Plan Space Search

- state-space search: 节点表示系统状态，边为 action，最后解的形式是 action 序列
- plan-space search: 节点表示 partially specified plans，边为 plan refinement operations，解的形式是 partial-order plans

Partial Plan

Partial Plan 即不完整的 Plan。一个完整的 Plan 是一个已经实例化的 action 的序列，partial plan 的不完整性可以体现在以下几个方面：

- action 不完整，包含的 action 可能只是完整 Plan 的一个子集
- action 之间的顺序信息不完整
- action 之间的依赖关系不完整，例如 applicable 和 relevant 信息
- action 的变量赋值不完整

从这里也可以看出来，plan-space search 中的 action 其实更接近 operator 的概念。

Partial Plan 可以由四元组 $\pi = (A, \prec, B, L)$ 定义

- $A = \{a_1, ..., a_k\}$: 一组被 partially instantinated 的 operator。也就是说其中的 action 可能仍然有 variable 没有被赋值。而且 A 并不包含 action 之间的顺序信息。
- $\prec$ Ordering constraints. 是 A 中 action 之间的顺序限制，例如 $a_i \prec a_j$
- B Binding constraints. 是 A 中 action 的赋值限制，定义了某个 action 的变量取值条件，比如 $x = y, x \neq y, x \in D_x$ 之类。
- L Causal links. 每个 causal link 更详细的描述了两个 action 之间的依赖关系。例如 $\langle a_i - [p] \rightarrow a_j \rangle$ 可以表示
  - $(a_i \prec a_j) \in \prec$
  - proposition p 是 a_i 的 effect 和 a_j 的 precondition，a_i 称为 p 的 producer a_j 称为 p 的 consumer
  - p 中与 a_i a_j 有关的变量赋值 $\in B$

Causal Link 是比较特殊的一个组成部分，其最核心的作用之一是避免 action 之间的冲突。例如我们有 causal link $\langle a_i - [p] \rightarrow a_j \rangle$ 和另一个希望加在 a_i 和 a_j 之间的 action a_k，那么 a_k 的 effect 就不能与这个 causal link 冲突，否则 a_k 就会破坏掉 a_i 的 effect 达成的 a_j 的 precondition。

Causal Link 的另一个作用是维护了我们已经达成的 precondition 的信息。如果一个 action 的某个 precondition 有一个接入的 causal link，那就说明这个 precondition 已经被满足了。

Plan Refinement Operations

Plan Refinement 是 plan-space search 中的边，将一个 partial plan 转变为另一个更完整的 partial plan。

- Adding Actions: 向 partial plan 中添加新的 action，通常是为了满足某些 preconditions 或者 goal conditions。
- Adding Causal Links：Causal Link 一定程度上描述了添加一个 action 的原因。而 causal link 中的 proposition 的 provider 和 consumer 可能是
  - provider：可能是一个 action 的 effect，也可能是 initial state 中一个 atom
  - consumer：可能是一个 action 的 precondition，也可能是一个 goal condition
- Adding Variable Bindings：Partial Plan 中的 Variable Binding 不仅仅局限于赋值，还可以是取值范围，或者 “和另一个变量相等” 这样的限制条件。Variable Binding 主要为了以下两个目的添加：
  - 为了将 operator 变成 solution 中的 action
  - 为了统一 provider 和 consumer 中的变量
- Adding Ordering Constraints：Ordering Constraints 是两个 action 在 partial plan 中的相对顺序。Ordering Constraints 主要为了以下几个目的添加
  - 所有的 action 必须在 initial state 之后
  - 所有的 action 必须在 goal 之前
  - 由 causal link 的添加引入
  - 避免 action 的冲突，例如不要把会破坏 causal link 的 action 放在中间，或者在有相同参数的 action 之间添加 causal link 以避免同时执行他们。这和下面提到的 Treat 概念有关。

Plan Space Search Problem

Initial Search State

这里指的是 Search 算法的初始状态，而不是 initial state

在 plan-space search 中，initial state 和 goal 都可以被 causal link 连接，或者出现在 ording constraints 里面。

Initial state 和 goal 可以用 dummy action 表示

- init state：没有 precondition，自己本身作为 effects
- goal：只有 precondition，没有 effects

这样的话初始情况下的 empty plan 为 $\pi_0 = (A, \prec, B, L) = (\{\text{init}, \text{goal}, (\text{init} \prec \text{goal}), \{\}, \{\})$

Successor Function

Successor Function 也就是前面提到的四种 Adding 操作，只是这四种操作有些情况下是需要一起执行的。例如添加一个 causal link 同时也要添加一个 ordering constraint 以及对应的 variable binding。

Goal Test：Flawless Partial Plan

首先明确以下几点

- Plan-Space Search 的最终结果依然是 partial plan，也就意味着 action 的顺序和 variable binding 可能仍然是不完全的。
- 一个 Partial Plan 是完备的，必须保证这个 partial plan 的所有可能的 action 顺序和 variable binding 都是正确的可以达到 goal 的。
- 仅仅保证 goal 作为 dummy state 的所有 precondition 都有 causal link 满足并不能保证 partial plan 是完备的，因为仍然会有下面提到的 Threat 存在。

Threat（威胁）：可以理解为当前的 partial plan 仍然存在的潜在冲突。例如，现有的 casusal link 和 ordering constraint 并不能完全避免 action a 的 effect 被 action b 破坏。而这通常意味着我们需要添加更多的 ordering constraints。

Threat 定义：当一个 action a_k 满足下面所有条件时，我们说这个 action 是 causal link $\langle a_i - [p] \rightarrow a_j \rangle$ 的一个 Threat

- a_k 的一个 effect $\neg q$ 可能和 p 矛盾。之所以说 “可能”，是因为 p q 中仍然可能有变量，而 Threat 只需要这些变量的取值可能导致矛盾。
- 新的 ordering constraints $(a_i \prec a_k)$ 和 $(a_k \prec a_j)$ 可以被添加进 $\prec$ 而不导致矛盾，换句话说 a_k 可以在 a_i a_j 之间执行。
- 如果第一条中的 p q 可以通过适当的 Variable Binding 来达到一致，那么这个 Binding 不能和 B 矛盾。

Flaw(漏洞): 可以理解为当前 partial plan 仍然不完善的地方，它可以是

- 一个 Threat
- 一个还没有被满足的 sub-goal。这不单单可以是最终的目标，也可以是某个 action 还没有 causal link 来支持的 precondition。

一个 Partial Plan π 可以作为一个 planning problem $\mathcal{P} = (\Sigma, s_i, g)$ 的 solution 的条件是

- π 没有 flow
- $\prec$ 是无环的
- B 中的赋值没有矛盾

从上面的条件，获得一个 solution 的基本思路就是保持 $\prec$ 和 B 合法性的同时，减少 flow。