

AI Planning

吉建民

USTC

jianmin@ustc.edu.cn

2024 年 4 月 9 日

Used Materials

Disclaimer: 本课件大量采用了网络课程课件，也采用了 GitHub 中开源代码，以及部分网络博客内容

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

Automatic Planning

- ▶ Planning in Artificial Intelligence is decision making about the actions to be taken.
- ▶ 自动规划 (Automatic Planning) 起源于 60 年代，是人工智能的一个重要领域。近年来，自动规划在问题描述和问题求解两方面分别取得新的突破，从而在理论和应用方面取得长足进展，应用在智能机器人、网络服务、自动驾驶等
- ▶ 自动规划有两大任务：
 - ▶ (1) 问题描述，如何方便 (紧凑、便于计算) 的表示规划问题
 - ▶ (2) 问题求解，如何高效的求解规划问题 (等价于一个搜索问题)
- ▶ 我们从这两个方面出发，介绍人工智能规划领域的主要工作，包括经典规划 (Classical Planning)，新经典规划 (Neoclassical Planning)，不确定规划 (Planning under Uncertainty)

规划方法分类

规划方法可以分为以下三类

- ▶ 领域特定 (Domain-specific): 针对具体领域专门设计的特定规划方法。通常会利用领域特性, 设计出更高效的算法
- ▶ 领域无关 (Domain-independent): 不针对具体领域的通用规划方法。相较于领域特定的规划方法有以下不同:
 - ▶ 领域无关规划是对规划方法共性的研究, 其成果可以用来提高领域特定规划方法的效率
 - ▶ 是对通用规划行为的研究, 从人工智能的角度研究其所表现出的理性行为
 - ▶ 对具体问题可以直接应用, 相对于领域特定方法是更廉价的规划器
 - ▶ 是一般智能体所需要具备的通用规划能力
- ▶ 可配置 (Configurable): 在领域无关方法的基础上, 针对具体问题可以增加控制信息, 从而利用问题领域特征, 使规划更高效

在 AI 领域, 规划 (Planning) 一般指 Domain-independent 和 Configurable Planning

智能规划

- ▶ 规划领域主要两个工作：如何方便的（紧凑并且方便求解）表达和如何高效求解（搜索）

- ▶ 表示：状态、行动

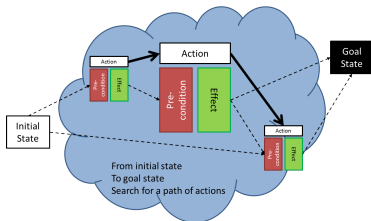
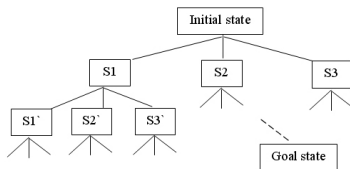
- ▶ **状态**：（动态）系统在某时刻的情况，问题的状态描述

初始状态、目标状态

- ▶ **行动**：状态空间上的部分映射，对状态描述进行变换的一组操作

- ▶ 求解：计算（搜索）出从初始状态变化到目标状态的一个操作序列

- ▶ 状态空间搜索，偏序规划，规划图，
Planning as {SAT, CSP, ILP, ...}, 等

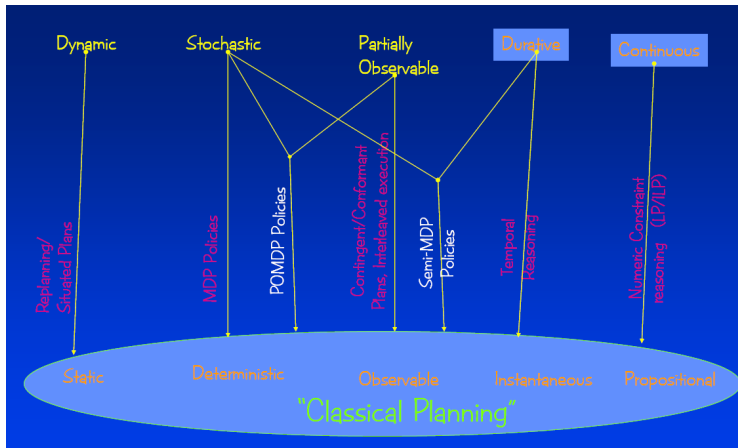


规划问题基本假设

- ▶ 规划问题非常复杂，为简化问题而提出一些简化的假设（经典规划基本假设）：
 - ▶ (A0) 有限系统 (Finite system): 问题只涉及有限的状态、行动、事件等
 - ▶ (A1) 完全可观察 (Fully observable): 永远知道系统当前所在的状态
 - ▶ (A2) 确定性 (Deterministic): 每个行动只会导致一种确定的影响
 - ▶ (A3) 静态性 (Static): 不存在外部行动，环境所有的改变都来自控制者的行动
 - ▶ (A4) 状态目标 (Attainment goals): 目标是一些需要达到的目标状态
 - ▶ (A5) 序列规划 (Sequential plans): 规划结果是一个线性行动序列
 - ▶ (A6) 隐含时间 (Implicit time): 不考虑时间持续性
 - ▶ (A7) 离线规划 (Off-line planning): 规划器不考虑执行时状态

规划问题分类

利用上述假设我们可以区分所研究的规划问题，其中经典规划问题就是满足从 A0 到 A7 所有假设的规划问题



经典规划扩展

- ▶ 经典规划虽然做了很多简化，但复杂性还是很高，在实际应用中很难使用
- ▶ 新经典规划在这方面带来了可喜的进展，分别用图规划技术、SAT 技术、CSP 技术等方法来解决经典规划问题
- ▶ 另一方面，现实不可避免的具有不确定性 (uncertainty)，因为：
 - ▶ 信息不完全性: 对世界的描述是不可能完全的
 - ▶ 不可预测性: 外部事件的发生是不可预测的
 - ▶ 行动不确定性: 有些行动效果本质上就是不确定，如投骰子
- ▶ 对不确定性的描述可以采用非确定性 (Nondeterministic) 方法，也可以采用概率方法; 对于环境观察可以是完全的，也可以是部分的或者完全不可观察的

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

经典规划基本假设

- ▶ 经典规划就是满足从 A0 到 A7 所有假设的规划问题，这种系统是确定的，静态的，有限的，完全可观察的，并且是受限目标和隐藏时间的状态转移系统
- ▶ 经典规划的任务，简单说就是：Computing paths from an initial state to a goal state in the transition graph.
 - ▶ 已知 transition graph，用 Dijkstra 算法（时间复杂度为 $O(n \log n)$ ）
 - ▶ 但状态空间一般非常大 ($10^9, 10^{12}, 10^{15}, \dots$)，所以无法构造出整个 transition graph
 - ▶ 规划算法要避免构造出整个 transition graph

经典规划

- ▶ 经典规划只考虑确定的、静态的、有限的、完全可观察的、离散环境的、目标受限和忽略时间的状态转移系统
- ▶ 经典规划的主要问题包括：
 - ▶ 如何在不显式枚举的情况下，形式化描述系统状态和动作。描述本身要紧凑，同时便于求解
 - ▶ 如何在选定描述的基础上，有效的进行解的搜索
- ▶ 即使在受限条件下，规划问题的求解仍然是非常困难的，奢求用经典规划技术来解决实际规划问题是不现实的
- ▶ 需要用一种通用的表达方式，compact 表达状态和动作，并且便于搜索求解，一般的思路是：
 - ▶ 用 features 来表达单个状态。状态为 features 特定值的集合
 - ▶ 用 operators 来计算状态转移。operators 为 features 之间的转换关系
 - ▶ 不需显示表达出所有状态，只给出初始状态，然后通过 operators 计算出需要的状态

经典规划的集合论表达 (Set-Theoretic Representation)

- ▶ 用有限的命题符号集 (L) 来表达状态转移系统：
 - ▶ $S \subseteq 2^L$, 状态 s 为 L 的子集, 命题的集合, 表示在 s 上真的命题
 - ▶ action 是三元组 $\langle precondition, effect^-, effect^+ \rangle$
 - ▶ S 的性质: 对于任意状态 s 和可应用于 s 的行动 a , 集合 $(s \setminus effect^-(a)) \cup effect^+(a) \in S$
 - ▶ 如果 a 可应用于 s , 则状态转移函数为 $\gamma(s, a) = (s \setminus effect^-(a)) \cup effect^+(a)$, 否则无定义
- ▶ 规划问题是在状态转移系统的基础上增加初始状态和目标状态集
 - ▶ $s_0 \in S$
 - ▶ $g \subseteq L$, 目标状态集合为 $S_g = \{s \in S \mid g \subseteq s\}$
- ▶ 规划是一个动作序列 $\pi = \langle a_1, \dots, a_k \rangle$, 如果 $g \subseteq \gamma(s_0, \pi)$, 则规划 π 是此规划问题的一个解
- ▶ 并不是每一个状态转移系统都可以用集合论表达, 但可以构造一个与其等价的系统, 而此系统可以用集合论表达

经典规划的经典表达 (Classical Representation)

- ▶ 经典表达是对集合论表达的推广，使用一阶逻辑符号，用公式来表达状态集和行动，通过语义解释来确定具体的状态和行动
 - ▶ 用一阶逻辑语言（有限多谓词，变元，常元，没有函数符号）来描述系统。一个状态是一个 grounded atoms 集合。谓词又分为状态谓词 (fluent) 和刚性关系 (rigid relation)，前者是状态集的函数，后者不随状态变化而变化
 - ▶ 规划操作是一个三元组 $o = (name(o), precondition(o), effects(o))$ ，其中：
 - ▶ $name(o)$ ，操作的名字，形如 $n(x_1, \dots, x_k)$
 - ▶ $precond(o)$ 和 $effects(o)$ 分别是 o 的前提和效果，都是文字集。刚性关系不能出现在任何操作 o 的效果中。action 是 ground instance of an operator
- ▶ 例如： $move(r, l, m)$ 表示机器人 r 从位置 l 移动到位置 m ：
 - ▶ $precond: adjacent(l, m), at(r, l), \neg occupied(m)$
 - ▶ $effect: at(r, m), occupied(m), \neg occupied(l), \neg at(r, l)$
- ▶ 经典表达 grounding 后与集合论表达等价，不过 grounding 结果可能指数增大
- ▶ 状态变量表达 (State-Variable Representation):
 - ▶ 状态为向量值，动作为函数映射。经典表达与状态变量方法在表达能力上是等价的

发展历史

- ▶ 自动规划最初受自动推理证明很大的影响，用 Situation Calculus 的方式对初始状态、目标状态和行动做公理化描述，使用归结定理证明来构造求解
- ▶ 然而，这种方式遇到框架问题的困难，从而引入对经典规划的描述问题，其目的之一就是为框架问题提供一个简单的解法：
 - ▶ STRIPS 假设：在效果中没提及的每个文字保持不变
- ▶ STRIPS 就是这方面早期的工作。这里介绍的经典表达和 STRIPS 有同样的表达能力（STRIPS 不同与自动定理证明不同的是，将解的搜索和对系统的逻辑描述分离了）
- ▶ ADL 权衡了一阶表达能力和推理的复杂性，之后扩展的 PDDL，为规划问题的标准描述语言
- ▶ PDDL (The Planning Domain Definition Language) 是由 IPC (International Planning Competition 两年一届) 定义的标准语言。已有一大批基于 PDDL 的通用规划器
- ▶ STRIPS 或 PDDL 方式与 Situation Calculus 方式都可以刻画动态系统，两者在规划、预测方面的效果相同，但如果需要更复杂的推理（如诊断），则只能使用 Situation Calculus

问题描述的计算复杂性

- ▶ 我们考虑在不同的问题描述下，经典规划存在解 (PLAN-EXT) 和存在固定长度解 (PLAN-LEN) 问题的复杂度。严格说来

PLAN-EXT = $\{P \mid P \text{ 是一个存在解的规划问题}\}$

PLAN-LEN = $\{(P, n) \mid P \text{ 是一个存在长度小于等于 } n \text{ 的解的规划问题}\}$

- ▶ 经典规划问题有如下判定性结论：
 - ▶ 在不允许函数符号的情况下，PLAN-EXT 和 PLAN-LEN 都是可判定的。
 - ▶ 在与允许函数符号的情况下，PLAN-EXT 是半可判定的，而 PLAN-LEN 是可判定的。
- ▶ 对于经典表达的经典规划问题，PLAN-EXT 是 EXPSpace-complete 的，PLAN-LEN 是 NEXPTIME-complete 的。在表达中允许条件效果，并不会增大复杂性

经典规划的复杂性

Kind of representation	How the operators are given	Allow negative effects?	Allow negative preconditions?	Complexity of PLAN-EXISTENCE	Complexity of PLAN-LENGTH
Classical rep.	In the input	Yes	Yes/no	EXPSPACE-complete	NEXPTIME-complete
		No	Yes	NEXPTIME-complete	NEXPTIME-complete
			No	EXPTIME-complete	NEXPTIME-complete
	In advance	Yes	Yes/no	PSPACE ^b	PSPACE ^b
			Yes	NP ^b	NP ^b
		No	No	P	NP ^b
Set-theoretic or ground classical rep.	In the input	Yes	Yes/no	PSPACE-complete	PSPACE-complete
		No	Yes	NP-complete	NP-complete
			No	P	NP-complete
	In advance	Yes/no	No ^a /no ^c	NLOGSPACE-complete	NP-complete
			Yes/no	Constant time	Constant time
		Yes/no	Yes/no	Constant time	Constant time
State-variable rep.	In the input	Yes ^d	Yes/no	EXPSPACE-complete	NEXPTIME-complete
	In advance	Yes ^d	Yes/no	PSPACE ^b	PSPACE ^b
Ground state-variable rep.	In the input	Yes ^d	Yes/no	PSPACE-complete	PSPACE-complete
	In advance	Yes ^d	Yes/no	Constant time	Constant time

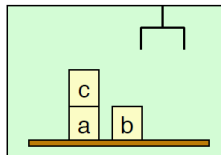
^a No operator has > 1 precondition.

^b With PSPACE- or NP-completeness for some sets of operators.

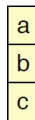
^c Each operator with > 1 precondition is the composition of other operators.

^d There is no way to keep the operators from having negative effects.

Example: blocks world



Initial state



goal

初始状态：

$on(A) \wedge on(C, A) \wedge on(B) \wedge clear(B) \wedge clear(C) \wedge handempty$

目标： $on(A, B) \wedge on(B, C)$

Example: blocks world (con't)

操作:

unstack(x,y)

Pre: $\text{on}(x,y)$, $\text{clear}(x)$, handempty

Eff: $\sim\text{on}(x,y)$, $\sim\text{clear}(x)$, $\sim\text{handempty}$,
 $\text{holding}(x)$, $\text{clear}(y)$

stack(x,y)

Pre: $\text{holding}(x)$, $\text{clear}(y)$

Eff: $\sim\text{holding}(x)$, $\sim\text{clear}(y)$,
 $\text{on}(x,y)$, $\text{clear}(x)$, handempty

pickup(x)

Pre: $\text{ontable}(x)$, $\text{clear}(x)$, handempty

Eff: $\sim\text{ontable}(x)$, $\sim\text{clear}(x)$, $\sim\text{handempty}$, $\text{holding}(x)$

putdown(x)

Pre: $\text{holding}(x)$

Eff: $\sim\text{holding}(x)$, $\text{ontable}(x)$, $\text{clear}(?x)$, handempty

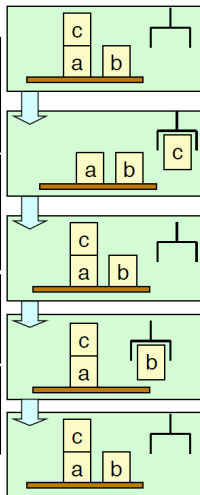


Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

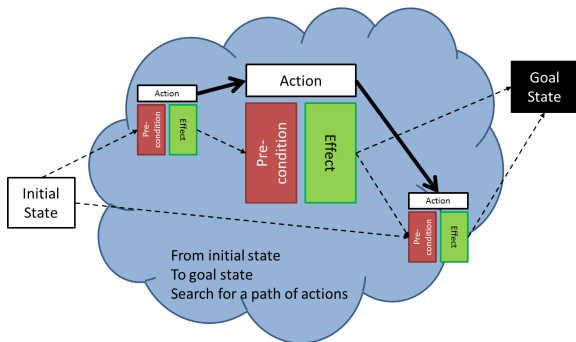
不确定规划

非确定规划

概率规划

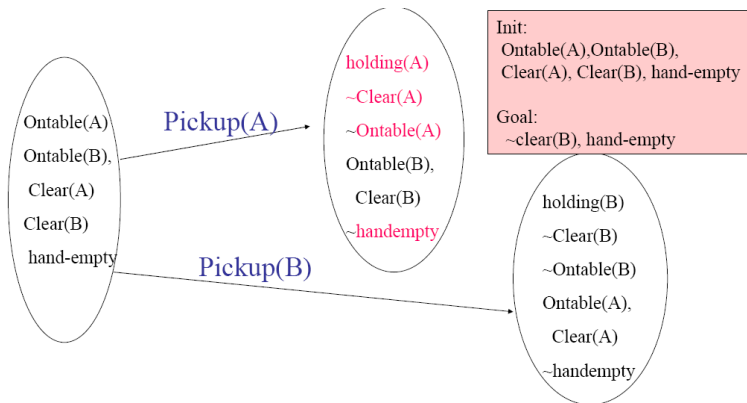
状态空间搜索方式

- ▶ 在状态转移图中搜索从初始状态到目标状态的一条路径
 - ▶ 前向搜索 (Forward Search): 从初始状态出发向前搜索
 - ▶ 后向搜索 (Backward Search): 从目标状态向后搜索
 - ▶ 启发式搜索: 利用启发式函数 (例如, 估计从状态到目标的距离) 进行前向或后向搜索



前进规划 (progression)

- ▶ 从初始状态开始，考虑所有可行的行动，进行深度或广度搜索， $\gamma(s, a)$
- ▶ 可靠并完全

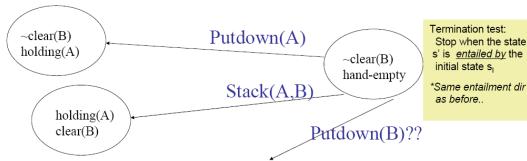


回归规划 (regression)

- ▶ 从目标出发，将当前目标还原为回归子目标， $\gamma^-(g, a)$
- ▶ 与 Forward 相比，一般有较小的分支数
- ▶ An action a is relevant for a goal g if
 - ▶ $g \cap effects^+(a) \neq \emptyset$
 - ▶ $g^+ \cap effects^-(a) = \emptyset$ and $g^- \cap effects^+(a) = \emptyset$

Init:
Ontable(A), Ontable(B),
Clear(A), Clear(B), hand-empty

Goal:
 \sim clear(B), hand-empty



SPRIPS 规划

- ▶ 类似 backward, 每次选择一个相关的 action, 但只将 action 的 precondition 作为下一步迭代的目标
- ▶ 当状态满足 precondition 后执行此动作, 并不再回溯
- ▶ STRIPS 是不完备的
- ▶ 算法过程: return a sequence of actions that transforms s into g
 1. Calculate the difference set $d = g - s$
 2. If d is empty, return an empty plan
 3. Choose action a whose add-list has most formulas contained in g
 4. $p' = STRIPS(s, \text{precondition of } a)$
 5. Compute the new state s' by applying p' and a to s
 6. $p = STRIPS(s', g)$
 7. return $p'; a; p$

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

规划空间搜索

- ▶ 状态空间是最直接的搜索方式，规划是状态转移图中的一条路径，规划求解自然也就是状态空间上的搜索问题
- ▶ 但很多时候，状态空间上的搜索是在以不同的顺序不断说明一组行动不可行
- ▶ 因此提出规划空间：节点为局部具体化的规划，弧为规划的求精操作（refinement operation）
- ▶ 求解操作基于极小承诺原则（least commitment principle）：
don't commit to orderings, instantiations, etc., until necessary
- ▶ PSP (Plan-Space Planning) 算法：找到 plan 的缺陷，选择一个缺陷，找到解决缺陷的所有方法，选择一个方法，按此方法对 plan 求精。PSP 算法是可靠完全的

偏序规划 (Partial-Order Planning)

- ▶ 规划问题：给定 s, g ，要找到行动序列 a_1, \dots, a_n 使得：
 1. a_1 在 s 上可执行
 2. 执行 a_i 后 a_{i+1} 可执行
 3. 执行 a_n 后 g 为真

前行，回归规划都是在过程中生成“部分计划”，满足 (2)，但只满足 (1) 或 (3) 之一；而偏序规划在过程中满足 (1) 和 (3)，可以不满足 (2)

- ▶ POP 算法：以伪计划 $P = (s, e)$ 为起点，在保持 P 为一个 POP 部分计划的前提下，不断向 P 中添加新的因果链，直至得到一个 POP 计划
- ▶ 因果链形如： $a_p \xrightarrow{Q} a_c$, $Q \in \text{eff}(a_p) \cap \text{pre}(a_c)$ 是 a_p 对 a_c 前提条件的贡献
- ▶ POP 算法，类似 PSP，不过对两类不同的缺陷（子目标和威胁）采取不同的处理方式。先处理子目标，再处理相应的威胁。

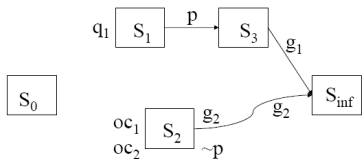
偏序规划

1. **Plan Selection:** Select a plan P from the search queue
2. **Flaw Selection:** Choose a flaw f (open cond or unsafe link)
3. **Flaw resolution:**
 - If f is an open condition,
choose an action S that achieves f
 - If f is an unsafe link,
choose promotion or demotion
 - Update P
 - Return NULL if no resolution exist
4. If there is no flaw left, *return* P

1. Initial plan:



2. Plan refinement (flaw selection and resolution):



状态空间和规划空间规划的比较

- ▶ 状态空间有限，而规划空间无限。但规划空间规划通常有较小的搜索空间
- ▶ 状态空间有显式的中间的状态，而规划空间中没有，没有明确的状态的概念。如果状态的概念清晰，可以有效利用领域知识和控制知识
- ▶ 规划空间，将对动作的选择和其顺序分离

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

新经典规划

- ▶ 经典规划虽然有很多成果，但在很长时间并没有在实际应用中广泛使用
- ▶ 新经典规划，还是基于原有经典规划的问题描述，但是采用新的技术来求解规划问题，这些方法主要有
 - ▶ 图规划技术 (更紧凑的搜索空间)
 - ▶ SAT、CSP、ILP 技术等

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

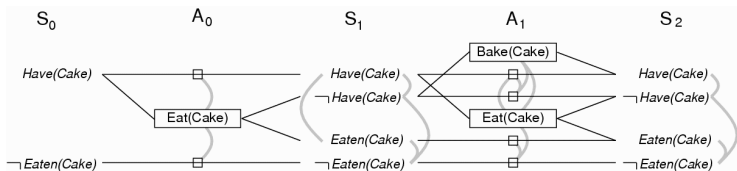
不确定规划

非确定规划

概率规划

规划图 (Planning Graph)

- ▶ Planning Graph 是对 possible plans 约束的描述。“If valid plan exists, it's a subgraph of the planning graph.”，并且可以在多项式时间构造
- ▶ 规划图是有向分层图：
 - ▶ 两种节点：Proposition P , Action A
 - ▶ 三种边：Precondition: $P \rightarrow A$, Add: $A \rightarrow P$, Delete: $A \rightarrow \neg P$
 - ▶ 两层：Proposition 和 Action
- ▶ Action level: 前提条件被上一层满足的 actions + no-op actions (for frame problem)



规划图方法

- ▶ 规划图方法是两个阶段交替执行：图扩展 (graph expansion) 阶段和解提取 (solution extraction) 阶段
 - ▶ 图扩展阶：正向扩展规划图直到目标状态的所有命题都出现为止
 - ▶ 解提取阶段：反向搜索规划图以求出规划解
- ▶ 规划图的构造过程：
 1. 用初始状态真的所有文字构造 P_1 层；
 2. 用前提被满足的行动构造 A_1 层；
 3. 再根据 A_1 层的 effects 构造 P_2 层 (包括 no-ops, 惯性)；
 4. ……
- ▶ 在构造过程中还要维护一组互斥关系 (Mutual Exclusion relations), 来删除不相容的命题和行动：
 - ▶ 两个行动 (或文字) 是互斥的, 如果不存在可行规划同时包含两者
 - ▶ 两个行动是互斥的, 如果:
 - ▶ 干扰 (Interference): 一个破坏另一个的效果或前提;
 - ▶ 需求竞争 (Competing needs): 两者的前提互斥。
 - ▶ 两个命题是互斥的, 如果同时获得两者所有方式都是互斥的。

规划图方法 (con't)

- ▶ 规划图的特点：
 - ▶ 命题和行动数目都单调递增;
 - ▶ 命题互斥和行动互斥关系单调递减;
 - ▶ 在有限步骤 k 以后, 所有层都相同。
- ▶ 规划图的构造时间是规划问题大小的多项式倍
- ▶ 规划解 (valid plan) 是规划图的一个子图, 满足如下条件:
 - ▶ 同一层的行动彼此不干扰;
 - ▶ 在规划中每个行动的前提都被满足;
 - ▶ 目标被满足。
- ▶ 规划图技术的算法：
 1. 先构造规划图 (PG) 直到所有的目标都可达且不存在互斥 (如果 PG 构造失败, 则规划失败);
 2. 在 PG 中搜索规划解;
 3. 如果没有找到, 则在 PG 中增加一层, 重新再试一次, 直到找到结果为止。

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

Planning as {SAT, CSP, ILP, ...}

- ▶ Grounding 以后的规划问题复杂度是 PSPACE-complete, 因为规划本身的长度有可能是指数的
- ▶ 只在有限长度 k 内计算 plan, 则复杂度为 NP-complete
- ▶ 所以将规划问题翻译为其他解决 NP-hard 的经典问题, Planning as:

- ▶ SAT: 命题公式可满足性问题 (Propositional Satisfiability Problem)
 - ▶ Situation Calculus 方式。将规划问题用命题逻辑方式表达, 使得每个 model 对应一个 plan
 - ▶ 结合规划图和 SAT。先构造 k 层的规划图, 将此图转化为 SAT 问题, 再用 SAT 求解器来计算里面的 plan。比较成功, 代表有 Black-Box (1998 年规划竞赛优秀者之一), SatPlan (2004, 2006 年冠军)
- ▶ CSP: 约束满足问题 (Constraint Satisfaction Problem)
 - ▶ 将规划问题翻译为一个 CSP 问题, 用其求解器求解。也可以通过规划图来翻译 GP-CSP。通常 CSP 能给出一个更紧凑的表达
 - ▶ 将 CSP 技术应用到规划中
- ▶ 其它: ILP (Integer Linear Programming), 模型检查 (Model Checking), ASP (Answer Set Programming)

小结

- ▶ 规划的核心问题是表示和求解（搜索）
- ▶ 经典规划用命题（或一阶）表示状态和行动
- ▶ 状态空间规划一般分为前进（progression）和回归（regression）规划
- ▶ 偏序规划为规划空间规划
- ▶ 规划图可以多项式时间构造出来，规划解是规划图的一个子图。规划图算法就是反复扩展规划图和尝试提取解的过程
- ▶ 一般规划问题（Grounding 以后）如果不限定规划长度，则计算复杂性为 PSPACE-complete；若限定长度，则为 NP-complete，可以等价翻译为其他解决 NP-hard 的经典问题，如 SAT, CSP, ILP, 等

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

分层任务网络规划 (Hierarchical Task Network (HTN) Planning)

- ▶ 分层任务网规划和经典规划类似，但增加一个方法集合 (method)，告诉系统，如何将一类任务分解为更小的子任务（可能有偏序约束），规划过程就是递归的将那些非原子任务分解到原子任务。
- ▶ HTN 规划基本过程：
 1. 输入规划问题 P ;
 2. 如果 P 只包含原子任务，则解决冲突并返回规划结果；如果冲突无法解决，则返回失败；
 3. 在 P 中选择一个非原子任务 t ;
 4. 选择 t 的一个扩展，并用它替换 t ;
 5. 查找 P 中任务间的交互，给出解决方案，从中选择一个执行；
 6. 回到 2 继续执行。
- ▶ 因为 HTN 可以方便的根据领域知识加入如何分解 task 的方法，HTN 应用很广，而且效果很好，在规划大赛中总是前几名

HTN Examples

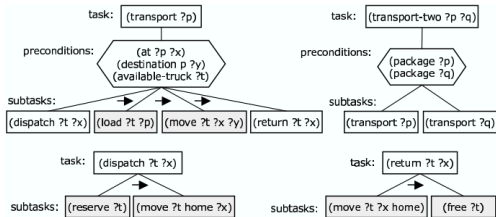
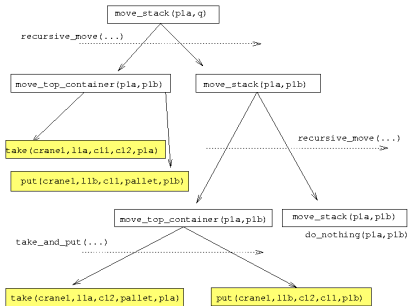
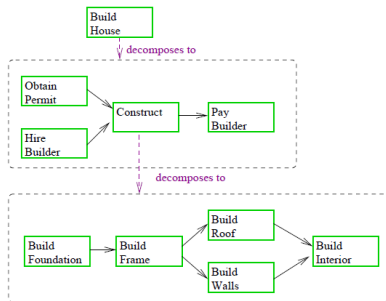


Table of Contents

自动规划概述

经典规划 (Classical Planning)

规划问题

状态空间规划 (State-Space Planning)

规划空间规划 (Plan-Space Planning)

新经典规划 (Neoclassical Planning)

规划图技术 (Planning-Graph Techniques)

Planning as {SAT, CSP, ILP, ...}

经典规划的扩展

分层任务网规划

不确定规划

非确定规划

概率规划

不确定性领域中进行规划 (Planning under Uncertainty)

- ▶ 经典规划有如下三个限制：
 - ▶ 确定性 (determinism), 但现实情况更多是不确定的：
 - ▶ 由于信息的不完全, 我们不能确定具体在哪个状态。
Uncertainty about the state of the world.
 - ▶ 行动的结果可能是不确定的。Uncertain effects for actions.
 - ▶ 在执行决策的过程中, 外部事件可能改换世界。External events.
 - ▶ 完全可观察性 (full observability), 但实际情况更多的是部分可见
 - ▶ 状态目标 (Attainment goals), 由于动作结果的不确定和可能的执行失败, 目标也相应扩展的更灵活, 主要有两种：
 - ▶ 效用函数, 规划求解的任务是使得效用函数的值最大化
 - ▶ 用时态逻辑公式表达目标, 规划求解的任务是使得, 规划中行动令此时态公式为真

Planning under Uncertainty (con't)

- ▶ 由于不确定性，规划的执行可能对应于多条不同的执行路径，需要规划算法能高效的分析所有动作各种可能的执行结果。可以有如下方式：
 - ▶ 重规划: 先假设没有意外发生，从而离线规划出一个规划结果；在执行时监控执行过程；一旦出现意外则重现规划，此时重规划可以选择重新规划也可以选择修补现有规划
 - ▶ 条件规划: 规划时考虑所有可能发生的情况，在规划策略中处理每种可能的事件
- ▶ 两种方式，前者没办法未雨绸缪，后者又考虑过甚（不可能真正处理完所有的可能），需要平衡
- ▶ 用概率来表达 uncertainty 能实现较好的平衡，Probabilistic planning: Plan ahead for likely contingencies that may need steps taken before they occur.

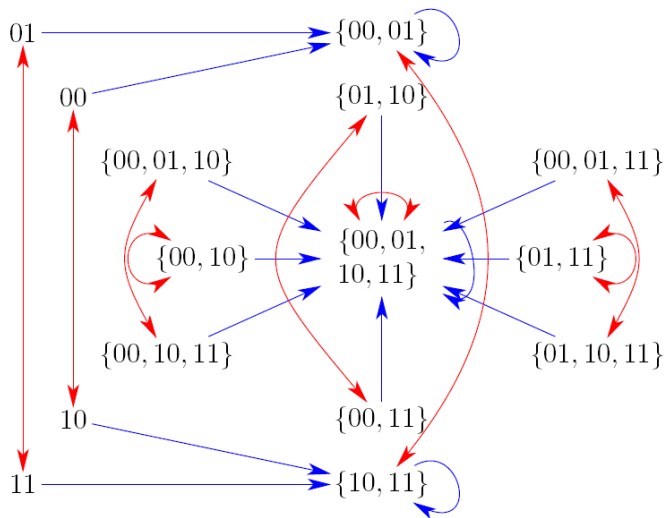
非确定规划 (Nondeterministic Planning)

- ▶ 在命题逻辑下处理非确定规划，根据规划结果的质量有如下分类：
 - ▶ 弱规划 (Weak plan): 其中某种执行方式可能达到目标
 - ▶ 强循环规划 (Strong cyclic plan): 每种执行方式最终都有可能达到终止状态，且都为目标状态
 - ▶ 强规划 (Strong plan): 是强循环规划，且执行方式不存在循环
 - ▶ 符合规划 (Conformant plan): 是强规划且没有观察
- ▶ 在不同的要求下，非确定规划问题又可以分类为：
 - ▶ 具备完全观察的强非确定规划 (Strong nondeterministic planning with full observability)
 - ▶ 由于具备完全观察，所以可以用无记忆策略 (memoryless strategy $\pi : S \rightarrow O$).
 - ▶ 具体算法有两种：1. 看作是 AND-OR search. 2. Dynamic programming (backward).
 - ▶ 符合规划 (Conformant planning): 没有观察的规划，EXPSPACE-complete
 - ▶ plans are sequences of actions
 - ▶ 不知道具体的 state，用 belief state。具体的算法也只在 belief state 上处理。但 belief state 的空间非常大

非确定规划 (Nondeterministic Planning)

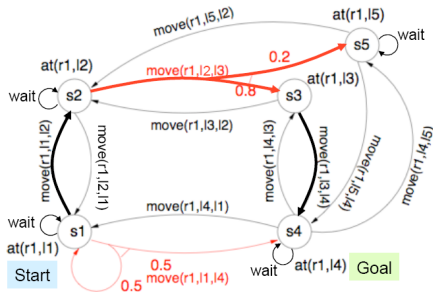
- ▶ 具备部分观察的非确定规划 (Nondeterministic planning with partial observability), 有三种方法:
 - ▶ 将部分观察问题转化为在信念状态上完全可观察的规划问题
 - ▶ 在与或树 (AND/OR trees) 做前向搜索
 - ▶ 从目标信念状态出发, 采用动态规划形式的后向搜索来构造可解的信念状态
- ▶ 非确定性规划相关的一些复杂性结论:
 - ▶ 对于实例化以后采用经典表示, 具备完全观察的非确定规划, 强 (循环) 规划结果是否存在的计算复杂性是 EXP-complete
 - ▶ 符合规划的计算复杂性是 EXPSPACE-complete
 - ▶ 部分可观察规划的计算复杂性是 2-EXP-complete

The Belief Space: An Example

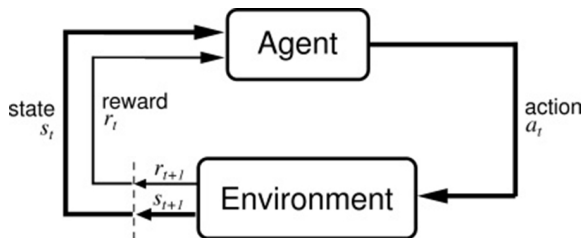


概率规划 (Probabilistic Planning)

- ▶ 如果将不确定信息用概率的方式表达出来，就是 Probabilistic planning
- ▶ PDDL 也已扩展出 PPDDL (Probabilistic PDDL) 用概率表达不确定性 (2004 年)
- ▶ 可以直接使用 MDP/POMDP 方法，也可以使用其他方法，如 Planning based on Markov Decision Processes
 - ▶ 将规划问题表示称为一个优化问题来解决



马尔可夫决策过程 (MDP) 基本框架



- ▶ MDPs 立足于 agent 与环境的直接交互
- ▶ 只考虑离散时间，假设 agent 与环境的交互过程可分解为一系列“阶段”，每个阶段由“感知-决策-行动”构成
- ▶ 已知环境模型

MDP 模型

- ▶ MDP 模型是一个四元组 (S, A, T, R) ，其中
 - ▶ S 是一个有限集，其中每个元素 $s \in S$ 代表一个状态
 - ▶ A 是一个有限集，其中每个元素 $a \in A$ 代表一个行动
 - ▶ $T: S \times A \rightarrow \Pi(S)$ 称为状态转移函数，将每一对“状态 行动”映射为 S 上的一个概率分布，用记号 $T(s, a, s')$ 表示在状态 s 上执行 a 达到 s' 的概率
 - ▶ $R: S \times A \rightarrow \mathcal{R}$ 称为回报 (reward) 函数， $R(s, a)$ 表示在 s 上执行行动 a 所得到的即时回报 (直接回报)
- ▶ 如何规定一个 MDP 模型是一个具体应用问题，不是 MDPs 本身研究的内容
- ▶ MDPs 研究的主题是，给定一个 MDP 模型，如何求最优策略，即期望回报最大的策略

State-Value Function and Action-Value Function

- ▶ 回报 (return): 回报 G_t 是从时刻 t 开始的总折扣奖励:

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots = \sum_{k=1}^{\infty} \gamma^k R_{t+k+1}$$

- ▶ 状态值函数 (state-value function): 状态值函数 $V_{\pi}(s)$ 是从状态 s 出发, 按照策略 π 采取行动得到的期望回报:

$$\begin{aligned} V_{\pi}(s) &= E_{\pi}(G_t \mid S_t = s) \\ &= E_{\pi}(R_{t+1} + \gamma G_{t+1} \mid S_t = s) \\ &= E_{\pi}(R_{t+1} + \gamma V_{\pi}(S_{t+1}) \mid S_t = s) \end{aligned}$$

- ▶ 行动值函数 (action-value function, action-state-value function): 行为值函数 $Q_{\pi}(s, a)$ 是从状态 s 出发, 采取行动 a 后, 然后按照策略 π 采取行动得到的期望回报:

$$\begin{aligned} Q_{\pi}(s, a) &= E_{\pi}(G_t \mid S_t = s, A_t = a) \\ &= E_{\pi}(R_{t+1} + \gamma Q_{\pi}(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a) \end{aligned}$$

贝尔曼等式

- 利用贝尔曼等式，可以用递归的方式来计算状态的**期望收益值**：

$$V^*(s) = \max_a Q^*(s, a)$$

$$Q^*(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$$

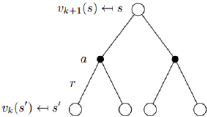
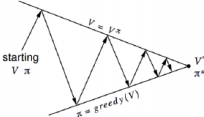
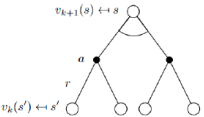
$$V^*(s) = \max_a \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V^*(s')]$$

最优值函数的计算



Richard Bellman
(1920 - 1984)

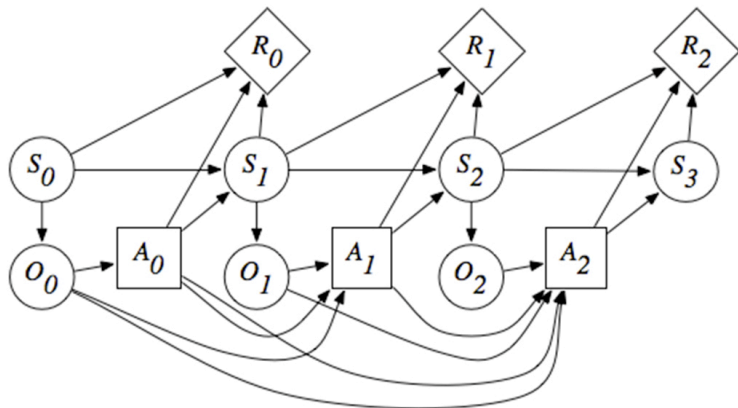
Dynamic Programming

Algorithm	<i>Iterative Policy Evaluation</i> 	<i>Policy Iteration</i> 	<i>Value Iteration</i> 
Bellman Equation	Bellman Expectation Equation	Bellman Expectation Equation Policy Iteration + Greedy Policy Improvement	Bellman Optimality Equation
Problem	Prediction	Control	Control

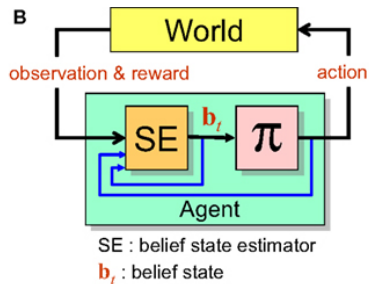
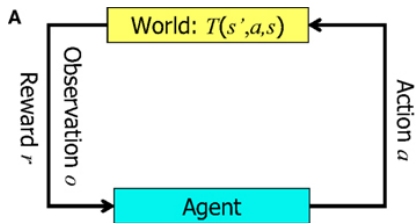
POMDPs 基本定义

- ▶ MDPs 刻画了行动的不确定性：事先不知道，事后知道；观察是确定的，知道行动的效果
- ▶ 一般情况下，行动和观察都是不确定的
- ▶ 一个 POMDP 模型是一个六元组 (S, A, T, R, Ω, O) ，其中 S, A, T, R 的定义与 MDP 模型相同； Ω 是一个有限集，其中元素称为“观察”； $O: S \times A \rightarrow \Pi(\Omega)$ 称为观察函数，有时记为 $O(s', a, o)$ ， a 代表执行的行动， s' 是 a 达到的结果状态， o 在上述条件下出现的观察， $O(s', a, o)$ 是执行 a 达到 s' 观察到 o 的概率 $P_r(o | s', a)$
- ▶ 在 POMDP 问题中，“环境”被看成一个“黑箱”，一个状态是黑箱某个时期的内部情况，观察是这个黑箱的“输出”

POMDPs



MDP vs. POMDP



POMDP to MDP

- ▶ B 在 POMDPs 中相当于 MDPs 中 S 的作用, 设法将一个 POMDP 问题转化为一个变形的 MDP 问题, 使得相应的 MDP 模型为 (B, A, τ, ρ) , $\tau: B \times A \rightarrow \Pi(B)$, $\rho: B \times A \rightarrow \mathcal{R}$

$$\tau(b, a, b') =_{df} P_r(b' | b, a) = \sum_o P_r(b' | a, b, o) P_r(o | a, b)$$

$$P_r(b' | a, b, o) = \begin{cases} 1, & \text{if } SE(b, a, o) = b' \\ 0, & \text{otherwise} \end{cases}$$

$$\rho(b, a) =_{df} \sum_s b(s) R(s, a)$$

- ▶ 由于 B 是无穷的, 很难直接用值迭代算法求解

机器人上的规划 (Planning in Robotics)

- ▶ 机器人涉及的规划问题：
 - ▶ Task planning: Classical Planning, MDP/POMDP
 - ▶ Path and motion planning, 包括: Navigation planning, Manipulation planning
 - ▶ 本质上是在参数空间中, 找一条路径
 - ▶ Perception planning
 - ▶ Reinforcement Learning
- ▶ 经典规划框架对于机器人应用来说: too hard and too simple
 - ▶ too hard: it is intractable;
 - ▶ too simple: action sequences are not adequate as a representation of a real robot's program.

小结

- ▶ 分层任务网络 (HTN) 规划可以方便的根据领域知识加入如何分解任务的方法，从而可以解决实际问题
- ▶ 经典规划算法假设有完备的和正确的信息、确定性的和完全可观察的环境，实际问题通常违反这些假设
- ▶ MDP/POMDP 为不确定规划问题采用概率手段给出了统一的数学描述框架
- ▶ 经典规划框架对机器人应用来说：Too hard and too simple