Vedecká rada Fakulty matematiky, fyziky a informatiky
Univerzity Komenského v Bratislave

RNDr. Michal Čertický

Autoreferát dizertačnej práce

Action Model Learning: Challenges and Techniques

(Učenie sa modelu akcií: výzvy a techniky)

na získanie vedecko-akademickej hodnosti philosophiæ doctor
v odbore doktorandského štúdia:
9.2.1. informatika

Bratislava 2013

Dizertačná práca bola vypracovaná v internej forme doktorandského štúdia na Katedre aplikovanej informatiky Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislave.

Predkladateľ: RNDr. Michal Čertický
Katedra aplikovanej informatiky
Fakulta matematiky, fyziky a informatiky
Univerzita Komenského
Mlynská dolina
842 48 Bratislava

Školiteľ: Doc. PhDr. Ján Šefránek, CSc.
Katedra aplikovanej informatiky,
Fakulta matematiky, fyziky a informatiky
Univerzita Komenského, Bratislava

Oponenti: Prof. Dr. Michal Pěchouček, MSc.
Katedra počítačů,
Fakulta elektrotechnická,
České vysoké učení technické, Praha

Prof. RNDr. Jiří Pospíchal, DrSc.
Ústav aplikovanej informatiky,
Fakulta informatiky a informačných technológií,
Slovenská technická univerzita, Bratislava

Doc. RNDr. Lubomír Popelínský, Ph.D.
Katedra teorie programování,
Fakulta informatiky,
Masarykova univerzita, Brno

Obhajoba dizertačnej práce sa koná dňa: 26. septembra 2013

pred komisiou pre obhajobu dizertačnej práce doktorandského štúdia vymenovanou predsedom spoločnej odborovej komisie, vo vednom odbore 9.2.1. informatika,

na Fakulte matematiky, fyziky a informatiky Univerzity Komenského, Mlynská dolina.

Predseda spoločnej odborovej komisie:
prof. RNDr. Branislav Rovan, PhD.
Fakulta matematiky, fyziky a informatiky
Univerzita Komenského
Mlynská dolina
842 48 Bratislava

1 Úvod

Znalosti o kauzálnych závislostiach v rámci domény, popisujúce podmienky vykonateľnosti a efekty jednotlivých akcií v doméne, predstavujú nutnú podmienku pre plánovanie a cieľavedomé inteligentné správanie či už živých, alebo umelých agentov. Takéto znalosti sa bežne označujú pojmom *model akcií*.

Modely akcií sú v umelých systémoch bežne vytvárané ručne znalcami domény, alebo programátormi. Ich podrobná špecifikácia je však, obzvlášť v prípade zložitejších domén, náročná a zdĺhavá úloha. Okrem toho je občas potrebná ich aktualizácia, ktorá si vyžaduje potenciálne komplikované revízie znalostí a často vedie k chybám.

Automatické generovanie takýchto modelov akcií sa preto v posledných rokoch stalo zaujímavou výskumnou témou. Tento induktívny proces konštrukcie a následného spresňovania modelov akcií na základe agentových *pozorovaní* nazývame jednoducho *učením sa akcií*.

Po roku 2000 bolo navrhnutých niekoľko metód a techník na učenie sa akcií, využívajúcich širokú paletu techník z informatiky a umelej inteligencie. Spomedzi nich by sme ako prvý mali spomenúť algoritmus $SLAF$ [Amir-Chang, 2008], ktorý postupne konštruuje výrokovo-logickú formulu a neskôr ju interpretuje pomocou SAT solvera. Využitím SAT solverov (konkrétne váhovaného Max-SAT solvera) je mu podobný systém $ARMS$ [Yang-Wu-Jiang, 2007], ktorý sa však neučí na základe pozorovania stavov sveta. Akčné modely totiž indukuje spätne z histórie úspešných plánov. Príkladom kompletne deklaratívnej techniky je učenie sa pomocou logického programovania ASP [Balduccini, 2007]. Ako zástupcov techník vzdialenejších od logiky spomenieme učenie sa akcií cez trénovanie perceptrónov [Mourao-Petrick-Steedman, 2010], alebo viacúrovňové greedy prehľadávanie priestoru možných modelov [Zettlemoyer, 2003].

Diverzita týchto moderných metód spôsobuje, že každá z nich je vhodná pre použitie za iných podmienok a v rozličných druhoch domén. Po ich dôkladnej analýze sa nám podarilo odhaliť šesť základných vlastností, resp. výziev, ktoré sa tieto metódy snažia prekonať. Pojednávajú buď o tom, čo všetko sa daným prístupom dokážeme naučiť, alebo o jeho schopnosti vyrovnať sa s rôznymi komplikáciami spôsobenými charakterom domény:

- Učenie sa *pravdepodobnostných* modelov akcií,
- Indukcia *efektov* akcií a zároveň ich *podmienok vykonateľnosti*,
- Učenie sa *podmienených efektov*,
- Použiteľnosť v *čiastočne pozorovateľných* doménach,
- Odolnosť voči *senzorickému šumu* a *zlyhaniam akcií*,
- Dostatočná *rýchlosť* výpočtov (často súvisiaca s použitím *online*, resp. *offline* algoritmov).

V predkladanej dizertačnej práci sa najskôr venujeme týmto základným výzvam a skúmame rôzne spôsoby akými sa s nimi vyrovnávajú jednotlivé techniky. Neskôr ich použijeme pri návrhu a implementácii našich vlastných metód, pričom sa snažíme vyvinúť čo najuniverzálnejšie riešenie. Naše nové metódy následne analyzujeme na teoretickej úrovni, ale aj experimentálne, v rôznych fundamentálne odlišných doménach.

V druhej časti tohoto autoreferátu najskôr zavedieme základnú terminológiu a potrebné pojmy. Neskôr, v časti 3, zhrnieme naše ciele a načrtneme porovnanie našich vlastných metód s existujúcimi alternatívami. Nasledujúca, štvrtá, časť autoreferátu priblíži jednotlivé naše riešenia a výsledky ich teoretickej a experimentálnej analýzy. V ďalších častiach autoreferátu uvádzame zoznam použitej literatúry, zoznam publikovaných prác autora relevantných k tejto dizertačnej práci, ako aj ohlasov na ne.

2 Základné pojmy a označenia

Formálne sa dá model akcií chápať ako dvojica $\langle D, P \rangle$ kde D je symbolický opis dynamiky domény vo vhodnom jazyku a P je pravdepodobnostná funkcia definovaná na prvkoch D. Súčasné prístupy sa vo výbere reprezentačného jazyka alebo štruktúry líšia. Niektoré z nich používajú štandardné akčné jazyky ako STRIPS [Fikes-Nilsson, 1971], PDDL [McDermott et al., 1998], alebo $\mathcal{C}$ [Guinchiglia-Lifschitz, 1998], ktoré sa následne dajú priamo použiť ako vstup pre plánovač. Iné reprezentujú D rôznymi špecializovanými štruktúrami, ktoré napomáhajú samotnému procesu učenia sa, pričom však musia byť neskôr preložené do iného jazyka, kompatibilného s plánovačom.

Opis dynamiky domény D popisuje akým spôsobom dané akcie prevádzajú svet z jedného diskrétneho stavu na iný. Stavy sveta opisujeme pomocou tzv. *fluentov*. Fluentom nazývame buď inštanciovaný atóm prvorádovej logiky f (pozitívny fluent), alebo jeho negáciu $\neg f$ (negatívny fluent). Každý z nich reprezentuje jeden konkrétny príznak danej domény. Množinu všetkých fluentov našej domény označujeme $\mathcal{F}$. Hovoríme, že komplementárnym fluentom k f (označovaným $\overline{f}$) je fluent $\neg f$ a naopak ($\overline{\neg f} = f$). Stavom sveta "s"je potom taká množina fluentov, pre ktorú platí, že z každej dvojice vzájomne komplementárnych fluentov sa v nej nachádza práve jeden. V tomto zmysle je "s" *úplným* a *konzistentným* popisom stavu sveta. Množina všetkých možných stavov sveta je definovaná ako $\mathcal{S} = \{s \mid \forall(f, \neg f) \in \mathcal{F} : (f \in s \wedge \neg f \notin s) \vee (f \notin s \wedge \neg f \in s)\}$. Keďže máme často do činenia s čiastočne pozorovateľnými a zašumenými doménami, potrebujeme od stavu sveta odlíšiť pojem *pozorovania*. Pozorovanie "o" je definované ako akákoľvek neprázdna množina fluentov. Pozorovanie teda nemusí byť (a často ani nie je) úplné, ani konzistentné.

Akcia (resp. inštancia akcie) je v našom prípade definovaná ako dvojica pozostávajúca z mena akcie a usporiadanej n-tice konštantných parametrov, korešpondujúcich k objektom nachádzajúcim sa v našej doméne. Množinu všetkých akcií označujeme ako $\mathcal{A}$. Učenie prebieha na základe spracovania tzv. *príkladov*, teda trojíc (o, a, o'), kde o a o' sú dve za sebou nasledujúce pozorovania a $a \in \mathcal{A}$ je akcia vykonaná medzi nimi. Množina príkladov, ktoré sú použité v procese učenia sa nazýva *trénovacia množina*. Každý z týchto príkladov môže byť považovaný za *pozitívny* alebo *negatívny* vzhľadom na niektorý prvok z D, pričom definície pozitívnych a negatívnych príkladov sa líšia v závislosti od zvoleného reprezentačného jazyka.

Učenie akcií sa potom dá chápať ako algoritmický proces, ktorý na základe pozorovaní modifikuje D tak, aby čo najviac príkladov z trénovacej množiny bolo pozitívnych, a čo najmenej ich bolo negatívnych vzhľadom na prvky z D, pričom môže využívať a modifikovať P.

3 Ciele dizertačnej práce

Primárnym cieľom predkladanej dizertačnej práce je vývoj takých nových metód učenia sa modelov akcií, ktoré by disponovali lepšími vlastnosťami ako súčasné alternatívy. Aby však bolo možné naplniť tento zámer, je nutné zmapovať pomerne neprehľadnú situáciu v tejto mladej výskumnej oblasti, a identifikovať jednotlivé atribúty a výzvy, ktoré sú pre túto problematiku podstatné. Ciele tejto dizertačnej práce sú teda formulované nasledovne:

- Analyzovať existujúce metódy učenia sa akcií a identifikovať množinu podstatných atribútov, resp. výzviev, ktoré sú diskutované v relevantných publikáciách.
- Porovnať tieto metódy a odhaliť ich nedostatky. Kritériami sú spomínané výzvy a aplikovateľnosť jednotlivých metód za rôznych podmienok.
- Navrhnúť vlastné algoritmy a preskúmať ich vlastnosti.
- Experimentálne overiť navrhnuté metódy v rozličných druhoch domén.

Analýza existujúcich metód ukázala, že väčšina z nich sa dokáže vyrovnať len s menšou podmnožinou spomínaných výziev. Tabuľka na obr. 1 znázorňuje prehľadné porovnanie existujúcich alternatív s našimi dvoma metódami, prezentovanými v tejto dizertačnej práci. Vidíme,

Paper	Method name	Partially observable domains	Probabilistic action models	Dealing with action failures and noise	Both preconditions and effects	Conditional effects	Online
[Amir-Chang, 2008]	SLAF	yes	no	only when failure is explicitly known	no	no	yes
[Yang-Wu-Jiang, 2007]	ARMS	yes	no	no	yes	no	no
[Balduccini, 2007]	A-Prolog with ASP semantics + Learning module	yes	no	no	yes	yes	no
[Mourao-Petrick-Steedman, 2010]	Perceptron Algorithm	yes	yes	yes	no	no	yes
[Pasula-Zettlemoyer-Kaelbling, 2007]	Greedy Search	no	yes	yes	yes	yes	no
Chapter 10 or [Čertický, 2012c]	Reactive ASP	yes	no	yes	yes	no	no
Chapter 8 or [Čertický, 2012a] [Čertický, 2012b]	3SG Algorithm	yes	yes	yes	yes	yes	yes

Obr. 1: Porovnanie metód na základe vlastností.

Paper	Method name	Blocks World (fully observable)	Blocks World (partially observable)	SyRoTek Platform	Unreal Tournament 2004
[Amir-Chang, 2008]	SLAF	yes	yes	no	no
[Yang-Wu-Jiang, 2007]	ARMS	yes	yes	no	no
[Balduccini, 2007]	A-Prolog with ASP semantics + Learning module	yes	yes	no	no
[Mourao-Petrick-Steedman, 2010]	Perceptron Algorithm	yes	yes	yes	yes
[Pasula-Zettlemoyer-Kaelbling, 2007]	Greedy Search	yes	no	no	no
Chapter 10 or [Čertický, 2012c]	Reactive ASP	yes	yes	yes	no
Chapter 8 or [Čertický, 2012a] [Čertický, 2012b]	3SG Algorithm	yes	yes	yes	yes

Obr. 2: Porovnanie metód na základe použiteľnosti v rôznych druhoch domén.

že jedna z našich metód (online algoritmus $3SG$) spĺňa všetky vytýčené požiadavky, zatiaľ čo tá druhá (učenie pomocou reaktívneho ASP) len polovicu z nich.

Na obr. 2 zasa vidíme porovnanie všetkých metód na základe kompatibility s jednotlivými doménami. Pre toto porovnanie, ako aj pre následné praktické experimenty, sme zvolili nasledujúce štyry domény: Akčná počítačová hra Unreal Tournament 2004 [Gemrot et al., 2009], reálna robotická platforma SyRoTek [Kulich et al., 2010] a dve verzie domény Blocks World [Nilsson, 1980] (plne a čiastočne pozorovateľná).

Tento výber je odôvodnený ich stupňujúcou sa náročnosťou z hľadiska pozorovateľnosti, náhodnosti a nárokov na rýchlosť výpočtov. Najnáročnejšou z týchto domén je hra Unreal Tournament 2004, v ktorej je použiteľná iba jedna z analyzovaných alternatív a jedna z nami vyvinutých metód.

4 Hlavné výsledky a ich význam

4.1 Online algoritmus $3SG$

Prvá z nových metód, predkladaných v tejto práci, je zložená z dvoch navzájom sa dopĺňajúcich zložiek: *reprezentačnej štruktúry* $\mathcal{EF}$, pomocou ktorej vyjadrujeme model akcií, a online *algoritmu* $3SG$, ktorým sa dokážeme takto reprezentované modely učiť.

4.1.1 Reprezentačná štruktúra $\mathcal{EF}$

Výber vhodnej reprezentačnej štruktúry je doležitou súčasťou tejto metódy. Mala by byť čo najkompaktnejšia, aby bolo možné učenie v reálnom čase pri veľkom množstve pozorovaní, avšak chceme zachovať dostatočnú expresivitu, aby sme dokázali vyjadriť komplexné modely akcií.

Klasickým príkladom reprezentačnej štruktúry, ktorá je použitá napríklad v [Amir-Russel, 2003, Amir-Chang, 2008], je tzv. *tranzičná relácia* $\mathcal{TR} \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{S}$. Tá vyjadruje vzťah medzi dvoma kompletnými popismi stavu sveta a akciou, a je najrobustnejšia vzhľadom na vyjadrovaciu silu, ale aj na priestorovú náročnosť.

Reprezentačná štruktúra, s ktorou pracuje náš algoritmus $3SG$ sa volá *efektová formula* ($\mathcal{EF}$). Na rozdiel od $\mathcal{TR}$, nie je definovaná ako relácia, ale ako konečná množina výrokovo-logických atómov dvoch druhov s rozdielnym významom:

1. a^f znamená že *"akcia* **a** *spôsobuje fluent* **f***"*
2. a^f_c znamená "aby **a** spôsobila **f**, musí platiť **c**" (resp. fluent **c** je podmienkou pre a^f)

Výhodou $\mathcal{EF}$ je kompaktnosť, ktorá ju robí vhodnou pre použitie pri učení sa v reálnom čase. Jej priestorová zložitosť je vyjadrená ako $\mathcal{O}\big(|\mathcal{A}| \cdot (|\mathcal{F}| + |\mathcal{F}|^2)\big)$, čo predstavuje zníženie oproti $\mathcal{O}\big(|\mathcal{A}| \cdot (2^{|\mathcal{F}|/2})^2\big)$ v prípade $\mathcal{TR}$.

Zníženie priestorovej zložitosti $\mathcal{EF}$ je v prvom rade spôsobené priradením implicitného významu absencii niektorých atómov, a taktiež miernym znížením jej expresivity. $\mathcal{EF}$ totiž, na rozdiel od tranzičnej relácie, nedokáže reprezentovať disjunktívne podmienky efektov. Učenie sa efektov s disjunktnými podmienkami je však podstatne ťažší problém a náš algoritmus ho, tak isto ako ostatné súčasné metódy, nerieši. Schopnosť reprezentovať takéto podmienky by preto bola pre nás zbytočná.

4.1.2 Algoritmus a jeho vlastnosti

Algoritmus $3SG$ prijíma na vstupe jeden (najnovší) príklad (o, a, o') a model akcií $\langle \mathcal{EF}, P \rangle$. Na základe tohoto príkladu potom upravuje náš model. Skratka $3SG$ znamená *"Simultaneous Specification, Simplification and Generalization"* a napovedá niečo o štruktúre samotného algoritmu. Ten je zložených z troch zodpovedajúcich častí:

- Ako *generalizáciu* (zovšeobecnenie) chápeme pridávanie atómov typu a^f do $\mathcal{EF}$, po tom, čo odpozorujeme zmenu hodnoty f po vykonaní akcie a. Pridanie týchto prvkov totiž zväčšuje množinu príkladov, ktoré náš model pokrýva.

- *Špecifikáciou* označujeme generovanie atómov typu a_c^f (podmienok), ak vstupný príklad je negatívnym vzhľadom na nejaké už naučené a^f. Podmienky a_c^f totiž zmenšujú počet príkladov pokrytých našim modelom.
- *Simplifikáciou* (zjednodušením) $\mathcal{EF}$ je jednoducho povedané zabúdanie tých prvkov, ktoré sa za nejaký časový interval nestihli dostatočne potvrdiť.

Prvou dôležitou vlastnosťou $3SG$ je, že ide o *online* algoritmus. V prípade učenia sa akcií to znamená, že namiesto celej histórie odpozorovaných príkladov dostáva na vstupe vždy len jeden (najnovší) príklad, ktorý použije na nezvratnú modifikáciu modelu. Nevýhodou online algoritmov oproti ich offline alternatívam je, že ich výsledok závisí od poradia vstupov ktoré dostávajú, a teda ich výsledky môžu byť menej presné. Na druhej strane ale online algoritmy pracujú s podstatne menšími vstupmi, čo ich robí výrazne rýchlejšími a vhodnými pre učenie sa v reálnom čase.

Označme si veľkosť vstupnej $\mathcal{EF}$ ako n a veľkosť väčšieho z dvoch vstupných pozorovaní o, o' ako m. *Časová zložitosť* algoritmu $3SG$ je polynomiálna vzhľadom na veľkosť vstupu. V prípade, že $\mathcal{EF}$ implementujeme ako B-strom, je táto zložitosť konkrétne $\mathcal{O}\left((m^2+n^2)\log n\right)$.

Okrem toho (ako vidíme na obr. 1) sa $3SG$ dokáže učiť pravdepodobnostné modely akcií s podmienkami vykonateľnosti a podmienenými efektami, a to v čiastočne pozorovateľných, zašumených doménach s možnosťou zlyhania akcií.

4.1.3 Experimenty

Naše experimenty v doméne Unreal Tournament 2004, ktorá predstavuje najväčšiu výzvu zo všetkých diskutovaných prostredí (viď obr. 2) ukázali, že veľkosť nášho modelu, reprezentovaného efektovou formulou $\mathcal{EF}$, sa v priebehu času drží na približne rovnakej úrovni. To znamená, že vďaka simplifikácii (zabúdaniu), nám nenarastá veľkosť vstupu, a teda sa reálny čas behu algoritmu nezvyšuje.

Počas jedného 48 minút trvajúceho experimentu sme zhotovili trénovaciu množinu 21733 príkladov, z ktorých každý obsahoval približne 140 odpozorovaných fluentov. Spracovanie tejto množiny nám aj napriek pomerne neefektívnej implementácii algoritmu trvalo menej ako 13 minút. Rýchlosť algoritmu sa teda javí dostatočná pre použitie v relatívne komplexných doménach, náročných na rýchlosť spracovania odpozorovaných príkladov.

Vysoká miera senzorického šumu a časté zlyhania akcií v reálnej robotickej platforme SyRoTek tak isto nepredstavovali závažnejší problém. Rýchlosť učenia a kvalita výsledných modelov, vyjadrená pomocou F-miery [Lewis, 1994, Makhoul et al., 1999, Rijsbergen, 1979], zohľadňujúcej presnosť, tak isto ako aj úplnosť nášho modelu, bola aj v tejto doméne uspokojivá.

4.2 Učenie sa cez reaktívne ASP

Druhá metóda prezentovaná v tejto dizertačnej práci nie je z hľadiska doménovej kompatibility tak univerzálna, ako algoritmus $3SG$ (viď obr. 2). Napriek tomu však má jednu výhodu - ide o jednoducho implementovateľné, kompletne deklaratívne riešenie.

Táto technika je založená na paradigme logického programovania nazývanej Answer Set Programming (ASP) [Gelfond-Lifschitz, 1988, Gelfond-Lifschitz, 1991, Baral, 2003]. Použitie ASP na účely učenia sa akcií nie je novinkou. Ideovo podobný prístup publikoval už M. Balduccini v [Balduccini, 2007].

Touto našou metódou však reagujeme na uvedenie novšieho formalizmu, nazývaného *reaktívne ASP* [Gebser et al., 2011]. Okrem použitia reaktívnej variácie ASP sa od Balducciniho metódy tá naša odlišuje jednoduchšou a kompaktnejšou reprezentačnou štruktúrou. V konečnom dôsledku tak disponuje rozdielnou množinou vlastností, čo ju robí vhodnou pre použitie za iných podmienok (obr. 1 a 2).

4.2.1 Reprezentačná štruktúra $\mathcal{AL}$

V tomto prípade reprezentujeme náš model akcií ako množinu literálov jazyka *A-Prolog*. Jazyk A-Prolog, s nemonotónnou sémantikou navrhnutou Gelfondom a Lifschitzom na prelome 90-tych

rokov pre ASP [Gelfond-Lifschitz, 1988, Gelfond-Lifschitz, 1991], sa úspešne používa na reprezentáciu a usudzovanie o rôznych druhoch znalostí. Nami navrhnuté kódovanie modelu akcií je kompaktnejšie ako kódovanie použité v Balduccinihо metóde, avšak nedajú sa ním vyjadriť podmienené efekty.

Hlavnou myšlienkou za týmto spôsobom reprezentácie je zjednodušujúci predpoklad, že pre ktorúkoľvek dvojicu vzájomne opačných fluentov $(f, \neg f)$ a každú akciu a, musí platiť práve jedno z nasledujúcich tvrdení:

1. a **spôsobuje**, že (pozitívny) fluent f bude platiť,
   (reprezentované literálom jazyka A-Prolog "$causes(a, f)$")
2. a **spôsobuje**, že **opačný** (negatívny) fluent $\neg f$ bude platiť,
   (reprezentované literálom "$causes(a, \neg f)$")
3. alebo a **zanecháva hodnotu** týchto fluentov nezmenenú.
   (reprezentované literálom "$keeps(a, f)$")

Navyše chceme, aby náš model obsahoval informáciu o vykonateľnosti jednotlivých akcií. Z tohoto pohľadu, pre každý fluent f (či už pozitívny, alebo negatívny) a každú akciu a musí platiť práve jedno z nasledujúcich tvrdení:

1. f **je podmienkou** vykonateľnosti akcie a (reprezentované literálom "$pre(a, f)$"),
2. alebo f **nie je podmienkou** vykonateľnosti akcie a ("$\neg pre(a, f)$").

Efekty a podmienky vykonateľnosti jednotlivých akcií teda môžu byť reprezentované množinou takýchto literálov. Navyše, pre akýkoľvek takto reprezentovaný model akcií platí, že jeho priestorová zložitosť je presne $|\mathcal{A}| \cdot |\mathcal{F}|/2 \ + \ |\mathcal{A}| \cdot |\mathcal{F}|$.

4.2.2 Učenie sa s reaktívnym ASP a vlastnosti

Samotné učenie modelov akcií pomocou reaktívneho ASP prebieha nasledovným spôsobom:

1. Spustíme reaktívny ASP solver *iClingo* [Gebser et al., 2011] s nami vytvoreným statickým logickým programom (detailný opis je v dizercačnej práci). Solver *iClingo* je serverová aplikácia, ktorá vypočíta prvý z množstva tzv. *stabilných modelov* tohoto statického logického programu a následne počúva na špecifikovanom porte, čakajúc na updaty od *controller* aplikácie. Takto vypočítaný stabilný model je množina A-Prolog literálov, ktorá v našom prípade obsahuje $\mathcal{AL}$-reprezentáciu jedného možného modelu akcií.
2. *Controller* je klientská aplikácia, ktorá po každom odpozorovanom príklade spracuje dáta zo senzorov agenta, zakóduje ich do A-Prologovskej reprezentácie a pošle ich takto serveru (solveru *iClingo*).
3. Solver následne konfrontuje tento najnovší príklad s vopred vypočítaným stabilným modelom. Vďaka nemonotónnej sémantike ASP môže nový príklad spôsobiť, že model ktorý bol pred tým platný, bude musieť byť zahodený (ak je odpozorovaný príklad negatívny vzhľadom na niektoré z tvrdení, ktoré model obsahuje). Ak sa toto stane, *iClingo* solver vypočíta nový model, ktorý nie je v konflikte so žiadnym z predchádzajúcich pozorovaní (prípadne môžeme povoliť istú mieru tolerancie konfliktov).

Pri každom výpočte nového modelu si solver uloží v pamäti všetky čiastkové výsledky, aby ich mohol recyklovať pri nasledujúcom výpočte. Táto optimalizácia je z praktického hľadiska dôležitá, a dokáže ušetriť veľa času, nakoľko výpočet nového stabilného modelu zahŕňa exponenciálne algoritmy [Simons et al., 2002]. To je jednou z výhod reaktívneho ASP oproti jeho staršej, statickej verzii.

Pri takomto postupe má náš agent k dispozícii vždy jeden model akcií. Ten sa postupom času nahrádza novými, ktoré popisujú reálnu dynamiku domény čoraz presnejšie.

Okrem rozdielov vyplývajúcich z použitia reaktívnej verzie ASP sa naša metóda líši od súvisiaceho Balducciniho prístupu v dvoch podstatných vlastnostiach, vyplývajúcich z rozličného kódovania modelov akcií. Obe metódy síce reprezentujú akcie ako množinu A-Prologovských literálov, avšak s rozdielnym významom. Zatiaľ čo naše kódovanie priamo opisuje vplyv akcií na doménu, Balducciniho literály sú de facto syntaktickým prekladom modelu zapísaného v akčnom jazyku $\mathcal{C}$ [Guinchiglia-Lifschitz, 1998]. To má za následok, že:

1. Naša metóda sa nedokáže učiť podmienené efekty ($\mathcal{AL}$ kódovanie ich nepodporuje). Výsledné modely sú teda menej expresívne.

2. Jednoduchšia, sémanticky-založená reprezentačná štruktúra nám umožňuje jednoducho modifikovať náš systém tak, aby sa vyrovnal so senzorickým šumom a zlyhaním akcií. Predkladaná metóda je teda použiteľná aj v nedeterministických doménach.

4.2.3 Experimenty

Pre naše experimenty sme si v tomto prípade zvolili dve verzie domény Blocks World (plne a čiastočne pozorovateľnú). Blocks World je totiž pomerne často používaný benchmark, ktorý využil aj Balduccini pri experimentoch so svojou metódou. To nám umožnilo porovnať náš prístup s jeho alternatívou.

Aj keď sú obe metódy offline a využívajú exponenciálne algoritmy, výsledky experimentov naznačujú, že náš prístup je (vďaka použitiu reaktívneho ASP a stručnejšej reprezentačnej štruktúry) podstatne rýchlejší.

Okrem toho sa potvrdilo naše očakávanie, že kvalita naučených modelov (vyjadrená F-mierou) stúpa o niečo pomalšie za prítomnosti čiastočných pozorovaní. Tie však neovplyvnili rýchlosť samotných výpočtov.

5 Zoznam použitej literatúry

[Agrawal, 1994] R. Agrawal, R. Srikant. *Fast algorithms for mining association rules.* In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB): 487-499. 1994.

[Agre-Chapman, 1987] P. E. Agre, D. Chapman. *Pengi: an implementation of a theory of activity.* In Proceedings of the Sixth National Conference in Artificial Intelligence (AAAI-87): 268-272. 1987.

[Aizerman-Braverman-Rozoner, 1964] M. A. Aizerman, E. M. Braverman, L. I. Rozoner. *Theoretical foundations of the potential function method in pattern recognition learning.* Automation and Remote Control 25: 821-837. 1964.

[Amir-Chang, 2008] E. Amir, A. Chang. *Learning partially observable deterministic action models.* Journal of Artificial Intelligence Research, Volume 33 Issue 1: 349-402. 2008.

[Amir-Russel, 2003] E. Amir, S. Russel. *Logical Filtering.* In Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI'03). 2003.

[Armijo, 1966] L. Armijo. *Minimization of functions having Lipschitz continuous first partial derivatives.* Pacific Journal of Mathematics, volume 16. 1966.

[Baioletti-Marcugini-Milani, 1998] M. Baioletti, S. Marcugini, A. Milani. *C-SATPlan: A SATPlan-based Tool for Planning with Constraints.* In Proceedings of AIPS-98 Workshop on Planning as Combinatorial Search. 1998.

[Balduccini, 2007] M. Balduccini. *Learning Action Descriptions with A-Prolog: Action Language $\mathcal{C}$*. In Proceedings of Logical Formalizations of Commonsense Reasoning, 2007 AAAI Spring Symposium. 2007.

[Baral, 2003] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press. 2003.

[Bayer-McCreight, 1972] R. Bayer, E. M. McCreight. *Organization and maintenance of large ordered indexes*. Acta Informatica 18. II., Volume 1, Issue 3: 173-189. 1972.

[Bertsekas, 1999] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific. 1999.

[Bishop, 2006] C. M. Bishop. *Pattern Recognition and Machine Learning, 1st edition*. New York, NY : Springer. 2006.

[Blum-Furst, 1997] A. Blum, M. Furst. *Fast Planning Through Planning Graph Analysis*. Artificial Intelligence - Volume 90:281-300. 1997.

[Blum-Langford, 2000] A. Blum, J. Langford. *Probabilistic Planning in the Graphplan Framework*. Lecture Notes in Computer Science. Volume 1809. 2000.

[Borodin-El-Yaniv, 1998] A. Borodin, R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, New York. 1998.

[Buro-Furtak, 2004] M. Buro, T. M. Furtak. *RTS games and real-time AI research*. In Proceedings of the Behavior Representation in Modeling and Simulation Conference (BRIMS). 2004.

[Chang-Lee, 1973] . C. Chang, R. C. T. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press. 1973.

[Cook, 1971] S. A. Cook. *The complexity of theorem-proving procedures*. In proceedings of the third annual ACM symposium on Theory of computing (STOC '71). 1971.

[Čertický, 2008] M. Čertický. *An Architecture for Universal Knowledge-based Agent*. MEi:CogSci 2008 Proceedings. 2008.

[Čertický, 2009] M. Čertický. *An Architecture for Universal Knowledge-based Agent*. Master Thesis, Faculty of Mathematics and Physics, Comenius University, Bratislava. 2009.

[Davis-Putnam, 1960] M. Davis, H. Putnam. *A computing procedure for quantification theory*. Journal of the ACM, 7: 201-215. 1960.

[Deb, 2005] K. Deb. *Multi-objective Optimization*. Search Metodologies (Springer). 2005.

[Domshlak-Hoffman, 2007] C. Domshlak, J. Hoffmann. *Probabilistic planning via heuristic forward search and weighted model counting*. Journal of Artificial Intelligence Research 30:565-620. 2007.

[Eiter et al., 2000] T. Eiter, W. Faber, N. Leone, G. Pfeifer, A. Polleres. *Planning Under Incomplete Knowledge*. Lecture Notes in Computer Science, Volume 1861: 807-821. 2000.

[Eiter et al., 2005] T. Eiter, E. Erdem, M. Fink, and J. Senko. *Updating action domain descriptions*. Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05): 418–423. 2005.

[Farritor-Dubowsky, 2002] S. Farritor, S. Dubowsky. *A Genetic Planning Method and its Application to Planetary Exploration*. ASME Journal of Dynamic Systems, Measurement and Control, 124(4): 698-701. 2002.

[Fikes-Nilsson, 1971] R. E. Fikes, N. Nilsson. *STRIPS: A new approach to the application of theorem proving to problem solving*. Artificial Intelligence 5(2): 189-208. 1971.

[Fox-Long, 2003] M. Fox, D. Long. *PDLL2.1: An Extension to PDLL for Expressing Temporal Planning Domains*. Journal of Artificial Intelligence Research, Volume 20. 2003.

[Freund-Schapire, 1999] Y. Freund, R. Schapire. *Large margin classification using the perceptron algorithm*. Machine learning 37: 277-296. 1999.

[Gebser et al., 2008] M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, S. Thiele. *Engineering an Incremental ASP Solver*. In Proceedings of the 24th International Conference on Logic Programming (ICLP'08). 2008.

[Gebser et al., 2011] M. Gebser, T. Grote, R. Kaminski, and T. Schaub. *Reactive Answer Set Programming*. In Proceedings of 11th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'2011): 54-66. 2011.

[Gelfond-Lifschitz, 1988] M. Gelfond, V. Lifschitz. *The stable model semantics for logic programming*. In Proceedings of ICLP-88: 1070-1080. 1988.

[Gelfond-Lifschitz, 1991] M. Gelfond, V. Lifschitz. *Classical negation in logic programs and disjunctive databases*. New Generation Computing: 365-385. 1991.

[Gemrot et al., 2009] J. Gemrot, R. Kadlec, M. Bída, O. Burkert, R. Píbil, J. Havlíček, L. Zemčák, J. Simlovič, R. Vansa, M. Stolba, T. Plch, C. Brom. *Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents*. Agents for games and simulations: 1–15, Springer-Verlag, Berlin, Heidelberg. 2009.

[Gordon-Kochen, 2007] M. Gordon, M. Kochen. *Recall-precision trade-off: A derivation*. Journal of the American Society for Information Science: 145-151. 2007.

[Guinchiglia-Lifschitz, 1998] E. Guinchiglia, V. Lifschitz. *An Action Language Based on Causal Explanation: Preliminary Report*. In Proceedings of 15th National Conference of Artificial Intelligence (AAAI'98). 1998.

[Gupta-Nau, 1992] N. Gupta, D. S. Nau. *On the Complexity of Blocks-World Planning*. Artificial Intelligence 56(2-3): 223-254. 1992.

[Hopcroft-Motwani-Ullman, 2007] J. E. Hopcroft, R. Motwani, J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, Boston/San Francisco/New York: 368. 2007.

[Hromkovič, 2010] J. Hromkovič. *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer-Verlag. 2010.

[Jensen, 1999] R. M. Jensen. *OBDD-based Universal Planning in Multi-Agent, Non-Deterministic Domains*. Master's Thesis, Technical University of Denmark Lyngby, DK-2800, Department of Automation. 1999.

[Jensen, 2003] R. M. Jensen. *Efficient BDD-Based Planning for Non-Deterministic, Fault-Tolerant, and Adversarial Domains*. School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213. 2003.

[Jensen-Veloso, 2000] R. M. Jensen, M. M. Veloso. *OBDD-based deterministic planning using the UMOP planning framework*. In Proceedings of the AIPS-00 Workshop on Model-Theoretic Approaches to Planning: 26-31. 2000.

[Kambhampati, 2000] S. Kambhampati. *Planning Graph as a (Dynamic) CSP: Exploiting EBL, DDB and other CSP Search Techniques in Graphplan*. Journal of Artificial Intelligence Research. 2000.

[Kautz-Selman, 1992] H. A. Kautz, B. Selman. *Planning as Satisfiability*. In Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI'92): 359-363. 1992.

[Kautz-Selman-Jiang, 1997] H. Kautz, B. Selman, Y. Jiang. *A general stochastic approach to solving problems with hard and soft constraints.* The Satisfiability Problem: Theory and Applications. 1997.

[Kulich et al., 2010] M. Kulich, K. Košnar, J. Chudoba, J. Faigl, L. Přeučil. *On a mobile robotics e-learning system.* In Twentieth European Meeting on Cybernetics and Systems Research: 597–602. 2010.

[Kutluhan-Hendler-Nau, 1994] E. Kutluhan, J. Hendler, D. S. Nau. *HTN planning: Complexity and expressivity.* In AAAI-94 Proceedings. 1994.

[Lee-Lifschitz, 2003] J. Lee, V. Lifschitz. *Describing Additive Fluents in Action Language C+.* In Proceedings of IJCAI-03. 2003.

[Leone et al., 2006] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, F. Scarcello. *The DLV system for knowledge representation and reasoning.* ACM Trans. Comput. Log. 7: 499–562. 2006.

[Lewis, 1994] D. D. Lewis, J. Catlett. *Heterogeneous uncertainty sampling for supervised learning.* In Proceedings of the Eleventh International Conference on Machine Learning:148–156. Morgan Kaufmann. 1994.

[Lifschitz-Turner, 1999] V. Lifschitz, H. Turner. *Representing transition systems by logic programs.* In Proceedings of the 5th International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR-99). 1999.

[Lifschitz, 2002] V. Lifschitz. *Answer Set Programming and Plan Generation.* Artificial Intelligence, vol. 138: 39-54. 2002.

[Littman et al. 1998] M. L. Littman, J. Goldsmith, M. Mundhenk. *The computational complexity of probabilistic planning.* Journal of Artificial Intelligence Research 9:1-36. 1998.

[Lopez-Bacchus, 2003] A. Lopez, F. Bacchus. *Generalizing GraphPlan by Formulating Planning as a CSP.* In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03). 2003.

[Makhoul et al., 1999] J. Makhoul, F. Kubala, R. Schwartz, R. Weischedel. *Performance measures for information extraction.* 1999.

[Marquis, 2000] P. Marquis. *Consequence finding algorithms.* In D. Gabbay, P. Smets (Eds.), Handbook of Defeasible Reasoning and Uncertainty Management Systems, Volume 5: Algorithms for defeasible and uncertain reasoning. Kluwer. 2000.

[McDermott et al., 1998] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, D. Wilkins. *PDDL - The Planning Domain Definition Language.* Draft. Available at http://www.cs.yale.edu/ dvm. 1998.

[McMahan-Gordon, 2005] H. B. McMahan, G. J. Gordon. *Fast exactplanning in Markov decision processes.* In Proceedings of ICAPS. 2005.

[Minsky-Papert, 1969] . M. L. Minsky, S. A. Papert. *Perceptrons.* The MIT Press. 1969.

[Moskewitz et al., 2001] M. W. Moskewitz, C. F. Madigan, Y. Zhao, L. Zhang, S. Malik. *Chaff: engineering an Efficient SAT Solver.* In Proceedings of the 38th Design Automation Conference (DAC'01). 2001.

[Mourao-Petrick-Steedman, 2008] K. Mourao, R. P. A. Petrick, M. Steedman. *Using kernel perceptrons to learn action effects for planning.* In Proceedings of the International Conference on Cognitive Systems (CogSys 2008). 2008.

[Mourao-Petrick-Steedman, 2010] K. Mourao, R. P. A. Petrick, M. Steedman. *Learning action effects in partially observable domains*. In Proceeding of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence. 2010.

[Niemela-Simons-Syrjanen, 2000] I. Niemela, P. Simons, T. Syrjanen. *Smodels: A System for Answer Set Programming*. In Proceedings of the 8th International Workshop on Non-Monotonic Reasoning. 2000.

[Nilsson, 1980] N. J. Nilsson. *Principles of Artificial Intelligence*. Tioga, Palo Alto. 1980.

[Otero-Varela, 2006] R. Otero, M. Varela. *Iaction, a System for Learning Action Descriptions for Planning*. In Proceedings of 16th International Conference on Inductive Logic Programming, ILP 06. 2006.

[Pasula-Zettlemoyer-Kaelbling, 2007] H. M. Pasula, L. S. Zettlemoyer, L. P. Kaelbling. *Learning Symbolic Models of Stochastic Domains*. Journal of Artificial Inteligence Research, Volume 29: 309-352. 2007.

[Pednault, 1989] E. P. D. Pednault. *ADL: exploring the middle ground between STRIPS and the situation calculus*. Proceedings of the first international conference on Principles of knowledge representation and reasoning. 1989.

[Pednault, 1994] E. P. D. Pednault. *ADL and the state-transition model of action*. Journal of Logic and Computation 4:467–512. 1994.

[Rintanen, 2009] J. Rintanen. *Planning and SAT*. Handbook of Satisfiability: 483-504, IOS Press. 2009.

[Rijsbergen, 1979] C. J. V. Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition. 1979.

[Rosenblatt, 1958] F. Rosenblatt. *The perceptron: a probabilistic model for information storage and organisation in the brain*. Psychological Review 65(6): 386-408. 1958.

[Russel-Norvig, 2003] S. Russel, P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall. 2003.

[Sacredoti, 1975] E. D. Sacredoti. *The Nonliner Nature of Plans*. IJCAI'75 Proceedings of the 4th international joint conference on Artificial intelligence - Volume 1. 1975.

[Selman-Kautz-Cohen, 1993] B. Selman, H. Kautz, B. Cohen. *Local search strategies for satisfiability testing*. Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge 26. 1993.

[Simons et al., 2002] P. Simons, I. Niemela, and T. Soininen. *Extending and Implementing the Stable Model Semantics*. Artificial Intelligence, 138(1-2):181-234. 2002.

[Slaney-Thiebaux, 2001] J. Slaney, S. Thiebaux. *Blocks World revisited*. Artificial Intelligence 125: 119-153. 2001.

[Subrahmanian-Zaniolo, 1995] V. S. Subrahmanian, C. Zaniolo. *Relating Stable Models and AI Planning Domains*. Logic Programming, Proceedings of 12th ICLP: 233-246. 1995.

[Tate, 1977] A. Tate. *Generating Project Networks*. IJCAI'77 Proceedings of the 5th international joint conference on Artificial intelligence - Volume 2. 1977.

[Theocharous-Kaelbling, 2003] G. Theocharous, L. Kaelbling. *Approximate planning in POMDPs with macro-actions*. In Proceedings of Advances in Neural Information Processing Systems 16. 2003.

[Wilkins, 1984] D. E. Wilkins. *Domain-Independent Planning Representation and Plan Generation*. Artificial Intelligence - Volume 22, Issue 3. 1984.

[Yang-Wu-Jiang, 2007] Q. Yang, K. Wu, Y. Jiang. *Learning action models from plan examples using weighted MAX-SAT*. Artificial Intelligence, Volume 171: 107-143. 2007.

[Zettlemoyer, 2003] L. S. Zettlemoyer, H. M. Pasula, L. P. Kaelbling. *Learning probabilistic relational planning rules*. MIT TR. 2003.

6 Zoznam relevantných publikovaných prác autora

[Č13] M. Čertický, *Action models and their induction*, Organon F, International Journal of Analytic Philosophy 20 (Supplementary Issue 2) (2013), 206–215.

[Č14b] ———, *Real-time action model learning with online algorithm 3sg*, Submitted to Applied Artificial Intelligence journal (AAI) (2014).

[Č14a] ———, *Conformant planning with static causal laws and negation as failure: Decomposition of the asp approach*, Submitted to International Journal of Artificial Intelligence (IJAI) (2014).

[Č12b] ———, *Action learning with reactive answer set programming: Preliminary report*, Proceedings of The Eighth International Conference on Autonomic and Autonomous Systems (ICAS 2012), 2012.

[Č12a] ———, *3sg algoritmus a učenie sa akčných modelov v reálnom čase*, Kognice a umělý život XII, 2012.

[Č12c] ———, *Učenie sa akčných modelov v reálnom čase pomocou algoritmu 3sg: vlastnosti a experimenty*, Študentská Vedecká Konferencia 2012, FMFI, Univerzita Komenského, 2012.

[Č10a] ———, *Fluent-free action representation with ik-strips planning formalism*, Študentská Vedecká Konferencia 2010, FMFI, Univerzita Komenského, 2010.

[Č10b] ———, *Ik-strips formalism for fluent-free planning with incomplete knowledge*, Technical Reports, Comenius University, Bratislava, 2010.

7 Zoznam ostatných publikovaných prác autora

[Jv13] M. Jakob and M. Čertický, *Simulation testbed for the accelerated development of autonomic mobility systems*, Accepted to International Systems Competition on Autonomic Features and Technologies for Road Traffic Flow Modelling and Control Systems (ARTS), 2013.

[Sv14] M. Stanescu and M. Čertický, *Predicting opponent's production in real-time strategy games with answer set programming*, Submitted to Journal IEEE Transactions on Computational Intelligence and AI in Games (TCIAIG) (2014).

[DČŠV11] V. Dziuban, M. Čertický, J. Šiška, and M. Vince, *Lightweight communication platform for heterogeneous multi-context systems: A preliminary report*, 11th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR), Log-IC workshop, vol. 738, 2011, pp. 39–44.

[Č13] M. Čertický, *Implementing a wall-in building placement in starcraft with declarative programming*, arXiv:1306.4460 [cs.AI] (2013).

[vv13] Martin Čertický and Michal Čertický, *Case-based reasoning for army compositions in real-time strategy games*, Scientific Conference of Young Researchers (SCYR), 2013, pp. 70–73.

[Mv11a] M. Vince M. Čertický, Šiška, *Lcp: A lightweight communication platform for heterogenous multi-context systems*, 6th Workshop on Intelligent and Knowledge oriented Technologies (WIKT), 2011.

[MV11b] V. Dziuban M. Vince, M. Čertický, *Fluent-free action representation with ik-strips planning formalism*, Študentská Vedecká Konferencia 2011, FMFI, Univerzita Komenského, 2011.

[Mv12] A. Šimko J. Šiška M. Čertický, M. Homola, *Inteligentný dom ako heterogénna agentová platforma*, Kognice a umělý život XII, 2012.

[Č10a] M. Čertický, *Ik-strips formalism for fluent-free planning with incomplete knowledge*, Študentská Vedecká Konferencia 2010, FMFI, Univerzita Komenského, 2010.

[Č10b] ———, *Plánovanie na symbolových modeloch sveta so stabilnomodelovou sémantikou*, Kognice a umělý život X, 2010.

[Č11a] ———, *Decomposition trick for planning with answer set semantics*, 6th Workshop on Intelligent and Knowledge oriented Technologies (WIKT), 2011.

[Č11b] ———, *Enhancing asp-based planning by heuristic graph-search techniques*, Technical Reports, Comenius University, Bratislava, 2011.

[Mv10] K. Rebrová M. Čertický, J. Frank, *Skúmanie funkcie obsahu pri pragmatickej inferencii s využitím sociálnych sietí*, Kognice a umělý život X, 2010.

8 Ohlasy na publikované práce autora

[OSU+13] Santiago Ontanón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, and Mike Preuss, *A survey of real-time strategy game ai research and competition in starcraft*, Journal IEEE Transactions on Computational Intelligence and AI in Games (TCIAIG) (2013).

[TST+12] Kouhei Takada, Yoshitaka Sakurai, Setsuo Tsuruta, Ernesto Damiani, Valerio Bellandi, Marco Anisetti, and Paolo Ceravolo, *An efficient language pipeline for flexible rule-based context representation*, Journal of Ambient Intelligence and Humanized Computing (2012), 1–12.

[Tv12] Martin Takáč and Ján Šefránek, *Semantics of distinguishing criteria: from subjective to intersubjective*, Interdisciplinary Description of Complex Systems - scientific journal **10** (2012), no. 3, 248–269.

[Ben11] L. Benc, *Terrasim: Simulated environment for learning agents*, Univerzita Komenského, Bratislava, 2011.

[Cie10] P. Cieker, *Using cuda technology for stable model generation of logic programs*, Univerzita Komenského, Bratislava, 2010.

[Lit13] L. Litvaj, *Učenie sa dynamiky domény a plánovanie v reálnom čase*, Univerzita Komenského, Bratislava, 2013.

9 Summary

Knowledge about domain dynamics, describing how certain actions affect the world, is called an *action model* and constitutes the essential requirement for planning and goal-oriented intelligent behaviour. Hand-writing these models, especially in complex domains, is often a time consuming and error-prone task. Alternative approach is letting the agents learn action models from their own observations.

Action learning, as a process of automatic construction and subsequent modification of action models based on the observations, has become a hot research topic in recent years, and various methods employing a wide variety of AI tools have been developed. Diversity of these methods renders each of them usable under different conditions and in various domains.

We have identified a common collection of important properties and used them as a basis for the comprehensive comparison and as a set of challenges that we were trying to overcome when designing our new methods.

First of them, an online algorithm called *3SG* (Simultaneous Specification, Simplification and Generalization), can be used to learn the *probabilistic* action models with *conditional effects* in the presence of *incomplete observations*, sensoric *noise*, and action *failures*, while keeping the computational *complexity low* (polynomial w.r.t. the size of the input), thus managing to cope with all the appointed challenges. Our experiments, conducted in two fundamentally different domains (action computer game Unreal Tournament 2004 and a real-world robotic platform SyRoTek), demonstrate its usability in computationally intensive scenarios, as well as in the presence of action failures and significant sensoric noise.

Our second method, based on a certain paradigm of logic programming called Reactive ASP (Answer Set Programming), is less versatile. It cannot learn the *probabilistic* action models or the *conditional effects* and it employs an exponential algorithm in the learning process. However, even despite being able to deal with only half of the challenges, this method may still prove practical in some situations because of its purely declarative nature. Our experiments conducted in two variants of Blocks World domain demonstrate not only its ability to deal with nondeterminism, but also the noticeable improvement in computation times, compared to other ASP-based approaches.