Hierarchical Planning

Planning

- *Sequence* of *actions* worked out beforehand
- In order to accomplish a task

Example : One level planner

- Planning for ”Going to Goa this Cristmas”
  - Switch on computer
  - Start web browser
  - Open Indian Railways website
  - Select date
  - Select class
  - Select train
  - ... so on
- Practical problems are too complex to be solved at one level

How Complex ?

- A captain of a cricket team plans the order of 5 bowlers in 2 days of a test match(180 overs).
  - Number of possibilities : $5^{180} = 25^{90}$
  - Much greater than 10^{87} (approx. number of particles in the universe)

Hierarchy in Planning

- Hierarchy of actions
  - In terms of *major* action or *minor* action

- Lower level activities would detail more precise steps for accomplishing the higher level tasks.

Ref : [1,2]

Example

- Planning for ”Going to Goa this Cristmas”
  - Major Steps :
    - Hotel Booking
    - Ticket Booking
    - Reaching Goa
    - Staying and enjoying there
    - Coming Back
  - Minor Steps :
    - Take a taxi to reach station / airport
    - Have candle light dinner on beach
    - Take photos

Motivation

- Reduces the size of search space

  Instead of having to try out a large number of possible plan ordering, plan hierarchies limit the ways in which an agent can select and order its primitive operators

Ref : [4]

Example

- 180 overs : 15 spells (12 overs each)
- 5 bowlers : 3 categories (2 pacer/2 spinner/1 pacer&1 spinner)

- Top level possibilities : 3^{15}
- Total possibilities < $3*3^{15}$ (much less than 5^{180})

Motivation contd...

- If entire plan has to be synthesized at the level of most detailed actions, it would be ***impossibly long.***

- Natural to 'intelligent' agent

Ref : [1]

General Property

- ***Postpone*** attempts to solve mere details, ***until*** major steps are in place.

- Higher level plan may run into difficulties at a lower level, causing the need to return to higher level again to produce appropriately ordered sequence.

Ref : [1,2]

Planner

- Identify a hierarchy of conditions
- Construct a plan in levels, postponing details to the next level
- Patch higher levels as details become visible
- Demonstrated using ABSTRIPS

Ref : [1,2]

ABSTRIPS

- Abstraction-Based STRIPS
- Modified version of STRIPS that incorporates hierarchical planning

Ref : [1,2]

Hierarchy in ABSTRIPS

- Hierarchy of conditions reflect the intrinsic difficulty of achieving various conditions.
- Indicated by criticality value.

Ref : [2]

Criticality

- A operation having *minimum criticality* can be trivially achievable, i.e., the operations having very less or no precondition.
  - Example : Opening makemytrip.com
- Similarly operation having many preconditions to satisfy will have higher criticality.

Patching in ABSTRIPS

- Each level starts with the goal stack that includes the plan obtained in the higher levels.
- The last item in the goal stack being the main goal.

Ref : [2]

Example

- Actions required for “Travelling to Goa”:
  - Opening makemytrip.com (1)
  - Finding flight (2)
  - Buy Ticket (3)
  - Get taxi(2)
  - Reach airport(3)
  - Pay-driver(1)
  - Check in(1)
  - Boarding plane(2)
  - Reach Goa(3)

Example

- 1st level Plan :
  - Buy Ticket (3), Reach airport(3), Reach Goa(3)
- 2nd level Plan :
  - Finding flight (2), Buy Ticket (3), Get taxi(2), Reach airport(3), Boarding plane(2), Reach Goa(3)
- 3rd level Plan (final) :
  - Opening makemytrip.com (1), Finding flight (2), Buy Ticket (3), Get taxi(2), Reach airport(3), Pay-driver(1), Check in(1), Boarding plane(2), Reach Goa(3)

Observation

- As the number of operator increases, performance of hierarchical planning comes out to be much better than one level planning

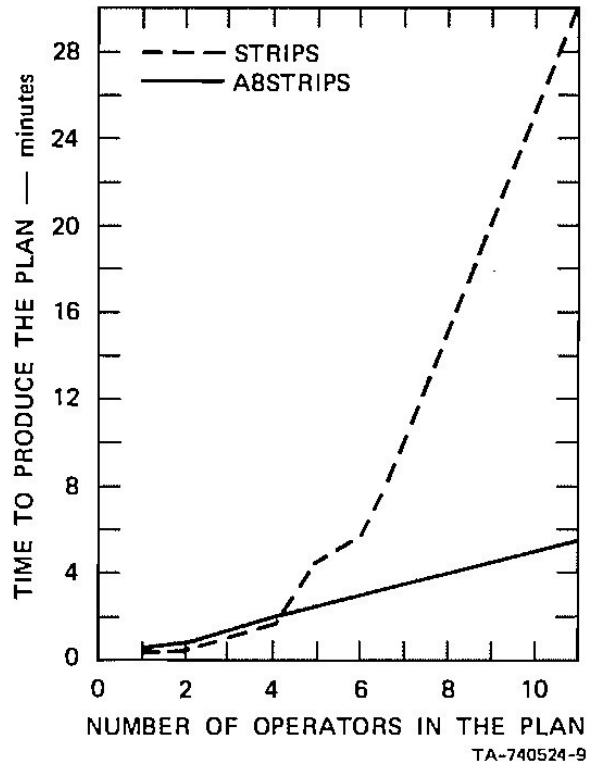
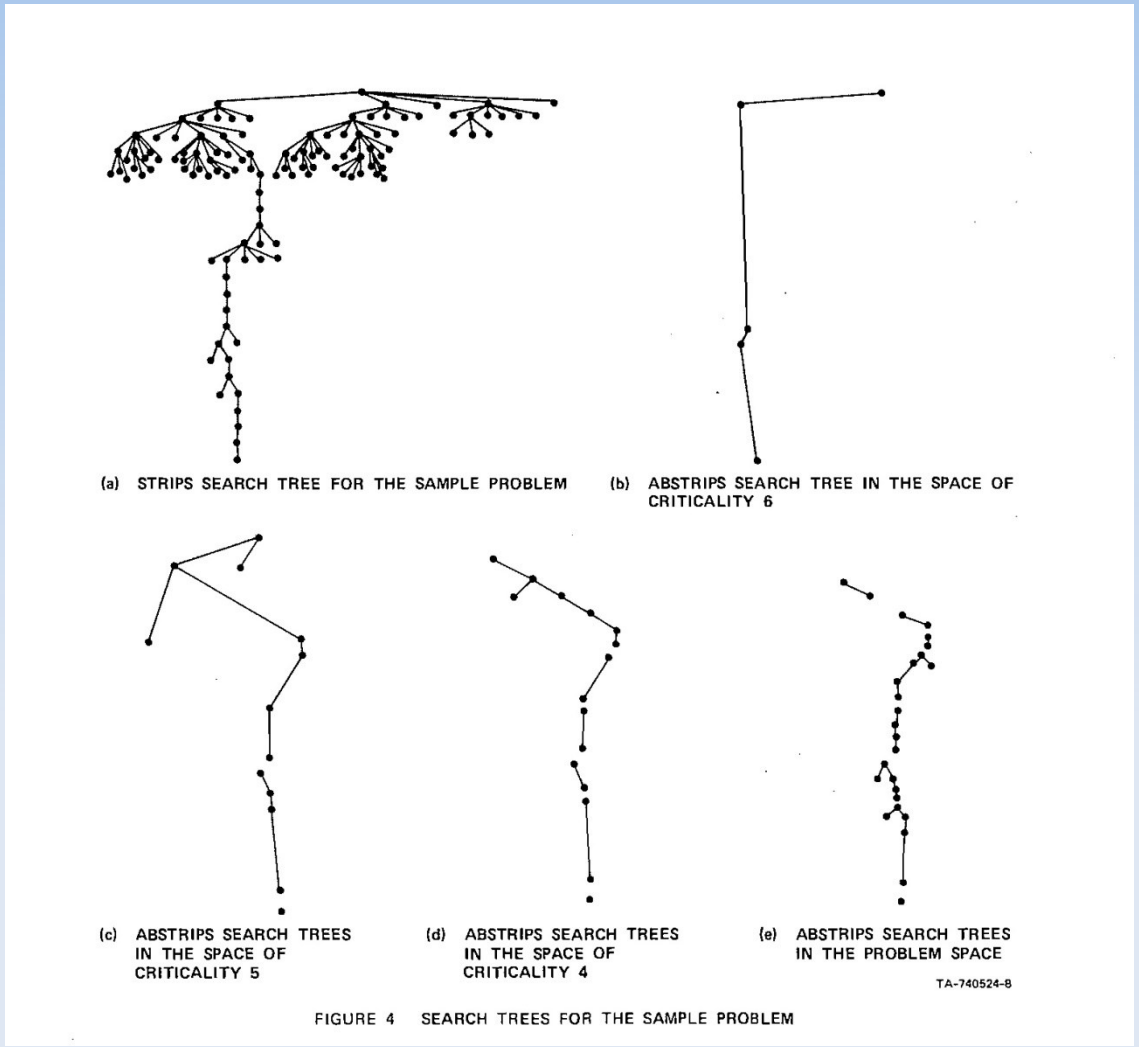


FIGURE 5 PLANNING TIME Vs. NUMBER OF OPERATORS

Ref : [1]

Observation contd…

- Search trees for STRIPS and ABSTRIPS for a sample problem
- Shows reduction in nodes explored



Ref : [1]

Task Network

- Collection of ***task*** and ***constraints*** on those tasks
- $((n_1, \alpha_1)$,…, $((n_m, \alpha_m)$,$\phi)$, where α_1 is task labeled with n_1 ,and boolean formula expressing constraints.
  - Truth constraint : (n, p, n') means p will be true immediately after n and immediately before n'.
  - Temporal ordering constraint : n $\prec$ n' means task n precedes n'.
  - Variable binding constraint : Λ,V, $=$, $\sim$ etc.

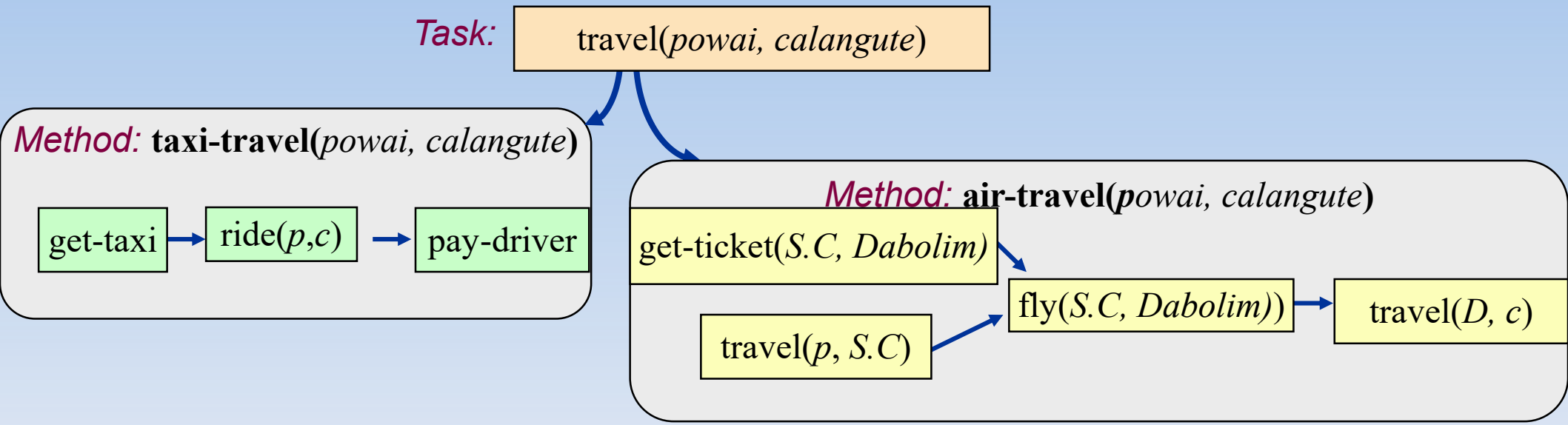
Ref : [3]

Hierarchical Task Network

- Hierarchy abstraction achieved through *methods*.
- A method is a pair (α, d) , where
  - α is the non-primitive task, and
  - d is the task network to achieve the task α

Ref : [3]

HTN examples



- ((n$_1$:get-taxi), (n$_2$:ride(x, y)), .., (n$_4$:get-ticket), (n$_5$:travel(x, a(x)), (n$_6$:fly(a(x),a(y)) … , ((n$_1$≺n$_2$)..)∨((n$_4$ ≺ n$_6$)∧(n$_5$ ≺ n$_6$)…)

Ontology and Hierarchical Planning

- Hierarchical planning in real world requires modeling an efficient, semantic, and flexible knowledge representation for both planning and domain knowledge.

- Ontology helps to conceptualize the hierarchy of operators and domain.

Ref : [5]

Example

- To perform operation ‘Buy ticket’ agent has to understand concept of ‘Buy’ and ‘ticket’
- Buy is an action, between seller and customer, involves finding a seller, customer should have money to buy etc.
- Ticket is an object, which has some price, has particular owner, has some validity etc.
- This conceptualizations are extremely important for planning in that domain.

Ref : [5]

Conclusion

- For complex problems hierarchical planning is much more efficient than single level planning.
- Improves performance as number of operator in the problem increases.
- HTN planning gives more expressivity
- Merging opens door to accomplish a complete plan from incomplete individual plans
- Integration with ontology opens door for automatic planning
- Reduces man machine gap.

References

1) E.D. Sacerdoti, Planning in a hierarchy of abstraction spaces, in: *Proc. of the 3rd International Joint conference on Artificial Intelligence*, 1973
2) Nils J. Nilsson: *Principles of Artificial Intelligence,* Springer 1982.
3) K. Erol, J. Hendler, and D. S. Nau. HTN planning: Complexity and expressivity. in: National Conference on Artificial Intelligence (AAAI), 1994
4) Jeffrey S. Cox and Edmund H. Durfee, 'Discovering and Exploiting Synergy Between Hierarchical Planning Agents', in: Second International Joint Conference On Autonomous Agents and Multiagent Systems, 2003
5) Choi H J Kang D, 'Hierarchical planning through operator and world abstraction using ontology for home service robots' ,in: *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on, 2009*