

  $SI(A_3^5, Actions_3) = 0.344$

  $SI(A_3^6, Actions_3) = 0.515$

  $SI(A_3^7, Actions_3) = 0.348$

  $SI(A_3^8, Actions_3) = 0.579$

  $SI(A_3^9, Actions_3) = 0.436$

  $SI(A_3^{10}, Actions_3) = 0.467$

- The set A_3^8 is most similar to the existing knowledge in $Planner_3$.

Example

Planner$_3$ - New Actions

$A_3^8 = \{a_3^1, a_3^2, a_3^3\}$, contains 3 new actions described as follows.

a_3^1(?x,?y) = a_3^2 (?x,?y)

Pre(a_3^1) = Pre(a_3^2) = Clear$_3$(?x), Clear$_3$(?y)

Add(a_3^1) = Add(a_3^2) = **In$_{co}$(?x,?y)**, Fastened$_3$(?x,?y)

Del(a_3^1) = Del(a_3^2) = Clear$_3$(?y)

Con(a_3^1) = Con(a_3^2) = -

a_3^3(?en,?bt,?es,?bs,?bs1,?cr)

Pre(a_3^3) = Inserted_wheel$_3$(?w1,?ae1),
Inserted_wheel$_3$(?w2,?ae2), Inserted_wheel$_3$(?w3,?ae3),
Inserted_wheel$_3$(?w4,?ae4), Fastened$_3$(?en,?es),
Fastened$_3$(?bt,?bs), **In$_{co}$(?en,?es)**, **In$_{co}$(?bt,?bs)**,
Wheel$_3$(?w1), Wheel$_3$(?w2),

Add(a_3^3) = Connected$_3$(?en,?bt), Assembled_base_car2$_3$

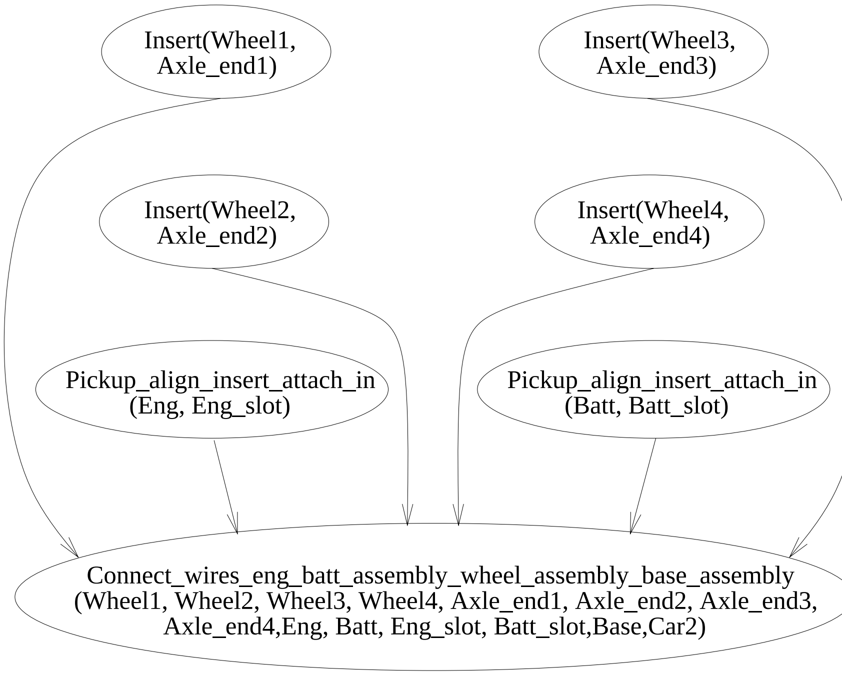
Del(a_3^3) = --

Con(a_3^3) = Wheel$_3$(?w3), Wheel$_3$(?w4), Axle_end$_3$(?ae1),
Axle_end$_3$(?ae2), Axle_end$_3$(?ae3), Axle_end$_3$(?ae4),
Engine$_3$(?en), Battery$_3$(?bt), Engine_slot$_3$(?es),
Battery_slot$_3$(?bt), Base$_3$(?bs1), **Car$_{co}$(Car2)**

Example

$Planner_3$ – New Mappings

$Planner_3$	**Common Ontology**
In_3(?x,?y)	In_{co}(x,?y)
Car_3(?x)	Car_{co}(?x)
Pickup_align_insert_attach_in_3(?x, ?y)	$Pickup_{co}$(?x) > $Align_{co}$(?x,?y) > $Insert_{co}$(?x,?y) > Attach_in_{co}(?x,?y)
Connect_wires_eng_batt_assembly_wheel_ assembly_base_$assembly_3$ (?w1,?w2,?w3,?w4,?ae1,?ae2, ?ae3,?ae4,?en,?bt,?es,?bs,?bs1,?cr)	Connect_$wires_{co}$(?en,?bt) > Eng_batt_$assembly_{co}$(?en,?bt,?es,?bs) > Wheel_$assembly_{co}$ (?w1,?w2,?w3,?w4, ?ae1,?ae2,?ae3,?ae4) > Base_$assembly_{co}$(?bs1,?cr)



Example

Planner$_3$ - Final Solution

Plan$_3$

A_3 =

a_3^1 = Insert_wheel$_3$(Wheel1, Axle_end1),

a_3^2 = Insert_wheel$_3$(Wheel2, Axle_end2),

a_3^3 = Insert_wheel$_3$(Wheel3, Axle_end3),

a_3^4 = Insert_wheel$_3$(Wheel4, Axle_end4),

a_3^5 = Pickup_align_insert_attach_in$_3$(Eng,Eng_slot),

a_3^6 = Pickup_align_insert_attach_in$_3$(Batt,Batt_slot),

a_3^7 = Connect_wires_eng_batt_assembly_wheel_assembly_base_assembly$_3$(Wheel1, Wheel2, Wheel3, Wheel4, Axle_end1, Axle_end2, Axle_end3, Axle_end4, Eng, Batt, Eng_slot, Batt_slot, Base, Car2)

O_3 = before(a_3^1, a_3^7), before(a_3^2, a_3^7), before(a_3^3, a_3^7), before(a_3^4, a_3^7), before(a_3^5, a_3^7), before(a_3^6_3, a_3^7)

Related Work

Knowledge Interchange Interface - A formal framework for knowledge sharing between planners.

- Given $Plan_{co}$, the KII learns new knowledge from it as follows.
  - If action a^k_{co} contains predicates that are always true or false, it is combined with one or more actions to eliminate these predicates. The resulting composite action is added to $Planner_i$.
  - All other actions are added without any change to $Planner_i$
  - All new predicates in the actions are added to $Pred_i$.
  - Translators are formulated for the actions and predicates and added to KB_i.
- Drawbacks
  - A subplan in $Plan_{co}$ that have an equivalent action in $Planner_i$, but no translator get mapped to new actions, causing duplication of knowledge in $Planner_i$.
  - Does not handle combinations of subplans in a flexible manner.
  - Adds the same knowledge to planners, regardless of the level of abstraction.

Related Work

Learning through Exploration and Experimentation - improve the models for planning by using failure-driven experimentation with the environment

- Construction by direct analogy - creates new operators based on operators that are already defined in the system. This is done by direct analogy through the types of objects that the operators are applied to.
- Creation of micro-operators – creates a new operator that performs some subaction of an existing operator.
  - Experimentation is used to determine which preconditions and effects can be discarded in the new operator.
  - A existing action is broken into a sequence of subactions.
- Drawbacks
  - Refines existing knowledge in the system.
  - Does not learn anything new from external sources.

Related Work

Explanation-Based Learning - generalize a set of given examples into descriptions of concepts.

- Learning composite rules - alters the search space that is explored by combining existing rules to form new, possibly redundant but more efficient, rules. improves the performance of a system without the introduction of new knowledge
- Learning search control knowledge - altering the method in which the search is conducted. Speeds up the search of the concept hypothesis space by formulating heuristics that index the operators in an appropriate order.

Plan Mining - used to discover significant knowledge from a large database of plans such as airline schedules etc.

- Generalization-based divide-and-conquer - used to find high-level, highly regular sequential patterns along more than one dimension.

Conclusion

- The knowledge acquisition methodology allows a planner in a collection of networked planners to update its domain theory using solution plans that is receives from its peers.
- The information (predicates and actions) contained in a solution plan that cannot be translated is used to extend the knowledge base of the receiving planner.
- The addition of the the necessary translators for the new knowledge means that it can be used to solve problem and communicate queries and solution plans in the future.
- The methodology guarantees that this information is correct in the domain of the planner and it is the most similar to the existing knowledge in the planner.