基于线性时序逻辑的智能体不确定行为规划

陈仲瑶, 方浩*

北京理工大学自动化学院, 北京 100081

* E-mail: fangh@bit.edu.cn

收稿日期: 2019-08-19; 接受日期: 2019-10-14; 网络版发表日期: 2020-04-26

深圳机器人基础研究中心项目国家自然科学基金(编号: U1913602)、重大国际(地区)合作研究项目、国家自然科学基金(批准号: 61720106011, 61873033, 61903035)、鹏城实验室和智能机器人与系统高精尖创新中心资助

摘要 随着人们对智能体需求的提高, 智能体的活动不再局限于简单环境与单一任务, 面向复杂的应用场景, 智能体需要具备自主完成决策与执行的能力. 本文研究了面向线性时序逻辑描述下的复杂任务智能体的不确定行为规划问题, 同时考虑任务成功率与任务执行成本, 这里不确定性因素包括智能体行为与环境属性, 任务描述由软、硬约束两部分表达. 文中应用形式化方法中模型检测的思想确定智能体行为选择策略, 其中应用马尔可夫决策过程构建智能体个体与集群模型, 应用双层自动机模型构建任务模型, 设计智能体-任务网络模型表征约束条件并通过一耦合线性规划完成策略求解, 并通过数值仿真测试对上述方法进行了验证. 结果表明含软、硬约束的复杂任务约束可被满足, 求解所得最优策略使智能体按约束强度完成任务, 且可通过调节惩罚因子控制网络模型的松弛程度调整生成的控制策略.

关键词 线性时序逻辑, 不确定行为规划, 双层自动机, 软、硬任务约束

1 引言

近年来, 很多研究关注如何解决智能体在考虑环境与自身约束情况下执行复杂任务的问题, 如文献[1]研究肠道环境智能体控制策略生成, 文献[2]中提到未知动态环境中多智能体协作算法设计, 其中面向复杂任务下智能体行为规划的研究越来越广泛, 很多研究中智能体的任务由线性时序逻辑(LTL)表征, 借任务公式的形式形象刻画智能体任务中对时间与逻辑的约束, 使得原本复杂的任务可以被准确形象地描述并转换为数学模型参与规划. 智能体行为规划较为成熟的方法框架是通过对智能体环境、执行任务的抽象建

模, 利用模型检测的思想求解智能体行为策略并由底层控制器实施, 形成高层规划与底层执行相结合的混杂控制.

面向时序逻辑描述的复杂任务, 智能体行为规划研究大多集中在智能体、环境模型确定的情况下, 如智能体LTL任务重建^[3]、时序逻辑与人机协同的结合^[4]等. 当智能体行为产生随机结果, 环境属性具有时变性、不确定性时, 在确定型环境下生成固定路径的算法可能会失效, 智能体不能通过固定确定的策略在满足任务要求的同时能够适应环境, 在先验知识不足的环境中或是智能体执行精度不高的情况均需要考虑不确定性因素. 这方面研究多用马尔可夫决策过程

引用格式: 陈仲瑶, 方浩. 基于线性时序逻辑的智能体不确定行为规划. 中国科学: 技术科学, 2020, 50: 516–525

Chen Z Y, Fang H. Probabilistic action planning based on linear temporal logic (in Chinese). Sci Sin Tech, 2020, 50: 516–525, doi: 10.1360/SST-2019-0292

(MDP)表征含不确定性的状态转移. 近年来, 对于马尔可夫决策过程的控制策略生成成为了研究的热点^[5~7], Lahijanani等人^[8]实现了不确定性计算逻辑树在马尔可夫链上的抽象验证工作, 在另外的研究中, 考虑了面向带限制的LTL公式最大化完成任务的问题^[9], 对于一个确定性与带有随机性的平台, 通过马尔可夫决策过程建模的智能体系统之间的配合问题得到了解决^[10], 在逻辑语言与控制对象上进行了扩展. 在含不确定因素的智能体规划研究中, 规划目标往往是最大化任务成功率或是保证某一指定的失败风险^[11]. 很多研究应用了不确定性模型检测方法^[11,12], 并设计出相关模型检测工具PRISM^[13], 但其在成本优化上略显不足, 也有研究通过值迭代的方法, 确定终止条件求解成本最优策略^[14], 但文章中忽视了对任务风险的考虑. 有研究通过线性规划的方法直接求解最优策略, 同时考虑到任务风险与成本的问题^[15], 但任务形式单一.

从任务层面上讲, 任务内容是可以按重要程度划分层级的. 很多研究^[15,16]都对任务间重要程度的差异及相应的智能体规划做分析, 任务公式可划分为不允许存在失败风险的硬约束部分与允许存在失败风险的软约束部分, 通过对任务描述程度的提高, 智能体需要完成的任务更详细且具有弹性, 相比使用LTL公式描述的“要求一定完成”的单一任务, 这种描述方式更具有应用意义, 如牺牲任务完成度以提高智能体效率.

针对以上情况, 本文提出面向线性时序逻辑任务的不确定智能体行为规划方法, 其中任务是含有软、硬约束描述的. 沿用模型检测的思想, 在求解最优控制策略前, 建立适当的环境模型与任务模型, 通过耦合线性规划求解最优策略, 系统满足任务约束, 实现任务执行成本的优化. 本文创新之处在于: 一是针对复杂任务描述形式构建任务模型, 建立双层网络模型表征任务执行情况; 二是面向任务软约束提出任务网络的松弛方法, 软、硬约束两部分对应任务可以按其重要性更合理地执行; 三是通过对智能体-任务网络模型做行为约束处理保证硬约束部分任务的完成.

论文结构安排如下: 第2节给出论文中参考应用的马尔可夫决策过程、线性时序逻辑与自动机模型的基本理论, 第3节完成由双层自动机模型对智能体任务的建模工作, 第4节提出智能体环境模型的构建以及智能体行为规划的方法框架, 第5节给出相应的仿真实验与结果, 第6节对论文内容加以总结并给出下一阶段的研

究发展方向.

2 预备知识

本节内容将围绕论文中所涉及的数学模型与表达做简要介绍.

2.1 有限状态转移模型

定义1(马尔可夫决策过程) 一个马尔可夫决策过程可以由一个6元组给出:

$$M \triangleq (X, U, D, p_D, c_D, x_0), \quad (1)$$

其中 X 是有限状态集合; $U(x)$ 是 X 中单个状态对应的可行行为集; $D = \{(x, u) \mid x \in X, u \in U(x)\}$, 是所有可能的状态行为对的集合; $p_D : X \times U \times X \rightarrow [0, 1]$ 描述状态间转移概率, 满足 $\sum_{x' \in X} p_D(x, u, x') = 1, \forall (x, u) \in D$; $c_D : D \rightarrow \mathbb{R}^{>0}$ 是某状态下执行行为的代价函数; x_0 是模型中的初始状态. 定义 $\text{Post}(x, u) = \{x' \in X \mid p_D(x, u, x') > 0\}$, $\forall (x, u) \in D$, $\text{Post}(x, u)$ 表示 D 中状态行为组合对应下一时刻状态转移的所有可能.

定义2(终止集^[11]) 终止集(End Component, EC)是马尔可夫决策过程模型中的状态子集. 设一MDPM, 它的子MDP图 (S, A) , 其中 $S \subseteq X$, $A : S \rightarrow 2^U$, 满足条件

$$\begin{aligned} S &\neq \emptyset, \emptyset \neq A(s) \subseteq U(s), \forall s \in S, \\ \text{Post}(s, u) &\subseteq S, \forall s \in S, \forall u \in A(s), \end{aligned} \quad (2)$$

即在子MDP中每个状态的可行行为集会缩小, 任何转移不能使其中状态脱离子MDP的状态集. 若由一个子MDP图 (S, A) 生成的有向图 $G_{(S, A)}$ 是强连通的, 则称 (S, A) 为 M 上的一个终止集. 对一个终止集 (S, A) , 当不存在其他的终止集 (S', A') 使得 $(S, A) \neq (S', A')$, $S \subset S'$, $A(s) \subset A'(s), \forall s \in S$, 则称它为最大终止集(Maximal End Component, MEC). 可接受MEC (Accepting Maximal End Component, AMEC)是指满足接受条件的终止集, 这与任务对应自动机模型相关.

2.2 线性时序逻辑与自动机模型

时序逻辑是通过原子命题的形式表示环境信息、

智能体信息, 经过一系列语法规则定义下的布尔符号与时序符号连接组成的用以描述系统规范的描述方式. 在本文中, 智能体的任务通过线性时序逻辑表达.

定义3(线性时序逻辑) 线性时序逻辑公式由原子命题集 AP 与布尔运算符、时序运算符组成, 公式形式为

$$\varphi \triangleq \top \mid p \mid \varphi_1 \wedge \varphi_2 \mid \neg \varphi \mid \varphi_1 \cup \varphi_2, \quad (3)$$

这里 $\top \triangleq \text{True}$, $p \in AP$ 是原子命题集中的命题, $\neg \triangleq \neg \top$. LTL公式中包含的时序运算符包括 $\circ$ (下一步), $\cup$ (直到), $\square$ (总是), $\diamond$ (最终), $\Rightarrow$ (指示符), 不同运算符的意义如图1所示.

图1中列举了满足不同任务要求的状态序列的个例, 深色表示该时刻满足任务 φ , 浅色表示该时刻满足任务 φ' . 其中 $\circ\varphi$ 表示下一时刻满足子式(子任务) φ , $\varphi \cup \varphi'$ 表示 φ 一直满足直至 φ' 满足, $\square\varphi$ 表示 φ 总是被满足, $\diamond\varphi$ 表示 φ 最终会满足, $\varphi' \Rightarrow \varphi$ 表示 φ' 满足后 φ 立刻满足.

LTL公式可表示为单词无穷序列的集合, 每个单词是位长为 $|AP|$ 的二进制数, 用于表示某状态下原子命题集中各个命题的满足情况. 无穷单词序列表示了状态序列的记录, 定义 $\text{Word}(\varphi) = \{w_1, w_2, \dots\}$, 其中 $w_i = w_i(1)w_i(2)\dots$, $i = 1, 2, \dots$, $w_i(j) \in 2^{AP}$, $j = 1, 2, \dots$, $\text{Word}(\varphi)$ 是满足LTL公式描述规范的语句的集合.

自动机模型一般是有限状态转移图模型, 具有检验语句与生成序列两个功能. 将符合语法要求的语句输入自动机, 可使自动机中状态发生迁移, 通过判断状态序列是否与自动机可接受状态相交得出语句与相应规范的符合情况; 或由自动机模型生成可行序列. 自动机模型可由LTL公式转换得到, 借助该模型可求得满足LTL公式描述下规范的所有状态序列^[12]. 本文应用确定Rabin自动机(Deterministic Rabin Autometa, DRA)为智能体任务建模.

定义4(确定Rabin自动机^[11]) 给定任务 φ , 确定Rabin自动机由一个5元组给出:

$$A_\varphi = (Q, 2^{AP}, \delta, q_0, Acc_A), \quad (4)$$

其中 Q 是模型有限状态集, AP 是任务 φ 规定下的原子命题集, 2^{AP} 是字母表, 包含 AP 下的单词, δ 是状态转移

图1 时序符号意义

Figure1 Interpretation of temporal symbols.

组合与转移条件, q_0 是初始状态, $Acc_A \subseteq 2^Q \times 2^Q$ 表示可接受状态对的集合, 可表示为 $Acc_A = \{(H_A^1, I_A^1), (H_A^2, I_A^2), \dots, (H_A^N, I_A^N)\}$, 这里 $H_A^i, I_A^i \subseteq Q, i = 1, 2, \dots, N$. 在确定Rabin自动机中, 一个状态序列 $q_0 q_1 \dots$ 是可接受的, 当 $\exists (H_A^i, I_A^i) \in Acc_A$, 使得 $\exists n \geq 0, \forall m \geq n, q_m \notin H_A^i$, 且 $\exists n \geq 0, q_n \in I_A^i$, 其中 $\exists$ 表示“存在无穷多个”. 任务模型中状态序列满足上述接受条件表示该序列满足由LTL公式所表示的任务约束.

3 智能体模型与任务模型构建

3.1 智能体模型

智能体运行环境建立在栅格化的地图上, 栅格中心代表一个离散的地理位置, 通过坐标表示. 智能体与环境上的不确定性在这里考虑两点: 环境属性上的不确定性与智能体行为造成状态转移的不确定性. 环境属性上的不确定性是指每个地理位置上的环境描述是具有随机性的, 例如(1, 1)位置有障碍的概率是0.9, 无障碍的概率是0.1. 智能体行为上的不确定性如图2所示, 假设智能体可执行前进行为, 但行为造成的转移结果不总是与指令要求一致, 存在小概率出现状态错误转移.

这里应用MDP模型表征智能体状态转移以及系

图 2 智能体行为不确定性

Figure 2 Uncertainty of agent action.

统存在的不确定性^[15]:

$$M = (X, U, D, p_D, (x_0, l_0), AP, L, p_L, c_D), \quad (5)$$

其中 X 是有限状态集合, 描述智能体位置信息; $U(x)$ 是 X 中单个状态对应的可行行为集; $D = \{(x, u) | x \in X, u \in U(x)\}$, 是所有可能的状态行为对的集合; $p_D : X \times U \times X \rightarrow [0, 1]$, 描述状态间转移概率, 满足 $\sum_{x' \in X} p_D(x, u, x') = 1, \forall (x, u) \in D$, $c_D : D \rightarrow \mathbb{R}^{>0}$ 是某状态下执行行为的代价函数; AP 表示环境属性对应的原子命题集; $L : X \rightarrow 2^{2^{AP}}$ 是某状态下所有可能的环境属性描述的集合; $p_L : X \times 2^{2^{AP}} \rightarrow [0, 1]$ 是某状态下不同环境属性描述的概率分布, 满足 $p_L : X \times 2^{2^{AP}} \rightarrow [0, 1]$; (x_0, l_0) 是初始组合状态, 其中 $l_0 \in L(x_0)$.

这里假设, 只有智能体状态为 x_t 时, 其位置的环境描述标签 l_t 才会观测到而被确定. 行为 u_t 的选择由控制策略决定. 控制策略的序列表示为 $\mu = \mu_0 \mu_1 \dots$, 其中 μ_t 是 t 时刻的控制策略, 这里应用固定随机的控制策略, 即 $\mu_t = \mu$, 且有 $\mu : X \times U \rightarrow [0, 1]$. 完整的运行序列表示为

$$R_T = (x_0 l_0 u_0)(x_1 l_1 u_1) \cdots (x_T l_T u_T). \quad (6)$$

如图3所示, 这里设 $T = \infty$, 即运行序列为无限长, 智能体需要持续满足给定任务的需求.

3.2 任务模型

对于复杂任务的描述, 任务公式由两个子公式给出, 分别描述硬约束部分与软约束部分, 两个公式形成组合任务, 体现两任务间重要程度的差异. 硬约束

图 3 运行序列

Figure 3 State sequence.

任务要求一定完成, 而软约束任务的完成可以存在风险, 或出现任务失败, 任务进程应以硬约束为主, 软约束任务的执行过程不能影响硬约束任务要求.

组合任务的形式为

$$\bar{\varphi} = \varphi_{\text{hard}} \wedge \varphi_{\text{soft}}, \quad (7)$$

其中, φ_{hard} 和 φ_{soft} 分别表示硬约束与软约束任务公式, 由上述描述方式, 两部分任务都需要满足, 通过转化得到对应确定Rabin自动机.

$$A_{\text{hard}} = (Q, 2^{AP}, \delta, q_0, Acc_A), \quad (8)$$

$$A_{\text{soft}} = (Q', 2^{AP'}, \delta', q'_0, Acc'_A). \quad (9)$$

在规划过程中, 由于两任务在重要程度上有区别, 不会将整体任务公式直接转化为自动机模型, 这里采用构建双层自动机^[15]的思路, 组合任务 $\bar{\varphi}$ 对应的双层 $\text{DRA}_{\bar{\varphi}}$ 定义如下:

$$\overline{A}_{\bar{\varphi}} = (\overline{Q}, 2^{\overline{AP}}, \overline{\delta}, \overline{q}_0, \overline{Acc}_A), \quad (10)$$

其中 $\overline{Q} = Q \times Q' \times \{1, 2\}$ 是自动机状态集, 最后一位为标志位; $\overline{AP} = AP \cup AP'$ 原子命题集为独立任务对应命题集的并集, 这规定了状态转移单词在语法上的合法格式; $\overline{q}_0 = (q_0, q'_0, 1)$ 是自动机任务状态初始状态; 这里自动机的接受状态定义方式^[19]为

$$Acc_A = \{(H_1^1, I_1^1), (H_2^1, I_2^1), \dots, (H_{N_1}^1, I_{N_1}^1)\}, \quad (11)$$

$$Acc'_A = \{(H_1^2, I_1^2), (H_2^2, I_2^2), \dots, (H_{N_1}^2, I_{N_1}^2)\}, \quad (12)$$

$$\overline{Acc_A} = \left\{ (\overline{H_1}, \overline{I_1}), \dots, (\overline{H_{N_1}}, \overline{I_{N_1}}) \right\} \cup \left\{ (\overline{H_1}, \overline{I_1}), \dots, (\overline{H_{N_2}}, \overline{I_{N_2}}) \right\}, \quad (13)$$

这里 $\overline{H_i} = H_i^1 \times Q' \times \{1, 2\}$, $\overline{I_i} = I_i^1 \times Q' \times \{1\}$, $i = 1, 2, \dots, N_1$, $\overline{H_j} = Q \times H_j^2 \times \{1, 2\}$, $\overline{I_j} = Q \times I_j^2 \times \{2\}$, $j = 1, 2, \dots, N_2$, $\overline{Acc_A}$ 中第一个集合是硬约束任务对应接受条件, 第二个集合是软约束任务对应接受条件. 如式(11)和(12)所示, 在组合状态下, 接受状态集合中的元素形式也需要改变为二元结构, 如上述定义, 当确认单一任务的接受条件时, 通过遍历另一任务状态得到全部接受状态对, 此外在状态描述上添加代表层级信息的标志位取值; δ 描述自动机中状态间转移关系, 存在 $(q'_1, q'_2, c') = \delta((q_1, q_2, c), l)$, 当满足: (i) 对应 A_{hard} 与 A_{soft} 中, l 分别是使状态由 q_1 至 q'_1 与 q_2 至 q'_2 的合法单词; (ii) 当 $q_1 \in I_i$, $i = 1, 2, \dots, N_1$ 且 $c = 1$ 时, $c' = 2$, 当 $q_2 \in I_j$, $j = 1, 2, \dots, N_2$ 且 $c = 2$ 时, $c' = 1$, 其余情况 $c' = c$.

将自动机拓展为双层结构, 如图4所示, 由标志位标明层数, 除层间转移外, 每层的状态与转换关系都是相同的, 层内的转换严格按照单词的要求. 从接受状态的定义上可知, 硬约束任务的接受状态被放置在第一层, 软约束任务的接受状态被放置在第二层, 双层间的转换条件是只有当系统所在层级对应的任务完成时可转换至另一层, 具体解释为, 初始状态在第一层, 系统状态由初始状态出发, 只有完成硬约束要求, 才能进入第二层, 在第二层中只有完成软约束要求, 才能返回第一层. 总任务的完成过程就是交替完成硬约束任务与软约束任务, 从自动机的角度上看就是在双层结构间做循环转移, 之后的行为规划算法也是在此基础上进行的.

图4 双层DRA模型
Figure 4 Double-layer DRA model.

3.3 群体智能体模型构建

在多智能体系统中, 智能体模型同样通过一马尔可夫决策过程 M_c 表征, 形式与式(5)相似, 其中状态、环境属性以及行为为多元组合状态, 并列表示每个个体当前时刻的信息, 如由 N 个智能体组成系统在 t 时刻的状态为 $\mathbf{x}_t = (x_t^1, x_t^2, \dots, x_t^N)$, 环境属性描述 $\mathbf{l}_t = (l_t^1, l_t^2, \dots, l_t^N)$, 行为 $\mathbf{u}_t = (u_t^1, u_t^2, \dots, u_t^N)$, 整个系统原子命题集 AP_c 由每个个体独自的原子命题集取并集得到, 概率数值、行为代价都基于个体相应数值计算获得. 系统整体行为 $\mathbf{u} = \mathbf{u}_1 \mathbf{u}_2 \dots$, 由全局规划所得策略, 根据系统状态与环境描述得到.

对于一个群体系统, 任务公式由每个个体独自任务公式通过逻辑运算符 $\wedge$ 连接获得, 总体任务区分软、硬约束两部分, 将系统任务公式按3.2小节中方法转化为自动机模型参与规划. 这里得到的群体系统任务模型与3.2小节中描述模型形式完全一致.

4 智能体行为规划问题求解

4.1 不确定环境下行为规划方法框架

4.1.1 Product自动机与AMEC提取

应用模型检测方法的思路, 生成Product自动机 $P = M \times A_\phi$, 由7元组表示^[15]:

$$P = (S, U, E, p_E, c_E, s_0, Acc_p), \quad (14)$$

其中 $S \subseteq X \times 2^{AP} \times Q$ 是Product自动机中的状态集, 是由位置信息、环境信息与任务状态形成的组合状态; U 是状态对应的行为集, $U(s) = U(x)$, $\forall s = (x, l, q)$; $E = \{(s, u) \mid s \in S, u \in U(s)\}$ 是Product自动机中状态与行为的组合; $p_E : S \times U \times S \rightarrow [0, 1]$ 描述状态间转移的概率, 满足 $p_E(s, u, s') = p_D(x, u, x') \cdot p_L(x', l')$, 其中 $s = \langle x, l, q \rangle$, $s' = \langle x', l', q' \rangle$, $(x, u) \in D$ 且 $q' = \delta(q, l)$; c_E 是成本函数, 满足 $c_E(\langle x, l, q \rangle, u) = c_D(x, u)$, $\forall (\langle x, l, q \rangle, u) \in E$; $s_0 = \langle x_0, l_0, q_0 \rangle \in S$ 是Product自动机上的初始状态; $Acc_p = \{(H_p^i, I_p^i) \mid i = 1, 2, \dots, N\}$ 是可接受状态对的集合, 其中 $H_p^i = \{\langle x, l, q \rangle \in S \mid q \in H_A^i\}$, $I_p^i = \{\langle x, l, q \rangle \in S \mid q \in I_A^i\}$, 一个无穷序列 $s_0 s_1 \dots$ 是可接受的, 当存在

一可接受对 (H_p^i, I_p^i) , 序列与 H_p^i 相交有限次而与 I_p^i 相交无限次, 这与DRA自动机上的定义是相似的.

确定性问题中Product自动机上无限状态序列 $s = s_0 s_1 \cdots$ 可以表示为^[11]

$$s = s_0 s_1 \cdots s_n (s_{n+1} s_{n+2} \cdots s_N)^\omega. \quad (15)$$

从0时刻到 n 时刻的有限序列称为系统轨迹前缀, 后面剩余的无限序列称为系统轨迹后缀, 这与任务自动机中状态序列 $q = q_0 q_1 \cdots q_n (q_{n+1} q_{n+2} \cdots q_N)^\omega$ 是类似的, 引入不确定性后, 序列中不再有周期循环的部分, 但仍可分解为从初始状态开始的有限长状态序列(轨迹前缀)和反复经过接受状态的无限长状态序列(轨迹后缀). 通过在Product自动机上提取AMEC $\mathcal{E}_{\text{acc}} = \{\mathcal{E}_{\text{acc}}^i, i = 1, 2, \dots\}$, 使得对于可接受对 (H_p^i, I_p^i) , H_p^i 中元素只存在于系统前缀, I_p^i 中某些元素存在于系统轨迹后缀在状态序列中反复出现. 这里前缀部分可理解为从初始状态进入AMEC内的部分, 后缀部分为AMEC内部状态转移的部分.

4.1.2 线性规划求解策略

根据初始状态与AMEC状态集的可达关系对Product自动机中状态做划分, 将AMEC以外模型转化成为存在两个吸收态集合的概率转移图模型, 系统状态最终进入AMEC或不良状态. 不良状态集合记为 S_d , AMEC对应状态集的并集记为 S_c . 剩余状态中初始状态可达的部分组成的集合记为 S_n .

在生成的Product自动机模型上完成点集划分后进行最优策略的求解. 面向同一行为策略 $\pi_{\text{pre}}: s \times u \rightarrow [0, 1]$, $s \in S_n, u \in U(s)$, 生成的所有状态序列, 以无限时长内系统在前缀部分状态 s 使用行为 u 的次数的期望值 $y_{s,u}$ 为规划变量, 约束条件描述不同状态下规划变量间的递推关系, 优化这一任务区间的总成本. 前缀部分最优策略通过求解一线性规划问题得到^[15].

后缀部分的策略同样以线性规划的方式求解, 定义AMEC内部硬约束任务常返态集合 I_{hard} . 面向同一行为策略 $\pi_{\text{suf}}: s \times u \rightarrow [0, 1]$, $s \in S_c, u \in U(s)$, 生成的所有状态序列, 序列定义为从 I_{hard} 出发最终返回 I_{hard} 的一次循环过程. 由于任务模型的设计, 一次循环过程中系统状态会依次经历软、硬约束任务的可接受状态.

以无限时长内系统在后缀部分状态 s 使用行为 u 的次数的期望值 $z_{s,u}$ 为规划变量, 约束条件描述规划变量间递推关系与前缀规划结果对后缀的影响, 优化执行后缀部分任务的单次循环成本. 同样通过求解一线性规划问题得到后缀部分最优策略.

考虑前缀部分与后缀部分独立规划中存在联系, 即系统初次访问后缀状态的概率分布受前缀部分策略生成的影响, 而这与后缀规划相关. 设计耦合线性规划以得到面向全程成本最优的策略^[15]. 规划目标由式(16)定义(α 为权重因子), 同时考虑前、后缀两部分约束条件.

$$\begin{aligned} \min \quad & \alpha C_{\text{pre}} + (1 - \alpha) C_{\text{suf}}, \\ \text{s.t.} \quad & \begin{cases} y_{s,u}, z_{s,u} \end{cases} \end{aligned} \quad (16)$$

前缀、后缀规划对应模型约束.

4.2 任务网络的松弛

在生成组合任务模型后, 按4.1小节中方法框架需对Product自动机做AMEC提取. 对AMEC的直接提取会对Product自动机产生过度裁剪, 例如原自动机中存在对于软约束任务的不良状态, 经过AMEC提取, 这种不良状态以及可能存在风险进入不良状态的转移都会被去除, 软约束任务给定时, 一般在完成上都存在失败风险, 经过行为约束后, AMEC规模会很小或出现AMEC不存在的问题, 同样也无法体现硬约束与软约束两部分在规划上的差异.

考虑任务表达中软约束要求, 很多相关研究^[15,20]都在智能体-任务模型中的状态间转移上做松弛处理, 通过在智能体运行过程中建立临时策略允许一些不合法的状态转移来实现. 这种转移是在任务层面的处理, 通常意味着一部分或整体任务的放弃, 通过牺牲任务完成度跳出不良状态. 通过对网络模型的松弛, 系统可以进入一些存在低风险的状态, 以加快任务完成进度, 而不会停滞于某一状态或某一状态集中.

在本文中, 这里需要对先前的任务模型进行修改. AMEC的提取首先需要依据一接受状态对 $(H_{\text{hard}}, I_{\text{hard}})$, 这里需要选用硬约束任务对应的接受状态对, 使得系统状态进入AMEC后, 硬约束任务一定得到满足, 假设AMEC得到后, 检验其中是否包含软约束任务接受状态对 $(H_{\text{soft}}, I_{\text{soft}})$ 中的 I_{soft} 元素, 如果存在, 则说明软约束任务可在保证硬约束任务完成的前提下完成, 如果

不存在, 则说明两任务间不相容, 组合任务的规划无法继续进行. 现有AMEC是根据 $(H_{\text{hard}}, I_{\text{hard}})$ 得到的, 里面可能存在 H_{soft} 中的状态, 这些状态在AMEC内部对于软约束任务是不良状态, 可能会在后缀策略 π_{suf} 下反复经过, 或进入后滞留在某任务层不能脱离, 使软约束任务失败. 尤其是后者, 在双层DRA中, 若系统状态一直不能到达其对应的 I_{soft} 中的状态, 会影响硬约束任务的进度, 从而使整个任务进度停滞, 这违反了对软约束任务的要求, 而这一部分状态又无法由AMEC提取的方式去除. 通常研究^[15,16]中, 此类问题的解决方法是设置在Product自动机层面的策略, 这类方法在任务层面外考虑了智能体状态层, 方法较为繁琐, 需要考虑很多环境因素, 且无法处理系统状态进入吸收态的问题. 这里通过调整任务自动机中状态转移关系解决这一问题: 将所有从 H_{soft} 离开或自循环的转移的末端状态设置为 I_{soft} 中的状态, 这样做的目的是, 一旦任务状态到达 H_{soft} 中, 则说明软约束任务失败, 视其本轮任务已经完成, 软约束任务终止而直接进入硬约束任务层, 仅仅对任务层结构的改变仍然可使系统在Product自动机上状态转移时满足合法. 在更新的网络模型中可以体现放宽了对智能体状态进入软约束任务失败状态与风险状态的限制, 此时提取得到的AMEC既能满足硬约束任务的要求, 同时也体现了软、硬约束任务之间的差异.

后缀规划中除了要考虑降低完成任务过程每次循环所需成本外, 还要降低系统状态进入 H_{soft} 的频率, 可以在目标函数上使用加权的形式, 由惩罚因子限制系统状态向 H_{soft} 内的转移. 在AMEC内定义一子MDP $Z_{\text{suf}} = (S_r, U_r, E_r, y_0, p_r, c_r)$, 与2.1小节中MDP定义类似, 将状态 I_{hard} 复制为 $I_{\text{in}}, I_{\text{out}}$, 一份继承进入该集合的转移, 一份继承脱离该集合的转移, 规划中定义 s_{bad} 代表 H_{soft} 中状态. 定义 $S'_r = S_r \setminus \{s_{\text{bad}}\}$.

$$\min_{z,s,u} \sum_{(s,u)} \left(\sum_{s' \in S'_r} z_{s,u} p_r(s, u, s') c_r(s, u) + z_{s,u} p_r(s, u, s_{\text{bad}}) d \right), \quad (17)$$

式中表达了后缀规划中目标函数的表达形式, 目标函数分为正常执行任务的行为成本与因软约束任务失败

而增加的惩罚量, 通过改变罚因子的取值可影响最优策略的求解.

4.3 前缀规划的行为约束

这里运用生成AMEC过程行为约束的思想^[11], AMEC的提取过程本质是对Product自动机上每个状态做行为筛选, 若一个状态上的某个行为存在违规转移, 即使系统状态脱离原状态集, 则将该行为从该状态上的行为集中裁剪掉, 直至被考虑的状态集内任何状态的行为都只能使系统状态在状态集内部转移, AMEC本身是强连通集, 用于确保后缀部分任务可完成, 而在这里考虑任务硬约束, 对系统前缀的规划中, 使用类似的方法, 使得系统状态保留在安全区域中, 而将使任务完成存在风险的状态, 行为从原集中删除.

这里设集合 S_n 为原集合, 该集合中元素对于Product自动机中初始状态可达, 集合去掉前缀部分中不良状态集合 S_d 与AMEC对应集合 S_c 的并集. 在通过线性规划求解策略前对模型做如下处理: 遍历该集合中所有状态元素, 若状态 s 下的行为 $a_s^i (i = 1, 2, \dots, N_s)$ 存在概率使系统状态由 S_n 转移至 S_d , 则将该行为从状态 s 的行为集 $U(s)$ 中移除, 若该状态在行为约束后无可进行行为, 则将状态 s 从原集合中移除, 并入集合 S_d , 重复此过程直至原集合状态集不再改变, 得到适应任务硬约束的前缀部分Product自动机. 这一方法将所有可能使系统进入不良状态的转移移除, 因此在执行过程中不会出现任务失败的情况. 这一方法面向硬约束任务, 用于求解满足全程硬约束要求的策略.

5 仿真测试

这一节中将给出仿真结果用于验证本文上述方法, 仿真工作通过Python完成, 在计算机上运行 (2.30 GHz CPU, 4 G内存). 仿真借助MDP_TG工具箱¹⁾, 自动机生成使用ltl2dstar工具²⁾.

在仿真中, 智能体工作环境是一个10 m×10 m的环境, 并做栅格化处理, 分为2 m×2 m的小区域. 区域坐标原点位于左下角, 整个地图分为25个离散位置, 智能体起始位置位于(1 m, 1 m)的位置. 环境属性包括

1) Guo M. MDP_TG. GitHub, 2017

2) Klein J. Ltl2dstar-LTL to deterministic streett and rabin automata. Ltl2dstar, 2007

{“风险区”、“障碍”、“基地1”、“基地2”、“基地3”}, 每个位置都有对每个环境属性概率分布的描述, 其中基地设置在固定位置, 障碍根据不同任务设置在不同位置. 智能体状态为自身所在位置的坐标, 以及正前方的方向. 自身行为包括“前进”、“后退”、“向左转”、“向右转”、“保持静止”, 每个行为都有较大概率正确完成, 实现期望的智能体状态改变, 小概率出现“故障转移”, 在仿真实验中, “前进”行为以小概率向侧前方移动, “后退”行为以小概率向侧后方移动, “向左转”、“向右转”行为以小概率按原方向旋转180°, “保持静止”行为以小概率向左、右方向旋转90°, 概率通过人为给定.

以单个智能体运行情况为例, 图5为运动机器人在给定环境下完成时序逻辑任务的仿真示意图, 其中地图边缘3个区域为基地, 中心深色区域为障碍, 周围浅色区域为风险区, 任务公式为

$$\varphi_{\text{hard}} = (\Box \Diamond \text{Base}_1) \wedge (\Box \Diamond \text{Base}_2) \wedge (\Box \Diamond \text{Base}_3) \wedge (\Box \neg \text{Obs}), \quad (18)$$

$$\varphi_{\text{soft}} = \Box \neg \text{Obs_Risk}, \quad (19)$$

意为三个基地间巡回并避开障碍, 尽量不经过风险区. 图中折线代表智能体行进的路径, 由图可知, 根据规划所得最优策略, 智能体能够满足风险要求地执行任务. 整个执行过程不违反硬约束要求, 同时在较为狭窄的区域, 智能体一定程度忽略了软约束要求以实现整体任务的顺利执行.

图6为软、硬约束形式任务与单一任务任务完成效率地对比图, 效率以规定时长内在Product自动机上循环到达接受状态的次数, 改变时长的设定, 每次实验观察100次运行结果并求得每次执行过程的效率均值. 图中两条曲线分别表示通过单一公式描述任务, 不加以重要性的区分, 与将任务分为两个公式表达两种方式的效率对比. 由图可知, 通过软、硬约束的描述, 加之以合理的建模与策略规划, 可以大大地提高任务整体的工作效率, 而不是停滞于某一任务状态. 给定环境与任务, 通过添加非法转移, 后缀部分状态数由328增加至510, 后缀部分任务完成自由度得以提高.

图7为规划惩罚项对软约束任务规划的影响示意图, 改变规划中惩罚项的权重, 记录规定时长内系统违反软约束任务的次数. 改变惩罚项权重系数取值, 记录100次运行结果中每次运行违反软约束任务的次数并求得

图5 运动机器人仿真示意图

Figure 5 Mobile robot simulation.

图6 任务效率对比示意图

Figure 6 Comparison of task efficiency.

图7 软约束任务风险示意图

Figure 7 Risk of soft specification.

均值作为软约束任务违反程度的测量值. 由图可知, 随着罚项权重的增大, 违反任务程度逐渐降低, 对软约束任务完成的要求也更为严格.

6 结论

本文在考虑不确定性因素情况下, 提出新的模型与算法完成面向LTL描述, 含软、硬约束表达任务的智能体行为规划, 满足风险约束并实现成本最优化.

考虑智能体行为与环境因素的不确定性, 智能体模型通过马尔可夫决策过程表示, 任务模型由双层DRA模型表示, 使用分层的方式区分描述软、硬约束. 在原任务模型基础上针对软约束任务添加非法转移边来应对对应Product自动机上AMEC的提取, 使得面向软约束部分任务, 网络模型得到松弛, 提高了任务整体的完成效率, 并可通过调节惩罚因子控制松弛程度. 考虑任务全程硬约束的要求, AMEC的性质保证了轨

迹后缀任务完成的安全性, 对于前缀部分则通过对模型做行为约束保证约束的满足. 通过耦合线性规划在Product自动机上求解最优随机策略以满足任务风险的约束与成本的最优, 并可求解策略应用于底层执行.

今后可以改进的内容主要是以下两方面.

(1) 多智能体规划中计算复杂度仿真过程中随着环境描述更加复杂、控制智能体数量的增加, 计算复杂度会增长迅速, 使得该方法难以与实际复杂环境相适应, 需要更有效率的方法来弥补多智能体在这方面出现的问题.

(2) 双层DRA的转移关系. 为体现软约束任务的特点, 在对双层DRA对应Product自动机做AMEC提取前进行了添加非法转移边的处理, 意义在于软约束任务失败后假定软约束任务完成而跳过当前一轮任务, 可进一步改进算法, 使得软约束任务仍能部分完成而不是整体被移除.

参考文献

- 1 Zhang Y S, Wang D L, Ruan X Y, et al. Control strategy for multiple capsule robots in intestine (in Chinese). *Sci Sin-Tech*, 2011, 41: 1551–1560 [张永顺, 王殿龙, 阮晓燕, 等. 肠道内多胶囊机器人的控制策略. 中国科学: 技术科学, 2011, 41: 1551–1560]
- 2 Zhuang Y, Gu M W, Wang W, et al. Multi-robot cooperative localization based on autonomous motion state estimation and laser data interaction (in Chinese). *Sci Sin-Inf*, 2010, 40: 1351–1362 [庄严, 顾明伟, 王伟, 等. 基于自主运动状态估计及信息交互的多移动机器人协作定位. 中国科学: 信息科学, 2010, 40: 1351–1362]
- 3 Guo M, Dimarogonas D V. Reconfiguration in motion planning of single- and multi-agent systems under infeasible local LTL specifications. In: *Proceedings of the 52nd IEEE Conference on Decision and Control*. Florence: IEEE, 2013. 2758–2763
- 4 Cho K, Oh S. Learning-based model predictive control under signal temporal logic specifications. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Brisbane: IEEE, 2018. 7322–7329
- 5 Wu B, Hu B, Lin H. A learning based optimal human robot collaboration with linear temporal logic constraints. ArXiv: [1706.00007](https://arxiv.org/abs/1706.00007)
- 6 Lahijanian M, Andersson S B, Belta C. Temporal logic motion planning and control with probabilistic satisfaction guarantees. *IEEE Trans Robot*, 2012, 28: 396–409
- 7 Ding X, Smith S L, Belta C, et al. Optimal control of markov decision processes with linear temporal logic constraints. *IEEE Trans Automat Contr*, 2014, 59: 1244–1257
- 8 Lahijanian M, Andersson S B, Belta C. Formal verification and synthesis for discrete-time stochastic systems. *IEEE Trans Automat Contr*, 2015, 60: 2031–2045
- 9 Cizelj I, Belta C. Control of noisy differential-drive vehicles from time-bounded temporal logic specifications. *Int J Robotics Res*, 2012, 33: 1112–1129
- 10 Ulusoy A, Wongpiromsarn T, Belta C. Incremental controller synthesis in probabilistic environments with temporal logic constraints. *Int J Robotics Res*, 2014, 33: 1130–1144
- 11 Baier C, Katoen J P. *Principles of Model Checking*. Cambridge: MIT Press, 2008
- 12 Vardi M Y. Probabilistic linear-time model checking: An overview of the automata-theoretic approach. In: *Proceedings of the 5th International AMAST Workshop on Formal Methods for Real-Time and Probabilistic Systems*. Berlin: Springer, 1999. 265–276

- 13 Kwiatkowska M, Norman G, Parker D. PRISM 4.0: Verification of probabilistic real-time systems. In: Proceedings of the International Conference on Computer Aided Verification. Berlin: Springer, 2011. 585–591
- 14 Ding X C, Smith S L, Belta C, et al. MDP optimal control under temporal logic constraints. *Mathematics*, 2011, 413: 532–538
- 15 Guo M, Zavlanos M M. Probabilistic motion planning under temporal tasks and soft constraints. *IEEE Trans Automat Contr*, 2018, 63: 4051–4066
- 16 Guo M, Dimarogonas D V. Reconfiguration in motion planning of single- and multi-agent systems under infeasible local LTL specifications. In: Proceedings of the 52nd IEEE Conference on Decision and Control. Florence: IEEE, 2013. 2758–2763
- 17 Guo M, Johansson K H, Dimarogonas D V. Motion and action planning under LTL specifications using navigation functions and action description language. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems. Tokyo: IEEE, 2013. 240–245
- 18 Altman E. Constrained Markov decision processes with total cost criteria: Occupation measures and primal LP. *Math Methods Operations Res*, 1996, 43: 45–72
- 19 Klein J, Baier C. Experiments with deterministic ω -automata for formulas of linear temporal logic. *Theor Comput Sci*, 2006, 363: 182–195
- 20 Guo M, Dimarogonas D V. Multi-agent plan reconfiguration under local LTL specifications. *Int J Robotics Res*, 2014, 34: 218–235

Probabilistic action planning based on linear temporal logic

CHEN ZhongYao & FANG Hao

School of Automation, Beijing Institute of Technology, Beijing 100081, China

With the increase of people's demand for agents, the activities of agents are no longer limited to simple environment and task format. Facing complex application scenarios, agents need to be able to make decisions and execute them autonomously. This paper studies the probabilistic action planning considering complex task constraints described by linear temporal logic. At the same time, the success rate and cost of task are both considered. The uncertain factors include agent behavior and environment attributes, and the task description is expressed by soft and hard constraints. The strategy of agent is generated here applying model checking in formal method. Single- and multi-agent model is established using Markov decision process, while task model is established using double-layer automata. Then, agent-task network model is designed to describe the constraints and the control strategy is solved through a coupled linear programming. The method above is verified through numerical simulation. The results show that the complex task constraints in the form of soft and hard constraints can be satisfied. The optimal strategy enable the agent to complete the task according to the constraint strength, and the control strategy can be adjusted by controlling the relaxation degree of control network model relevant to the penalty factor.

linear temporal logic, probabilistic action planning, double-layer automata, soft/hard constraint

doi: [10.1360/SST-2019-0292](https://doi.org/10.1360/SST-2019-0292)