Planners To Be Discussed

STRIPS

A* GOAP

Hierarchical Task Networks
Figure 1. A Brief Chronology of Some Well-Known Planning Systems.

The numbers in parentheses represent systems on which each planner has directly built (also shown as solid lines where possible). The dotted lines represent some of the important outside areas influencing the development of planning systems.

Figure 1: Methods for transporting a package \( ?p \), transporting two packages \( ?p \) and \( ?q \), dispatching a truck \( ?t \), and returning the truck. Arrows are ordering constraints. The shaded subtasks are *primitive* tasks that are accomplished by the following planning operators: (load \( ?t \) \( ?p \)) loads \( ?p \) onto \( ?t \); (move \( ?t \) \( ?x \) \( ?y \)) moves \( ?t \) from \( ?x \) to \( ?y \); (reserve \( ?t \)) deletes (available-truck \( ?t \)) to signal that the truck is in use; (free \( ?t \)) adds (available-truck \( ?t \)) to signal that the truck is no longer in use.
Figure 2: A plan for accomplishing (transport-two p1 p2) from the following initial state: 
{(package p1), (at p1 l1), (destination p1 l3), (available-truck t1), (at t1 home),
(package p2), (at p2 l2), (destination p2 l4), (available-truck t2), (at t2 home)}. 

SHOP2 Example

Figure 3: A SHOP2 method for a simplified version of the ZenoTravel domain.

procedure SHOP2(s, T, D)
    P = the empty plan
    T₀ ← \{t ∈ T : no other task in T is constrained to precede t\}
    loop
        if T = ∅ then return P
        nondeterministically choose any t ∈ T₀
        if t is a primitive task then
            A ← \{(a, θ) : a is a ground instance of an operator in D, θ is a substitution that unifies \{head(a), t\}, and s satisfies a's preconditions\}
            if A = ∅ then return failure
            nondeterministically choose a pair (a, θ) ∈ A
            modify s by deleting del(a) and adding add(a)
            append a to P
            modify T by removing t and applying θ
            T₀ ← \{t ∈ T : no task in T is constrained to precede t\}
        else
            M ← \{(m, θ) : m is an instance of a method in D, θ unifies \{head(m), t\}, pre(m) is true in s, and m and θ are as general as possible\}
            if M = ∅ then return failure
            nondeterministically choose a pair (m, θ) ∈ M
            modify T by removing t, adding sub(m), constraining each task in sub(m) to precede the tasks that t preceded, and applying θ
            if sub(m) ≠ ∅ then
                T₀ ← \{t ∈ sub(m) : no task in T is constrained to precede t\}
            else T₀ ← \{t ∈ T : no task in T is constrained to precede t\}
        end
    end
repeat
end SHOP2

Figure 5: A simplified version of the SHOP2 planning procedure.

SHOP2's Algorithm

```
procedure SHOP2(s, T, D)
    P = the empty plan
    T_0 ← {t ∈ T : no other task in T is constrained to precede t}
    loop
        if T = ∅ then return P
        nondeterministically choose any t ∈ T_0
        if t is a primitive task then
            A ← {(a, θ) : a is a ground instance of an operator in D, θ is a substitution that unifies {head(a), t}, and s satisfies a's preconditions}
            if A = ∅ then return failure
            nondeterministically choose a pair (a, θ) ∈ A
            modify s by deleting del(a) and adding add(a)
            append a to P
            modify T by removing t and applying θ
            T_0 ← {t ∈ T : no task in T is constrained to precede t}
        else
            M ← {(m, θ) : m is an instance of a method in D, θ unifies {head(m), t}, pre(m) is true in s, and m and θ are as general as possible}
            if M = ∅ then return failure
            nondeterministically choose a pair (m, θ) ∈ M
            modify T by removing t, adding sub(m), constraining each task in sub(m) to precede the tasks that t preceded, and applying θ
            if sub(m) ≠ ∅ then
                T_0 ← {t ∈ sub(m) : no task in T is constrained to precede t}
            else T_0 ← {t ∈ T : no task in T is constrained to precede t}
        end
    repeat
end SHOP2
```

Figure 5: A simplified version of the SHOP2 planning procedure.

Similar to Behavior Trees

Tasks

Conditions/Preconditions

Sequence

World Representation

Problem Decomposition
Similar to Behavior Trees

- Gate Unlocked?
- Open Gate
- Enter
- Thief Tools?
- Pick Lock
- Open Gate
- Big Weapon?
- Smash Gate
Tasks

Enter Compound

OpenGate

Enter

Unlock Gate (inv plasma cutter)

OpenGate (gate is unlocked)

Open

UnlockGate

UnlockGate (inv thief tools)

UnlockGate (inv big weapon)

UnlockGate (inv plasma cutter)

OpenGate (gate is locked)

Open

Pick Lock

Big Weapon?

Smash Gate
Killzone 2 Bots

(:method (select_weapon_and_attack_as_turret ?inp_threat)
  ( branch_use_bullets  // Only use bullets against humanoids and turrets.
   (and (or (threat ?inp_threat humanoid) (threat ?inp_threat turret))
     (distance_to_threat ?inp_threat ?threat_distance)
     (call lt ?threat_distance @weapon_bullet_max_range) )
   (attack_as_turret_using_weapon_pref ?inp_threat wp_bullets))
)

( branch_use-rockets // Don't use rockets against humanoids and turrets.
  (and (not (threat ?inp_threat humanoid)) (not (threat ?inp_threat turret))
    (distance_to_threat ?inp_threat ?threat_distance)
    (call lt ?threat_distance @weapon_rocket_max_range) )
  ((attack_as_turret_using_weapon_pref ?inp_threat wp_rockets))
) )

http://aigamedev.com/open/coverage/htn-planning-discussion/
Killzone 2 Bots

+ branch_mp_behave
  + (do_behave_on_foot_mp)
  + branch_medic_revive
    + (do_medic_revive)
      - branch_medic_revive_abort
      - branch_medic_revive_continue
      + branch_medic_revive
        (!forget active_plan **)
        (!remember - active_plan medic_revive [Soldier:TimmermanV])
        (!log_color magenta “Medic reviving nearby entity.”)
        (!broadcast friendlies 30.0 10.0 medic_reviving [Soldier:TimmermanV])
        (!select_target [Soldier:TimmermanV])
        + (walk_to_attack 5416 crouching auto)
        + (wield_weapon_pref wp_online_mp_bot_revive_gun)
          - branch_auto_and_have_active
          - branch_auto_wp_pref
          - branch_dont_switch_weapon
          + branch_switch_weapon
            (#0 = wp_online_mp_bot_revive_gun)
            + (wield_weapon_pref_internal wp_online_mp_bot_revive_gun)
        (!use_item_on_entity [Soldier:TimmermanV] crouching)
        (!forget active_plan **)