Využití formálních gramatik v automatickém plánování – na cestě k sjednocujícímu modelu

Roman Barták

Univerzita Karlova, Matematicko-fyzikální fakulta
Malostranské náměstí 25, 118 00 Praha

bartak@ktiml.mff.cuni.cz

Abstrakt. V práci navrhujeme použít atributové gramatiky jako sjednocující model pro existující modely plánovacích domén. Takový sjednocující model přinese do plánování řadu nových možností využitím existujících technik vytvořených pro formální gramatiky. Můžeme například ověřit, zda plán odpovídá danému doménovému modelu. Můžeme dokonce verifikovat, zda je daný model vnitřně konzistentní, což je problém, který plánovací komunita dosud ani neřešila. Můžeme využít techniky učení se gramatiky z příkladů slov jazyka pro automatické učení doménových modelů, tedy jednu z důležitých schopností autonomních systémů, která je v případě plánování zatím na velmi nízké úrovni. Samozřejmě je možné pomocí gramatik plány přímo generovat, případně naopak lze z pozorované části plánu odvodit, jaké akce budou následovat nebo které byly přehlédnuty. Formální gramatiky tam mohou posloužit jako vhodné médium pro transport technik a znalostí mezi tak vzdálenými komunitami jako je zpracování jazyka a plánování.

Klíčová slova: plánování, modelování, formální gramatiky.

1 Úvod

Plánování neboli uvažování o budoucích akcích je jednou z klíčových součástí umělé inteligence. Nejintenzivnější výzkum probíhá především v oblasti klasického (STRIPS) plánování, které je založené na ploché struktuře akcí provázaných kauzálními vazbami (předchozí akce poskytují předpoklady pro běh dalších akcí). Přes pokrok v efektivitě tzv. doménově nezávislých plánovačů se tyto plánovací systémy jen zřídka používají v aplikacích např. v robotice nebo v počítačových hrách. Zde jsou mnohem populárnější přístupy umožňující kódovat doménově specifickou informaci, která dramaticky zvyšuje efektivitu plánování. Často používaný je model hierarchických sítí úloh (HTN), kde je plánování založené na dekompozici úloh. V článku navrhujeme pro popis plánovacích domén použít abstraktní model atributových gramatik, který vychází z HTN, ale zároveň umožňuje popsat další modely plánovacích domén jako jsou STRIPS či procedurální modely. Popíšeme výhody a možná použití tohoto

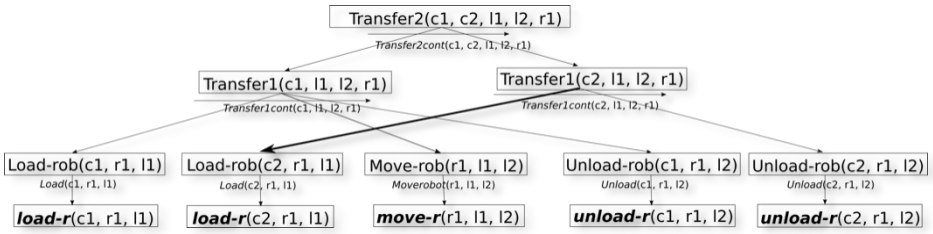
J. Steinberger, M. Zíma, D. Fiala, M. Dostal, M. Nykl (eds.)
Data a znalosti 2017, Plzeň, 5. - 6. října 2017, pp. 99-104.

přístupu a nastíníme výzkumný program vedoucí k tomu, aby navržený přístup mohl sloužit jako sjednocující model pro reprezentaci plánovacích domén.

2 Použité pojmy a existující práce

Plánování se zabývá hledáním posloupnosti akcí vedoucí ke splnění zadaného cíle [8]. Existuje řada plánovacích modelů, od klasického plochého STRIPS modelu, přes hierarchické plánování (HTN) až například po popis plánů formou nedeterministického programu. Klasické STRIPS plánování popisuje akce jako operátory, které mění vlastnosti světa. Akce má své předpoklady, typicky množinu atomických tvrzení, které musí ve světě platit, aby šlo akci aplikovat, a efekty popsané atomy, které budou nebo naopak nebudou platné po provedení akce. Cílem je nalézt posloupnost akcí, která převede daný stav světa na stav, kde platí požadovaná množina atomických tvrzení. Hierarchické plánovaní [6] používá podobné primitivní akce s předpoklady a efekty, ale skupiny akcí jsou sdružovány do úloh tvořících hierarchickou strukturu. Plánování potom spočívá v rozkladu zadané úlohy na podúlohy a dále až na primitivní akce, které lze seřadit tak, že tvoří klasický plán. Hierarchická struktura tak poskytuje návod, jak najít akce pro dosažení zadaného cíle.

Výzkumníci si již dávno všimli podobnosti bezkontextových gramatik a hierarchického plánovacího modelu založeného na popisu plánu jako dekompozice úloh. Formální gramatiky se zatím využily při důkazech složitosti HTN plánovacích problémů [10] a při rozpoznávání plánů [7]. Překvapivě ale zatím nikdo neukázal převod plného HTN modelu na formální gramatiku. Základním problémem je zde prolínání akcí/terminálů vzniklých z různých úloh/neterminálů, které bezkontextové gramatiky nemohou podchytit, viz obrázek 1. K takovému prolnutí dochází, když akce vzniklá z jedné úlohy slouží pro poskytnutí předpokladů akce z jiné úlohy. V uvedeném příkladu byla akce move-r(r1,l1,l2) vytvořena pro úlohu převozu kontejneru c1, ale zároveň lze stejnou akci použít pro přepravu kontejneru c2, který ovšem musí být naložen před přejezdem a vyložen až po něm. Z toho vznikne prolnutí akcí pocházejících z různých úloh.



Obr. 1. Příklad dekompozice úlohy pro převoz dvou kontejnerů na dvě podúlohy převozu jednotlivých kontejnerů a dále na primitivní akce. Tučná čára ukazuje prolnutí dílčích plánů.

Formální gramatiky slouží pro popis jazyků formou přepisovacích pravidel [9]. Jazyk je množina slov skládajících se z písmen, kterým se v gramatikách říká terminální

symboly (terminály). Pro jejich generování se používají pomocné symboly – neterminály. V této práci vycházíme z bezkontextových gramatik, ve kterých mají všechna přepisovací pravidla tvar X → w, kde X je neterminál a w je slovo složené z terminálních i neterminálních symbolů. Generování slova začínající z daného neterminálu lze popsat formou derivačního stromu, který se velmi podobá stromu rozkladu úlohy na podúlohy až na ono prolínání akcí. Pro jeho správné zachycení budeme používat atributové gramatiky [11] přidávající k symbolům atributy, které mohou být provázány podmínkami. Tyto podmínky potom umožní uspořádání akcí, tak aby odpovídalo kauzálním vazbám.

3 Atributové gramatiky jako plánovací model

V práci [3] jsme ukázali, že HTN, klasický STRIPS model a procedurální model [1] lze převést na atributovou gramatiku se specifickou podmínkou *timeline*. Hierarchická struktura se zde používá podobně jako rozklad úlohy na podúlohy případně pro generování sekvence úloh, zatímco *timeline* podmínka zajišťuje platnost kauzálních vazeb mezi primitivními operátory. Stručně řečeno, při rozkladu úlohy shromažďujeme události vážící se na změny platnosti daného atomického tvrzení a *timeline* podmínka potom zajišťuje, že tyto události lze konzistentně uspořádat. Konkrétně, pokud akce požaduje platnost nějakého atomu, tak jiná akce v pořadí před ní musí atom nastavit na platný. Zde je ukázka takového dekompozičního pravidla (množina TL sdružuje události pro *timeline* podmínku a množina I pořadová čísla akcí):

$$\begin{aligned} &\mathit{Transfer1}_{c,l,l1,l2,r}(I,TL) \quad \rightarrow \quad \mathit{Load\text{-}rob}_{c,r,l1}(I_1,TL_1). \\ &\qquad\qquad\qquad\qquad\qquad\quad \mathit{Move\text{-}rob}_{r,l1,l2}(I_2,TL_2). \\ &\qquad\qquad\qquad\qquad\qquad\quad \mathit{Unload\text{-}rob}_{c,r,l2}(I_3,TL_3) \\ &[TL = TL_1 \cup TL_2 \cup TL_3,\ I = I_1 \cup I_2 \cup I_3,\ max(I_1) < min(I_2),\ max(I_2) < min(I_3)]. \end{aligned}$$

Pokud bychom věděli, že v plánu se již vyskytuje akce pro přesun robota mezi místy l1 a l2, která byla například vygenerována z jiné úlohy, nemuseli bychom úlohu $\mathit{Move\text{-}rob}_{r,l1,l2}$ do rozkladu vůbec zahrnovat. V takovém případě je ale potřeba přidat podmínku, že před vyložením kontejneru bude robot na příslušném místě:

$$\begin{aligned} &\mathit{Transfer1}_{c,l,l1,l2,r}(I,TL) \quad \rightarrow \quad \mathit{Load\text{-}rob}_{c,r,l1}(I_1,TL_1). \\ &\qquad\qquad\qquad\qquad\qquad\quad \mathit{Unload\text{-}rob}_{c,r,l2}(I_3,TL_3) \\ &[TL = TL_1 \cup TL_3 \cup \{at(r,l2)@\,min(I_3)\},\ I = I_1 \cup I_3,\ max(I_1) < min(I_3)]. \end{aligned}$$

Pro jednu úlohu tak může existovat více dekompozičních pravidel a při plánování se vyberou takové dekompozice, kterými lze získat korektní posloupnost akcí. O správné uspořádání akcí se stará podmínka *timeline*, která zajišťuje navázání efektů a předpokladů akcí. Předpoklady jsou popisovány pomocí tzv. *before* událostí, zatímco efekty pomocí *after* událostí. Podmínka zajišťuje, že pro každý atomický výrok je *before* událost předcházena příslušnou *after* událostí resp. jinou *before* událostí stejného

typu. Obrázek 2 ukazuje posloupnosti událostí pro jednotlivé atomické výroky z plánu zobrazeném obrázkem 1. Plus index indikuje požadavek na platnost daného výroku (v předpokladu) resp. jeho nastavení (v efektu), mínus index požadavek na neplatnost výroku resp. jeho smazání.

index akce	0	1 load-r(c1, r1, l1)	2 load-r(c2, r1, l1)	3 move-r(r1, l1, l2)	4 unload-r(c1, r1, l2)	5 unload-r(c2, r1, l2)
loaded(r1, c1)	a^-	b^-a^+			b^+a^-	
loaded(r1, c2)	a^-		b^-a^+			b^+a^-
at(r1, l1)	a^+	b^+	b^+	b^+a^-		
at(r1, l2)	a^-			a^+	b^+	b^+
in(c1, l1)	a^+	b^+a^-				
in(c1, l2)	a^-				a^+	
in(c2, l1)	a^+		b^+a^-			
in(c2, l2)	a^-					a^+

Obr. 2. Ukázka vývoje platnosti atomických výroků z plánu na obrázku 1 pomocí before a after událostí. Nultá vrstva popisuje počáteční stav.

4 Výzkumný program

Pokud můžeme převést různé plánovací modely na model jediný reprezentovaný atributovou gramatikou, lze navrhovat algoritmy pracující jen s atributovou gramatikou a řešit tak problémy pro původní modely, což je základní vize použití jednotného modelu. Navíc pro řešení těchto problémů lze používat techniky vyvinuté pro (atributové) gramatiky a tím přirozeně využít existující výsledky v jiné oblasti.

4.1 (Automatické) modelování

Pro modelování plánovacích problémů existuje minimum softwarových nástrojů a uživatelé jsou často odkázáni na textový editor a vytvoření modelu ručně v jazyce PDDL [12]. Při modelování problémů pro HTN model je situace ještě náročnější v nutnosti navrhnout vhodnou hierarchickou strukturu dekompozice úloh. Ideální by tedy bylo, pokud by systém dokázal navrhnout model sám ze sady ukázkových plánů.

V práci [4] jsme ukázali, jak lze rozpoznat akce z dat získaných ze sensorů drona. Jedná se tak o přínos k problému mostu mezi analogovým světem, typickým pro robotiku, a světem symbolickým, typickým pro umělou inteligenci. Otevřenou otázkou je, jak se naučit parametry takových akcí, například že přímý let může být na různou vzdálenost. To má souvislost s učením se předpokladů a efektů akce, tj. jaké vlastnosti musí mít svět, aby šlo akci realizovat, a jak se svět po provedení akce změní. Přirozeně je také potřeba se učit, jak danou akci následně provést. Pokud získáme primitivní akce, je dalším krokem jejich organizace do hierarchické struktury. Zde je možné provádět různé úlohy, pozorovat příslušné posloupnosti akcí, v nich hledat opaku-

jící se sekvence, které lze označit jako úlohy. Zde se vlastě jedná o učení se gramatiky na základě zadaných slov a šlo by tedy použít postupy z formálních gramatik.

4.2 Práce s modely

Máme-li k dispozici model v podobě atributové gramatiky, lze s ním řešit zajímavé problémy, které byly dosud v plánování složité. Prvním z nich je ověření, zda daný plán odpovídá modelu. Pro HTN model zatím existuje jediný výpočtově poměrně náročný přístup převodem na problém Booleovské splnitelnosti [5]. V případě realizace modelu v atributové gramatice je možné použít techniky rozpoznání, zda slovo patří do daného jazyka, například známý CYK algoritmus [13]. Ten je potřeba upravit pro atributové gramatiky, a hlavně vzít v úvahu prolnutí akcí v plánu (obrázek 1). Stejný algoritmus, který je v podstatě založený na sdružování akcí do úloh dle pravidel metodou zdola-nahoru (použití gramatiky analytickým způsobem), lze potom použít na rozpoznání, jakou úlohu agent řeší, analýzou pozorované (i neúplné) posloupnosti akcí. Ještě zajímavější a dosud nikým neřešený problém je otázka vnitřní konzistence doménového modelu. Tato otázka je důležitá zvláště pro automaticky získané modely. V práci [2] jsme navrhli základní přístup pro verifikaci atributové gramatiky s rekurzí, kde atributy mohou nabývat hodnot z konečných domén. Zde je použita technika odvozená od redukce bezkontextové gramatiky. Otevřenou otázkou je, zda lze takto verifikovat také atributové gramatiky s *timeline* podmínkou, které se používají pro plánovací modely.

5 Závěr

Článek představuje vizi jednotného modelu plánovacích problémů použitím atributových gramatik a jejich použití pro řešení existujících i nových problémů kolem plánovacích modelů.

Literatura

1. Baier, J.A., Fritz, Ch. and McIlraith S.A.: Exploiting Procedural Domain Control Knowledge in State-of-the-Art Planners. In: Proc. of the Seventeenth International Conference on Automated Planning and Scheduling (ICAPS 2007). M.S. Boddy, M. Fox, and S. Thiébaux (Eds.). AAAI (2007), 26-33.
2. Barták, R. Dvořák, T.: On verification of workflow and planning domain models using attribute grammars. In: Proc. of the 15th Mexican International Conference on Artificial Intelligence. Springer (2017).
3. Barták, R., Maillard, A.: Attribute Grammars with Set Attributes and Global Constraints as a Unifying Framework for Planning Domain Models. In: Proc. of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS 2017), 45-53.
4. Barták, R., Vomlelová, M.: Using Machine Learning to Identify Activities of a Flying Drone from Sensor Readings, In: Proc. of Thirtieth International Florida Artificial Intelligence Research Society Conference (FLAIRS-30), AAAI (2017), 436-441.

5. Behnke, G., Höller, D., Biundo, S.: This Is a Solution! (... but Is It though?) Verifying Solutions of Hierarchical Planning Problems In: Proc. of the Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS 2017). L. Barbulescu, J.D. Frank, Mausam, S.F. Smith (Eds.). AAAI (2017), 20-28.
6. Erol, K., Hendler, J., Nau, D.S.: Semantics for Hierarchical Task-network Planning. Technical Report. University of Maryland at College Park, USA (1994).
7. Geib, Ch.W., Steedman, M.: On Natural Language Processing and Plan Recognition. In Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), M. Veloso (Ed.), IJCAI (2007). 1612-1617.
8. Ghallab, M., Nau, D.S., Traverso, P.: Automated planning - theory and practice. Elsevier, 2004.
9. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation, Addison-Wesley, 1979.
10. Höller, D., Behnke, G., Bercher, P., Biundo, S.: Language Classification of Hierarchical Planning Problems. In: Proc. of 21st European Conference on Artificial Intelligence (ECAI 2014). T. Schaub, G. Friedrich, and B. O'Sullivan (Eds.), Vol. 263. IOS Press (2014), 447-452.
11. Knuth, D.E.: Semantics of Context-Free Languages. Mathematical Systems Theory 2 (1968): 127-145.
12. McDermott, D.: The planning domain definition language manual. CVC Report 98-003, Yale Computer Science Report 1165, 1998.
13. Younger, D. H.: Recognition and parsing of context-free languages in time n^3. Inform. Control. 10 (2) (1967): 189-208.

Poděkování: Tento článek vznikl díky podpoře projektu GAČR P103-15-19877S.

Annotation:

Exploiting formal grammars in automated planning – towards the unifying model

The paper suggests using attribute grammars as a unifying framework for planning domain models. It is already possible to translate existing STRIPS, HTN, and PDCK models to attribute grammars with the timeline constraint. The paper discusses how the techniques developed for formal grammars can be used to solve problems with domain models, for example to validate if a plan conforms the model and if the model is internally consistent, and to guess which task is being performed by observing (even incomplete) sequence of actions. An interesting challenge is automated learning of the grammar from observed plans.