Semi-autonomous Planning: A novel approach for plan segmentation in multiagent planning

Cassio H. M. P. Pereira[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
Caixa Postal 19.081 – 81531-980 – Curitiba – PR – Brazil

chmppereira@inf.ufpr.br

Abstract. *This article presents an extension to the HEART planner, which has a mechanism to solve planning problems by using hierarchical task networks and partial ordering controlled by causal links that allows intermediate plans. This extension introduces a tool that allows for the usage of the planner in the distributed multiagent environment by exploiting that capability for creating subproblems, allowing less capable agents to help solve higher complexity problems without sacrificing the real-time execution. For that end it is presented a new type of flaw, noted external attribution flaw, and it is proven that its inclusion maintains the planner properties of soundness and completeness.*

Resumo. *Este artigo apresenta uma extensão ao planejador HEART, que possui um mecanismo para resoluver problemas de planejamento através de rede de tarefas hierárquicas e ordenação parcial controlada por vínculos causais, permitindo planos intermediários. Esta extensão é uma ferramenta que permite seu uso em ambiente multiagente distribuído aproveitando desta capacidade de geração de subproblemas, permitindo que agentes de menor capacidade possam auxiliar na resolução de problemas mais complexos sem sacrificar a execução em tempo real. Para tanto é apresentado um novo tipo de falha, denominada falha de atribuição externa, e é provado que sua inclusão mantém a corretude e completude do planejador.*

1. Introdução

A tomada de decisões é necessária em diversas situações, e ainda mais relevante quando existe um objetivo a ser atingido. Quando este objetivo é distante ou complexo se torna importante também a criação de passos a serem seguidos que permitam atingir este objetivo. A esta sequência de passos pode ser dado o nome de *plano*, e o processo que define este plano é o *planejamento*. Quando este planejamento ocorre através do uso de ferramentas automatizadas e técnicas de inteligência artificial então temos a área que será abordada, denominada *Problema de Planejamento em Inteligência Artificial*.

Este artigo foca na variante denominada *planejamento hierárquico*, em associação ao conceito de *planejamento multiagente distribuído*, apresentando um novo método de controle da participação de agentes de capacidade reduzida no processo, através da divisão do problema original em subproblemas conhecidamente tratáveis por estes agentes.

2. Motivação e Trabalhos relacionados

A funcionalidade apresentada neste artigo tem como inspiração principal um conceito apresentado para o sistema *CASPER* de controle de robôs de pesquisa exploratória. Este

conceito é então mapeado para seu equivalente em ambiente de planejamento multiagente, utilizando ordenação parcial por vínculos causais para melhor dividir subproblemas.

Para a implementação desta ideia foi escolhido o planejador hierárquico *HEART*, que tem como foco a explicabilidade de planos e geração de planos parciais para acompanhamento humano, originalmente visando maior capacidade de compreensão de motivos tanto de escolhas em planos válidos quanto de falhas em problemas sem solução.

Este conjunto de planejamento hierárquico, explicabilidade e planos parciais permite a implementação através de novos controles para ambiente multiagente e um novo tipo de *falha* para a divisão do problema entre agentes, permitindo assim a manutenção das características e capacidades originais do planejador.

2.1. Sistema *CASPER*

O sistema CASPER [Knight et al. 2001] foi desenvolvido para uso em robôs exploradores planetários que precisavam ter maior grau de autonomia. Antes disso robôs deste tipo dependiam de comunicação direta e constante para recebimento de ordens, o que para distâncias maiores limitava a complexidade das tarefas que poderiam ser executadas, pelo risco de imprevistos e outros fatores que necessitassem de reação imediata ou replanejamento. O sistema CASPER inovou através da mudança da responsabilidade do controlador remoto, que passou a se restringir à definição dos objetivos a serem buscados, deixando a tarefa de criar e acompanhar o andamento do plano para o agente executor.

Repassar a criação e controle do planos ao agente minimiza tempo gasto com comunicação externa, tornando viável a execução de tarefas mais complexas, tais como *planejamento em tempo real* e *planejamento em mundo aberto*, em sua junção *planejamento contínuo*, e a recepção e correção de planos com *planejamento em múltiplos níveis*.

Planejamento em tempo real é um subconjunto de planejamento temporal [Fox and Long 2003] apresentado originalmente em [Korf 1990] com o algoritmo *LRTA**. Tem como foco antecipar ao máximo o início da execução do plano, tentando manter o plano o mais próximo do ótimo e evitando *depressões heurísticas* (repetição de estados em rápida sucessão [Hernández and Baier 2012]).

Planejamento em mundo aberto é uma expansão do planejamento com incerteza [Boutilier et al. 1999] que descarta a suposição de mundo fechado, necessitando ações sensoriais para descobrir o estado inicial e atual do mundo, devido a valores e literais não conhecidos de antemão [Bonet and Geffner 2000].

A junção destes dois conceitos gera o problema denominado planejamento contínuo, em que o planejamento e a execução ocorrem em paralelo, com o planejador escolhendo ações, agendando, executando, conferindo o resultado esperado, e caso necessário corrigindo o plano através de *conserto iterativo* [Rabideau et al. 1999].

É nesta capacidade de receber e revisar objetivos de maior nível de abstração durante a criação e execução do plano que a expansão deste artigo se baseia, aplicando-a em ambiente multiagente para uso em situações em que os agentes possam compartilhar entre si a necessidade e capacidade de geração de planos em ambientes de alta complexidade (relativa à capacidade dos agentes executando o plano).

2.2. Planejamento multiagente

Um sistema pode ser denominado *multiagente* caso exista mais de um componente presente no ambiente, com capacidades bem definidas e tentando atingir algum objetivo através da manipulação do mesmo. Modelagem de problemas para agentes [Jennings 1999] vem sendo estudada em outros contextos por décadas, portanto a explicação se limitará ao contexto de sua aplicabilidade para o problema de planejamento.

Em planejamento multiagente o problema é usualmente abordado como uma coleção de subproblemas, a serem alocados aos agentes participantes para que sua execução atinja o maior número de objetivos comuns e privados possíveis. Este processo de resolução pode ser dividido nas seguintes etapas, adaptadas de [Durfee 2001]:

1. Alocação de objetivos a agentes;
2. Refinamento do objetivo global;
3. Coordenação pré-planejamento;
4. Montagem de planos individuais;
5. Coordenação pós-planejamento;
6. Execução do plano consolidado.

Devido à grande variedade presente nestes problemas cada etapa pode ser alterada, estendida ou suprimida para melhor conformar ao ambiente que está sendo aplicada, mas para o escopo a ser utilizado esta definição basta.

2.3. Ordenação Parcial

A ordenação parcial por vínculos causais foi formalizada em [Barrett and Weld 1994], e consiste em desconsiderar a intuição de se criar um plano linearmente, e ao invés disso utilizar as pré-condições e efeitos de cada ação como base para o ordenamento entre elas, como apresentado em [Sacerdoti 1975] e detalhado em [Weld 1994].

Para isto são atribuídos vínculos a conjuntos de literais indicando quais ações precedem outras em relação àquele conjunto (notada $s \xrightarrow{P} w$ significando que a ação s precede a ação w para o conjunto de literais P). Ações podem *ameaçar* este vínculo caso uma ação v diferente de s e w altere algum integrante de P, e a restrição de ordenação que a coloca antes de s ou depois de w é chamada de *condição de segurança*.

O algoritmo 1 mostra uma versão anotada do POCL (sigla em inglês, *Partial Order Causal Link*), adaptada de [Ghallab et al. 2004], que resolve agendamento de planos através de refinamento de falhas, e que será usada para explicar uma definição nova.

Algoritmo 1 POCL:

Entrada a agenda, $\mathcal{P}$ problema
Saída [Sucesso, Insucesso]

if Agenda $a = \varnothing$ **then**
  return Sucesso // Agenda populada faz parte da entrada, caso contrário o plano está pronto
end if
Falha $f \leftarrow escolhe(a)$ // Escolha heurística de falha
Resolvedor $R \leftarrow resolve(f, \mathcal{P})$
for all $r \in R$ **do**
  aplica(r, π) // Aplica o resolvedor ao plano parcial atual
  Agenda $a' \leftarrow atualiza(a)$
  if POCL($a', \mathcal{P}$) = Sucesso **then**
    return Sucesso
  end if
  reverte(r, π) // Resolução sem sucesso, reverte aplicação do resolvedor
end for
$a \leftarrow a \cup \{f\}$ // Falha não resolvida
return Insucesso // Volta para a última escolha de r

2.4. Planejador HEART

O planejador HEART, apresentado em [Gréa et al. 2018] foi usado para a prova do conceito de planejamento semi-autônomo, e também serviu para embasamento e prova teórica do mesmo. Foi escolhido por seu foco em explicabilidade, sua capacidade de gerar planos parciais e pelo uso de planejamento em múltiplos níveis através de *HTN*.

O conceito de planejamento explicável visa tornar um plano compreensível para humanos. O foco em aumento de performance e robustez de planejadores tende a afastar ferramentas da noção de deliberação por agentes[Ghallab et al. 2014]. Para permitir melhor entendimento das motivações por trás das ações escolhidas criou-se o campo de explicabilidade, que tem tido crescente interesse [Ghallab et al. 2016, Winikoff et al. 2018, Sreedharan et al. 2019]. Neste contexto o planejador HEART foca em gerar planos válidos mesmo nos primeiros momentos, ainda com alto nível de abstração, através do uso de redes hierárquicas de tarefas e ordenação parcial por vínculos causais.

O formalismo de *rede hierárquica de tarefas* (usualmente referenciado pela sua sigla em inglês *HTN*) [Erol et al. 1994] foi criado para agilizar a montagem de um plano através da identificação de etapas intermediárias, permitindo maior capacidade de representação pela separação de tarefas entre *primitivas* (correspondentes a uma ação na representação STRIPS) e *redes de tarefas*, tarefas abstratas que representam um conjunto de tarefas primitivas e indicam mesmo antes de sua especificação o que se pretende atingir naquele ponto do plano. Esta separação tem como uma das principais vantagens que a definição de ações a serem realizadas pode ser postergada, possibilitanto priorizar um plano válido em maior nível de abstração antes de definir o passo a passo.

3. Planejamento semi-autônomo

Nesta seção é apresentado um novo conceito, denominado *planejamento semi-autônomo*. Este novo conceito apresenta uma abordagem alternativa para planejamento baseada na

separação de responsabilidades utilizada pelo sistema CASPER e utilizando como base as características do planejamento HTN e multiagente.

3.1. Motivação para o novo conceito

No contexto de planejamento multiagente podemos classificar sistemas de acordo com a responsabilidade de geração de planos por seus agentes de duas maneiras. Podemos definir um sistema como *autônomo* caso seus agentes sejam responsáveis pela elaboração e execução do plano, controlando cumprimento de objetivos e observando a manutenção das restrições do problema. Em contraponto, um sistema cujo planejamento é *centralizado* pode ser definido como possuindo agentes não autônomos pois apesar de a execução das ações ser distribuída, a responsabilidade de geração, acompanhamento e resolução de conflitos são atribuídas a um agente centralizador.

Com base nesta classificação pode-se definir como planejamento semi-autônomo o meio termo entre os dois, onde parte da responsabilidade de geração e acompanhamento de subplanos mais restritos é repassada ao agente executor das ações, enquanto cabe a um agente *centralizador* ou *supervisor* a coordenação de planos em maiores níveis abstração. Este conceito não possui exploração ampla dentro da comunidade científica no contexto de planejamento, sendo abordado apenas tangencialmente em análises de coordenação multiagente totalmente distribuída ou por inteligência de enxame de algoritmos evolutivos [Godoy et al. 2016, Contreras-Cruz et al. 2017].

Esta nova maneira de realizar a divisão de responsabilidades usa uma rede hierárquica para permitir automatização da segmentação durante a resolução, acomodando agentes de capacidade variável sem necessitar intervenção manual.

3.2. Embasamento teórico

O sistema CASPER usa uma divisão de responsabilidades simplificada, na qual o agente remoto recebe apenas os objetivos, gera e executa o plano. Este processo pode ser mapeado para as etadas de planejamento multiagente da seguinte forma:

1. A alocação de objetivos é realizada pelo centro de controle, atribuindo todos os objetivos definidos ao resolvedor remoto (por haver apenas um agente disponível);
2. Todos os subproblemas são atribuídos a este mesmo agente;
3. O objetivo definido pelo centro de controle é passado ao agente para planejamento;
4. A etapa de montagem de planos individuais é o processo padrão de planejamento do sistema CASPER, começando pela definição de horizonte e objetivo parcial;
5. Por não haver outros agentes para auxiliar na definição do plano a coordenação é considerada como concluída assim que o plano parcial inicial for concluído;
6. Por utilizar de planejamento contínuo, a execução de ações ocorre com o plano parcial já definido e o processo volta para a etapa de planejamento individual.

O conceito de planejamento semi-autônomo como apresentado expande as primeiras etapas do CASPER, criando uma primeira passagem de planejamento que ocorre no centralizador, enquanto as etapas posteriores ocorrem em cada agente executor de ações. Para a correta definição dos pontos de parada entre estas duas passagens é necessário que o supervisor/centralizador conheça as capacidades e restrições de todos os agentes que vão participar do planejamento.

Tabela 1. Tabela de símbolos

$\mathcal{D}, \mathcal{P}$	Domínio e problema de planejamento
$pre(a), eff(a)$	Pré-condições e efeitos da ação a
$met(a)$	Métodos que decompõem a ação a
$age(a)$	Agentes capazes de realizar a ação a
$s \xrightarrow{P} w$	Vínculo da ação s como precedente de w pela causa P
$L^{\pm}(a)$	Vínculos causais de entrada (L^{-}) e de saída (L^{+}) do passo a.
$\phi^{\pm}(l)$	Função de incidência para planos parciais.
$a_a \succ a_s$	Restrição de ordenação, onde o passo a_a antecede o passo a_s
$\downarrow_f a_n$	Objetivo parcial, fluente f não suportado pelo passo a_n
$\downarrow_e a$	Atribuição em aberto, agente e não atribuído à ação a
$a_b \bigotimes l_t$	Ameaça, onde ação a_b quebra o vínculo l_t
$a \rhd_{\pi}^{\pm} a'$	Transposição dos vínculos da ação a' para a ação a
A_a^n	Conjunto de ações próprias da entidade a, até o nível de abstração n
$lv(a)$	Nível de abstração da entidade a
$a_n \bigoplus^m$	Decomposição da ação composta a_n utilizando o método m
$a_n \bigoplus^{m,e}$	Atribuição externa da ação a_n para agente e através do método m
$var : exp$	Separador a ser lido como var "tal que"exp

Neste novo modelo as etapas de coordenação são responsabilidade do centralizador, que gera planos de níveis de abstração variáveis, de acordo com a capacidade de cada agente executor. Será utilizada a nomenclatura estabelecida pelo planejador HEART, apresentada na tabela 1. As etapas ficam então definidas da seguinte maneira:

1. O centralizador monta o plano, dinamicamente atribuindo tarefas para agentes capacitados, e interrompendo ao atingir o ponto em que não existam falhas a serem resolvidas ou todos os agentes tenham uma ação composta atribuída;
2. O centralizador complementa cada subproblema com as ações atômicas que o antecedem e sucedem e ainda não atribuídas, para garantir que todos os passos tenham um agente executor vinculado;
3. Os subproblemas são repassados para os agentes executores;
4. Planejamento individual de cada agente. Havendo falhas cada agente pode realizar o reparo iterativo de seu subproblema, e caso seu subproblema não possua solução então o centralizador é avisado do insucesso;
5. Em caso de sucesso agentes comunicam o plano (apenas com ações atômicas) que resolvem seu subproblema, e em caso de insucesso o centralizador indica que aquela ação abstrata não está disponível àquele agente e reinsere a falha na agenda;
6. O problema está resolvido quando o plano armazenado no centralizador atinge $lv(\pi) = \varnothing$, a agenda está vazia e todos os agentes executores já apresentaram resposta válida de seus subproblemas. Caso ainda existam falhas na agenda após a resposta de todos os agentes (positiva ou negativa) o processo volta à definição de subproblemas da primeira etapa, até que seja atingido o estado objetivo ou que não haja resolvedores disponíveis (indicando problema sem solução).

Estas etapas requerem revisão e novas estruturas ao processo padrão do planejador HEART, tanto para adequação ao ambiente multiagente quanto para acompanhamento e

conferência, portanto vamos rever desde a definição do problema:

Definição 3.1. (Problema). O problema de planejamento passa a ser a tupla $\mathcal{P} = \langle \mathcal{D}, C_{\mathcal{P}}, a_0, \mathcal{E} \rangle$, onde $\mathcal{E}$ é o conjunto de agentes executores do plano. O supervisor não precisa estar listado, e aos agentes pode ser passado o problema com a definição original caso não seja desejada interação entre eles.

Para a escolha do agente executor responsável por ações abstratas é necessário que o centralizador saiba todas as ações que cada agente executor é capaz de resolver.

Definição 3.2. (Ação). Uma ação passa a ser definida pela tupla parametrizada $a(args) = \langle nome, pre, eff, met, age \rangle$, onde $age \subseteq \mathcal{E}$ é o conjunto de agentes que são capazes de executar a ação. Para ações abstratas este conjunto implica que o agente também é capaz de executar todos os métodos presentes em met desta ação recursivamente.

Esta nova informação é utilizada na etapa inicial do novo processo, e permite ao centralizador saber durante a decomposição quais são os agentes que estão aptos a receber dada ação como subproblema, e também permite ao processo identificar quando uma ação está indisponível (nenhum agente é capaz de resolvê-la).

A indicação de agentes é notada como um novo tipo de falha, chamada *falha de atribuição externa*.

Definição 3.3. (Falha). Uma falha de atribuição externa é um tipo específico de *condição aberta*, notada $\downarrow_e a$ que indica que uma ação a ainda não possui agente $e \in age(a)$ vinculado. Funciona de maneira similar a um subojbjetivo na definição original do HEART, com a diferença de que opera agentes sobre ações (ao invés de passos sobre fluentes).

Todo plano começa com falhas de atribuição externa para cada subobjetivo em aberto, a inclusão de qualquer ação sem agente definido também inclui uma falha de atribuição externa para a mesma, e a remoção desta ação remove também esta falha. O único meio de resolver esta falha é através de um novo resolvedor, chamado *resolvedor de atribuição externa*.

Definição 3.4. (Atribuição externa). O resolvedor de atribuição externa funciona de maneira análoga ao resolvedor de decomposição, porém indica ao centralizador que a ação será passada para resolução por um agente executor. Sua notação reflete esta similaridade, sendo notado onde necessário como $a \bigoplus_{\pi}^{m,e}$, indicando que a ação a do plano parcial π está sendo atribuída para resolução pelo agente e através do método m.

- *Resolvedor:* Uma falha de atribuição externa é resolvida por um *resolvedor atribuidor*. Este resolvedor repassa o requerente com um método $m \in met(a_n)$ no plano π para um agente externo $e \in age(a_n)$. Em caso de resposta postiva o agente vai devolver ações primitivas equivalentes à resolução completa da transposição: $a_n \bigoplus_{\pi}^{m,e} = \langle S_m \cup (S_\pi \backslash \{a\}), a_n \rhd^- I_m \cup a_n \rhd^+ G_m \cup (L_\pi \backslash L_p i(a_n)) \rangle$.
- *Efeito colateral:* Uma falha de atribuição externa é criada pela inclusão de uma ação por qualquer outro resolvedor, e invalidada por sua remoção:

$$\bigcup_{a_m \in S_m}^{f \in pre(a_m)} \pi' \downarrow_e a_m \bigcup_{a_b \in S_{\pi'}}^{l \in L_{\pi'}} a_b \bigotimes l \bigcup_{a_c \in S_m}^{lv(a_c) \neq 0} a_c \bigoplus$$

Este resolvedor é aplicado durante a etapa de passagem de subproblemas, e caso haja resposta positiva do agente executor então o requerente é substituído pelas ações atômicas que o agente reportar como resposta de seu subproblema. Em caso de resposta negativa a falha é reinserida na agenda para resolução posterior, e o agente é removido do conjunto age da ação. Efeitos colaterais deste resolvedor são tratados exatamente igual aos resultantes da aplicação de um resolvedor decompositor.

Para que o centralizador possa conferir o andamento da atribuição de tarefas é necessário que todas as ações dos planos parciais possuam também um identificador de qual agente foi responsável pela sua solução.

Definição 3.5. (Plano parcial). Ao plano parcial armazenado pelo planejador é inclusa a função $atr(a)$, que para cada ação $a \in \pi$ atribui um agente executor $e \in \mathcal{E}$ indicando que a ação a é de responsabilidade do agente e. Ações sem agente vinculado são atribuídas a algum agente durante a passagem de subproblemas (mesmo que atômicas), e o planejamento não é finalizado até que todas as ações do plano $\pi_{lv(c_0)}$ tenham sido atribuídas a um agente com sucesso.

Estas adequações são aplicadas ao algoritmo do POCL da seguinte maneira:

- A função $resolve$ passa a considerar o conjunto $\mathcal{E}$ durante a escolha de resolvedores e pode escolher o novo tipo de resolvedor onde adequado, sendo que sua prioridade é estritamente superior à resolução de falhas de decomposição padrão, para agilizar a passagem de subproblemas a agentes executores;
- Para resolvedores de atribuição externa as variáveis do conjunto R passam a conter o identificador do agente escolhido para a atribuição, e a cardinalidade do conjunto aumenta em até $|\mathcal{E}|$ devido à possibilidade de diversos agentes serem capazes de resolver dada ação;
- Quando r for atribuição externa a função $aplica$ passa a ser $atribuir$, situação na qual a execução dos passos posteriores do laço passa a ser de responsabilidade do agente executor. O plano π' passado ao agente externo vai ser composto da ação requerente do resolvedor, assim como todas as ações atômicas antecessoras e sucessoras à ação abstrata sendo decomposta que o agente possa executar e ainda não tenham sido atribuídas, garantindo a completude da atribuição de ações;
- As funções $atualiza$ e $reverte$ passam a ter o agente escolhido como parâmetro, onde $atualiza$ atribui às ações envolvidas nesta etapa o agente que as resolveu, enquanto $reverte$ remove o agente do conjunto age das ações que lhe foram atribuídas e não puderam ser resolvidas.
- Ao final de cada ciclo o planejador sincroniza os agentes executores, esperando até que todos tenham atualizado a agenda, e caso necessário as ações. Neste momento as ações do plano parcial são analisadas para definir qual o nível de abstração do próximo ciclo e o processo se reinicia.

4. Análise teórica

Nesta seção é provado que esta variante do planejador HEART mantém as características de seus métodos e planos resultantes: O planejador é completo, correto e seus planos abstratos sempre podem ser decompostos em soluções válidas.

A completude e corretude de POCL já foi provada em [Penberthy and Weld 1992], e as mesmas características já foram provadas para o planejador HEART em

[Grea et al. 2018]. Aqui será provado que as alterações feitas ao planejador não comprometem estas características. Uma propriedade relevante dos algoritmos POCL é que a estratégia de escolha de falhas não compromete sua corretude/completude. Ressaltando que para estas provas o caractere *:* quando apresentado na expressão *var : exp* deve ser lido como var *tal que* exp.

Teorema 1. (Atribuição externa preserva aciclicidade). *A atribuição de uma ação composta com métodos válidos em um plano acíclico a um agente executor capaz de resolvê-la resulta em um plano acíclico.* Formalmente: $\forall a_s \in S_\pi : a_s \not\succ_\pi a_s \implies \forall a'_s \in S_{a \bigoplus_\pi^{m,e}} : a'_s \not\succ_{a \bigoplus_\pi^{m,e}} a'_s$.

Demonstração. A atribuição externa funciona de maneira idêntica à decomposição normal para efeito de controle do plano, sendo sua única diferença o fato de sua aplicação acontecer remotamente, portanto: Ao realizar a atribuição externa da ação composta a com um método válido m para um agente executor e capacitado a resolvê-la dentro de um plano existente π, todos os passos S_m são adicionados ao plano refinado. Tanto π quanto m são garantidos por definição como acíclicos.

Podemos notar que $\forall a_s \in S_m : \left(\nexists a_t \in S_m : a_s \succ a_t \wedge \neg f \in eff(a_t)\right) \implies f \in eff(a)$. Por extenso, se uma ação a_s contribui um fluente f para o objetivo do método m então f está necessariamente presente nos efeitos de a. Como a escolha de de ações dá preferência às de maior nível de abstração durante a escolha de resolvedores então nenhuma ação de seus métodos vai estar presente no plano parcial atual quando a atribuição externa ocorrer. Isto pode ser notado como $\exists a \in \pi \implies S_m \coprod S_\pi$, significando que o grafo formado pelos planos parciais m e π não podem conter as mesmas arestas e portanto sua aciclicidade é preservada em sua inserção.

□

Teorema 2. (Atribuições externas resolvidas não reocorrem). *A aplicação de um resolvedor de atribuição externa sobre um plano π garante que $a \notin S'_\pi$ para todo plano parcial refinado a partir de π em que não houver reversão de a.*

Demonstração. Como apresentado na definição de métodos do planejador HEART, temos que $a \notin A_a$. Isto significa que a resolução por atribuição externa não pode introduzir a através de sua decomposição ou da decomposição de suas ações próprias, mesmo que resolvidas por um agente executor. De fato, após sua expansão e a finalização do ciclo atual, pela diminuição do nível do ciclo para $c_{lv(a)-1}$ sua escolha passa a ser impedida pelo seletor de resolvedores. Tal ação não pode estar contida em métodos de ações selecionadas posteriormente pois caso contrário, pela definição de níveis de abstração, seu nível de abstração seria ao menos $lv(a) + 1$.

□

Teorema 3. (Atingir o nível de abstração 0 com todas as ações atômicas atribuídas a um agente garante solubilidade). *Encontrar um plano que contenha apenas falhas de atribuição externa ou decomposição com ações de nível de abstração 1 e com agentes executores capazes de resolvê-las garante uma solução ao problema.*

Demonstração. A introdução da necessidade de atribuir ações (mesmo que atômicas) a agentes torna este ponto mais complexo em relação à versão original do planejador HEART.

Qualquer método m disponível ao agente e de uma ação composta $a : lv(a) = 1$ é por definição uma solução do problema atribuído ao agente $\mathcal{P}_a = \langle \mathcal{D}, \mathcal{C}_\mathcal{P}, a \rangle$. Como por definição $a \notin A_a$ e $a \notin A_{a \bigoplus_\pi^{m,e}}$ (significando que a não reocorre após decomposta pelo agente executor). Também por definição temos que a instanciação da ação e seus métodos mantém coerência das restrições (tudo está instanciado antes da escolha pelo resolvedor). Como o plano π possui apenas falhas de decomposição e de atribuição externa (que por definição não diferem em referência ao controle de geração do plano) e todas as falhas dentro de m possuem garantia de solubilidade, e ambas possuem garantia de aciclicidade através da decomposição/atribuição pela aplicação de qualquer $a \bigoplus_\pi^{m,e}$, então o plano possui solução.

□

Teorema 4. (Planos abstratos garantem solubilidade). *Encontrar um plano parcial π que possua apenas falhas de decomposição/atribuição garante a existência de solução do problema*.

Demonstração. Recursivamente, ao aplicar a demonstração anterior ao plano de nível de abstração superior notamos que a decomposição ao nível 2 garante solução, pois os métodos das ações compostas possuem garantia de solubilidade.

□

A ressalva necessária à demonstração do teorema 4 é de que a informação das capacidades dos agentes executores precisa estar correta para preservar a prova, o que pode não ser verdade em ambientes multiagente em que a privacidade escolhida pelos agentes impeça ou limite a passagem das limitações de capacidade ao supervisor. Para efeito do conceito apresentado neste artigo sempre há suposição desta corretude.

5. Aplicabilidade teórica e expansões possíveis

Com base nos resultados teóricos apresentados pode-se afirmar que a utilização desta variante do planejamento proposto pelo HEART pode ser utilizada em ambientes multiagente cooperativos, se aproveitando do mecanismo de divisão de responsabilidade para permitir a agentes de capacidade reduzida a resolução de problemas impossíveis sem auxílio.

A maior restrição para a utilização da ferramenta como implementada é a mesma que o planejamento HTN já possui, na necessidade de mapear os problemas para que possua ações abstratas corretamente tipadas e agrupadas de acordo com o resultado esperado pelo domínio, exacerbado pelo uso de notação diferente da mais usual a HTN que o planejador requer e pelo novo parâmetro de identificação de agentes capazes de executar cada ação abstrata.

No tocante a expansões, existe pesquisa para a utilização de Ontologias com o expresso intuito de auxiliar a criação de domínios para planejamento [Behnke et al. 2015] que poderia melhorar a aplicabilidade do planejador, assim como o planejador poderia

ser alterado para utilizar identificação de subproblemas independentes em domínios grandes [Lotem and Nau 2000], permitindo melhora na escolha de ações e agentes durante a atribuição externa. O processo de reparo de planos também poderia ser alterado para o proposto em [Höller et al. 2018], que afirma ser mais enxuto e compatível à estrutura HTN. Referente à teoria da técnica, este mesmo conceito poderia também ser aplicado à variantes de HTN mais voltadas à acompanhamento de agentes, tal como redes GTN [Alford et al. 2016] ou TMK [Hoang et al. 2005].

Referências

Alford, R., Shivashankar, V., Roberts, M., Frank, J., and Aha, D. W. (2016). Hierarchical planning: Relating task and goal decomposition with task sharing. In *IJCAI*, pages 3022–3029.

Barrett, A. and Weld, D. S. (1994). Partial-order planning: evaluating possible efficiency gains. *Artificial Intelligence*, 67(1):71–112.

Behnke, G., Bercher, P., Biundo-Stephan, S., Glimm, B., Ponomaryov, D. K., and Schiller, M. R. G. (2015). Integrating ontologies and planning for cognitive systems. In *Description Logics*.

Bonet, B. and Geffner, H. (2000). Planning with incomplete information as heuristic search in belief space. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 52–61. AAAI Press.

Boutilier, C., Dean, T., and Hanks, S. (1999). Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11(1):94.

Contreras-Cruz, M. A., Lopez-Perez, J. J., and Ayala-Ramirez, V. (2017). Distributed path planning for multi-robot teams based on artificial bee colony. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 541–548. IEEE.

Durfee, E. H. (2001). Distributed problem solving and planning. In *ECCAI Advanced Course on Artificial Intelligence*, pages 118–149. Springer.

Erol, K., Hendler, J., and Nau, D. S. (1994). HTN planning: Complexity and expressivity. In *AAAI*, volume 94, pages 1123–1128.

Fox, M. and Long, D. (2003). PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of artificial intelligence research*.

Ghallab, M., Nau, D., and Traverso, P. (2004). *Automated Planning: theory and practice*. Elsevier.

Ghallab, M., Nau, D., and Traverso, P. (2014). The actor′s view of automated planning and acting: A position paper. *Artificial Intelligence*, 208:1–17.

Ghallab, M., Nau, D., and Traverso, P. (2016). *Automated planning and acting*. Cambridge University Press.

Godoy, J., Karamouzas, I., Guy, S. J., and Gini, M. L. (2016). Implicit coordination in crowded multi-agent navigation. In *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pages 2487–2493. AAAI press.

Gréa, A., Matignon, L., and Aknine, S. (2018). HEART: HiErarchical Abstraction for Real-Time Partial Order Causal Link Planning. In *1st Workshop on Hierarchical Planning at 28th International Conference on Automated Planning and Scheduling (ICAPS)*, pages 17–25.

Grea, A., Matignon, L., and Aknine, S. (2018). How explainable plans can make planning faster. In *Workshop on Explainable Artificial Intelligence*, pages 58–64.

Hernández, C. and Baier, J. A. (2012). Avoiding and escaping depressions in real-time heuristic search. *Journal of Artificial Intelligence Research*, 43:523–570.

Hoang, H., Lee-Urban, S., and Muñoz-Avila, H. (2005). Hierarchical plan representations for encoding strategic game ai. In *Proceedings of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 63–68. AAAI Press.

Höller, D., Bercher, P., Behnke, G., and Biundo, S. (2018). Htn plan repair using unmodified planning systems. In *Proc. of the First ICAPS Workshop on Hierarchical Planning*, pages 26–30.

Jennings, N. R. (1999). Agent-based computing: Promise and perils. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1429–1436. Morgan Kaufmann Publishers Inc.

Knight, S., Rabideau, G., Chien, S., Engelhardt, B., and Sherwood, R. (2001). Casper: Space exploration through continuous planning. *IEEE Intelligent Systems*, 16(5):70–75.

Korf, R. E. (1990). Real-time heuristic search. *Artificial intelligence*, 42(2-3):189–211.

Lotem, A. and Nau, D. S. (2000). New advances in GraphHTN: identifying independent subproblems in large HTN domains. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 206–215. AAAI Press.

Penberthy, J. S. and Weld, D. S. (1992). Ucpop: a sound, complete, partial order planner for adl. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 103–114. Morgan Kaufmann Publishers Inc.

Rabideau, G., Knight, R., Chien, S., Fukunaga, A., and Govindjee, A. (1999). Iterative repair planning for spacecraft operations using the ASPEN system. In *Artificial Intelligence, Robotics and Automation in Space*, volume 440, page 99.

Sacerdoti, E. D. (1975). The nonlinear nature of plans. Technical report, Stanford Research Institute.

Sreedharan, S., Srivastava, S., Smith, D., and Kambhampati, S. (2019). Why can't you do that hal? explaining unsolvability of planning tasks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 1422–1430. AAAI Press.

Weld, D. S. (1994). An introduction to least commitment planning. *AI magazine*, 15(4):27–27.

Winikoff, M., Dignum, V., and Dignum, F. (2018). Why bad coffee? explaining agent plans with valuings. In *International Conference on Computer Safety, Reliability, and Security*, pages 521–534. Springer.