01 - Úvod, historie

čtvrtek 19. září 2024 14:49

Umělé = člověkem vytvořený artefakt (umělá hmota, umělý sníh, umělý kloub…)

- existuje nějaká přirozená věc, kterou je možno duplikovat
- existuje záměr člověka vytvořit duplikát oné přirozené věci
- došlo k provedení záměru

Inteligence:

- všeobecná schopnost individua **vědomě orientovat vlastní myšlení** na nové požadavky, je to všeobecná duchovní schopnost přizpůsobit se novým životním úkolům a podmínkám. (W. Stern)
- vnitřně členitá a zároveň globální **schopnost individua účelně jednat, rozumně myslet** a efektivně se vyrovnávat se svým okolím. (D. Wechsler)
- **schopnost zpracovávat informace**. Informacemi je třeba chápat všechny dojmy, které člověk vnímá. (J. P. Guilford)

Druhy inteligence

- Abstraktní
  - schopnost řešit dobře definované akademické problémy
  - jednoznačná odpověď
- Praktická
  - schopnost řešit problémy každodenního života
  - nejednoznačné zadání i řešení
- Sociální
  - Schopnost pohybovat se v sociálním prostředí
- Emoční

	…jako lidé	…racionálně
Myslet…	kognitivní modelování	logika
Jednat…	Turingův test	racionální agenty

Silná vs Slabá AI

- **Silná** - Povaha mysli je algoritmická, přičemž není podstatné, v jakém médiu (mozek, počítač) jsou algoritmy implementovány
- **Slabá** - Modelování dílčích projevů mysli (např. schopnosti usuzovat nebo řešit problémy).

Klasická vs Distribuovaná vs Nová AI

- **Klasická** - chápe inteligenci jako atribut jedné mysli
- **Distribuovaná** - chápe inteligenci jako produkt sociálních interakcí více myslí
- **Nová** - chápe inteligenci jako emergentní výsledek činnosti primitivních entit

02 - Vyhodnocování inteligence umělých systémů

čtvrtek 26. září 2024 14:30

Umí AI myslet a jak to poznat?

Slabá AI - pro poznání lidské mysli, umožnuje simulovat mentální schopnosti
Specifická AI - AI jako tvorba programu pro řešení specifických úloh
Silná AI - má rozumění a další kognitivní stavy, snaha vytvořit stroj srovnatelný s člověkem
Obecná AI - tvorba programů pro obecné řešení úloh a obecné inteligentní jednání

Rané Descartes

- Metodologický skepticizmus
  - Skrze pochybování k pevným principům vědění
- Racionalizmus x Empirizmus
  - rozum jako rozhodující zdroj vědění
  - smysly klamou
- Dualizmus
  - nemateriální mysl
  - materiální tělo a svět

- Publikace "Rozprava o metodě"
  - Rozporuje schopnost strojů schopnost rozumné řeči
  - Strojům chybí univerzálnost myšlení
    - Určité věci umí dobře, možná i lépe než člověk. Na jiných však pohoří.

Turing (článek + přednáška)

- Mohou stroje myslet?
- Test "imitační hra" (také "turingův test")
- Otázka strojového učení by měla nahradit otázka turingova testu
- Imitační hra
  - Mohou stroje myslet?
- Stroje zapojené do imitační hry
  - Co můžeme považovat za stroj?
  - Turing se nakonec omezuje na digitální počítače.
    - Paměť - ukládání informací
      - Paměť by teoreticky měla být nekonečná
    - Výkonná jednotka - provádění jednotlivých operací
    - Řídící jednotka
      - Zajišťuje správné provedení instrukcí
  - Instrukce
    - Typické instrukce
      - Sečti X a Y a ulož na pozici Z
    - Speciální instrukce
      - Cykly
      - Podmínky
      - Odkazování na instrukce v paměti
    - Instrukce je stroji předávána v číselné podobě
  - Stroje s diskrétním stavem
    - Turing uznává, že takové stroje v realitě neexistují
- Námitky
  - Teologická námitka
    - "myšlení je funkce lidské neumírající duše"
  - "strkání hlavy do písku"
    - Myslící stroje? To by bylo strašné! Stroje tedy nemyslí.

    - °argument z extrasenzorického vnímání
- Možné důsledky úspěchu ve hře
    - Stroj je srovnatelný s člověkem
    - Stroj je lepší než skutečný hráč

Myšlenkový experiment s čínským pokojem

- Jaké by bylo, kdyby mysl fungovala jako program?
- Člověk neumí čínsky, je zavřený v pokoji s knihou s instrukcemi
    - Zvenku vypadá jako instance programu pro čínštinu
      Zevnitř searl čínštině nerozumí, jen manipuluje se znaky
- => Searl se nenaučil čínsky -> ani stroj čínsky neumí

Agent -> interakce => prostředí

Interakce agenta s prostředím - Pozorování -> akce -> odměna

Agent (pi) -

- fce. Dostane interakční sekvenci -> namapuje na následující akci
- Podmíněná pravděpodobnostní míra nad prostorem akcí

Prostředí (mí)

- Fce. Přiřazující historii interakcí nějakou odměnu a pozorování
- Vyčíslitelná podmíněná pravděpodobnostní míra nad prostorem odměn a pozorování

03 - Řešení úloh ve stavovém prostoru

čtvrtek 3. října 2024 14:30

Máme:
- Definici problému
- Definici cíle
- Akce, které můžeme provádět

Př. Hanojské věže

Cíl:
- Přesunout disky z 1. tyčky na 3. tyčku

Pravidla:
- Lze přemístit jen 1 disk
- Větší nelze na menší

Akce:
- Přesun z tyčky X na tyčku Y

Obecná abstraktní reprezentace
- Generalizace - zobecnění skupiny problémů→ nalezení společných rysů
- Abstrakce - zjednodušení zanedbáním poddružných detailů
- Stavový prostor
  - abstraktní model úlohy a postupu jejího řešení
  - stavy - odpovídají situacím, které mohou dostat
  - Akce - přechody mezi stavy
  - Definován:
    - explicitně (mapová navigace)
    - implicitně (šachy)
- Máme tedy:
  - Množinu stavů S={s}
  - Množinu přechodů ϕ
  - Počáteční stav S0
  - Množina koncových stavů G={g}
- Řešením úlohy je sekvence přechodů (akcí) začínajících v počátečním stavu, vedoucích ke koncovému stavu
- Optimální sekvence má nejnižší cenu cesty

Algoritmy prohledávání
- Algoritmy se liší ve způsobu, kterým volí uzel, který má být rozvíjen v daném kroku.
- Založené na opakované expanzi zvoleného uzlu
  - Rozvinutí = aplikace všech použitelných operátorů
- Algo používají dva seznamy
  - Seznam rozvinutých uzlů (closed)
  - Seznam nerozvinutých uzlů (fringe)
- Pro každý uzel udržuji tyto info:
  - Stav odpovídající uzlu
  - Rodiče - uzel který byl rozvinut
  - Akci - operátor, který byl aplikován na rodiče
  - Cenu kroku
  - Cena cesty

Slepé algoritmy
- Pracují jen se stavovým prostorem, následnící rozvinutého uzlu jsou považováni za rovnocenné

- Algoritmy:
  - BFS - FIFO
    - máme jistotu, že vždy nalezneme koncový stav
    - musíme projít hodně uzlů
      - procházíme všechny, které mají menší hloubku než koncový
  - DFS - LIFO
    - Nemusíme najít koncový uzel - Problém "nekonečné" větve
    - V reálných využitích často ale výhodnější
  - Omezené do hloubky
    - Hledáme řešení jen do nějaké hloubky (procházet můžeme i hlouběji)
  - Iterativní prohlubování - (DFS)
    - Postupně zvětšujeme hloubku
    - opakované omezené prohledávání do hloubky s rostoucí hloubkou
  - Obousměrné prohledávání - (BFS)
    - Stavový prostor musí být explicitně zadán
    - Hledám od počátečního i koncového stavu
      - Od počátečního - klasický algo
      - Od konce
        - rozvinutí koncového stavu - nalezení všech předchůdců
    - Pokud nás obě techniky přivedou do stejného stavu - nalezl jsem cestu
    - Dvě úlohy s poloviční složitostí - menší složitost než jedna úloha s normální složitostí
  - Se stejnoměrnou cenou - (BFS)
    - Řeší číselné ohodnocení přechodů
    - Fringe uspořádáno podle path-cost
      - v každém kroku přerovnám
    - V případě stejných hodnot cen kroku se jedná o prosté BFS
    - Algo končí, až se koncový uzel dostane na první místo ve fringe

- Úplnost: algoritmus nalezne řešení, pokud existuje
- Optimalita: algoritmus nalezne optimální řešení (ve smyslu nejnižší ceny cesty)
- Časová složitost: počet expandovaných uzlů (odpovídá počtu kroků)
- Prostorová složitost: velikost potřebné paměti

algoritmus	úplnost	čas	prostor	optimalita
do šířky	ano	$O(b^d)$	$O(b^d)$	ano
do hloubky	ne	$O(b^m)$	$O(bm)$	ne
stejnoměrná cena	ano	$O(b^{(C/\varepsilon)})$	$O(b^{(C/\varepsilon)})$	ano
omezené do hloubky l	ano pro $l \geq d$	$O(b^l)$	$O(bl)$	ne
iterativní prohlubování	ano	$O(b^d)$	$O(bd)$	ano
obousměrné	ano	$O(b^{d/2})$	$O(b^{d/2})$	ano

Kde
b je větvící faktor
d je hloubka nejlepšího řešení
m je hloubka stavového prostoru
C je cena optimálního řešení

Pro malé stavové prostory s jedním koncovým stavem najdou všechny algoritmy stejné řešení !!!

Informované (Heuristické) algoritmy

- používají heuristickou fci. - číslo vyjadřující kvalitu uzlu z hlediska řešení úlohy
  - lze chápat jako odhad ceny, kterou budu potřebovat z daného uzlu do cíle
  - Pro jednu úlohu můžeme mít víc fcí.
    - Říkáme že lepší fce je informovanější
  - Př. autoavigace - nejkratší a nejrychlejší cesta
- Algoritmy:
  - Paprskové (beam search) - (BFS)
    - „paralelní“ heuristická varianta prohledávání do šířky
    - rozvinou se všechny uzly dané hloubky
    - do fringe se zařadí jen k nejlepších

        - Problém BFS - velikost bobtnající fronty
        - Tato metoda má zafixovanou maximální velikost fronty
    - Uspořádané (greedy best-first search) - (BFS)
        - Heuristická varianta prohledávání do šířky
        - Rozvineme uzel, zařadíme nalezené uzly
        - Seřadíme celý fringe podle hodnoty heuristické fce
    - A*
        - každý uzel S hodnotíme součtem:
            - g(s) - známé ceny cesty od počátku do S
            - h(s) - heuristickou fcí. odhadující cenu z S do koncového uzlu
        - Do seznamu closed přidáme i uzel, který tam již je, ale s vyšší hodnotu g(s)
        - fringe uspořádáme podle f(s)
    - Gradientní (hill-climbing s pamětí) (DFS)
        - následníky rozvíjeného uzlu seřadíme vzestupně podle hodnoty heurist. fce a zařadíme do fringe

- Úplnost: algoritmus nalezne řešení, pokud existuje
- Optimalita: algoritmus nalezne optimální řešení (ve smyslu nejnižší hodnoty f(s))
- Časová složitost: počet expandovaných uzlů (odpovídá počtu kroků)
- Prostorová složitost: velikost potřebné paměti

algoritmus	úplnost	čas	prostor	optimalita
Paprskové s šířkou k	ne	O(kd)	O(k)	ne
Best-first (greedy)	ano	$O(b^d)$	$O(b^d)$	ne
A^*	ano	exponenciální	všechny uzly v paměti	ano
Hill climbing	ano	$O(b^d)$	O(bd)	ne
Lokální Hill climbing	ne	O(d)	O(b)	ne

Kde
b je větvící faktor
d je hloubka nejlepšího řešení
m je hloubka stavového prostoru

Náhodné algoritmy
- vybírají uzly náhodně
- Algoritmy:
    - genetické algoritmy

Simulované žíhání
- Samostatná technika prohledávání
- můžeme použít pro řešení problému uváznutí v lokálním extrému u gradientního prohledávání.
- inspirovaná žíháním kovových slitin k získání jejich optimálních vlastností
    - materiály se zahřívají a pak pomalu ochlazují
- Princip:
    - **Inicializace:**
        - Zvolí se počáteční řešení problému.
        - Nastaví se počáteční "teplota" T, která určuje pravděpodobnost přijetí horších řešení.
    - **Iterační proces:**
        - **Generování nového řešení:**
            - V okolí aktuálního řešení se náhodně vybere nové řešení.
        - **Hodnocení řešení:**
            - Spočítá se změna kvality (např. energie) ΔE mezi novým a aktuálním řešením.
        - **Rozhodnutí o přijetí:**
            - **Pokud je nové řešení lepší** (ΔE>0), přijme se.
            - **Pokud je nové řešení horší** (ΔE<0), přijme se s pravděpodobností P=e^(−ΔE/T).

        - **Snížení teploty:**
            - Teplota T se postupně snižuje podle zvoleného chladicího schématu (např. geometrickou řadou Tnové=αT, kde 0<α<1).
    - **Koncové podmínky:**
        - Proces pokračuje, dokud není dosaženo koncové teploty nebo maximálního počtu iterací.
- **Proč přijímat horší řešení?**
  Přijetí horších řešení s určitou pravděpodobností umožňuje algoritmu překonat lokální minima a prozkoumat globální prostor řešení. To zvyšuje šanci na nalezení globálního optima.
- **Shrnutí:**
  Simulované žíhání je efektivní metoda pro hledání přibližných řešení složitých problémů, kde tradiční deterministické metody selhávají nebo jsou výpočetně náročné.

General Problem Solver

- algoritmus pro řešení úloh metodou postupného rozkladu úlohy na podúlohy
    - na základě výpočtu diferencí mezi aktuálním a cílovým stavem
- tři základní procedury:
    - TRANSFORM
    - REDUCE
    - APPLY

Další systémy pro řešení úloh

- STRIPS
    - následník systému GPS
    - stavy úlohy popsány formulemi predikátové logiky
- Planner
    - rozlišuje tvrzení a teorémy
- SOAR

04 - Splňování podmínek

čtvrtek 17. října 2024 14:36

Úloha je tvořena:

- konečnou množinou proměnných, popisujících stav světa
- definičním oborem pro každou proměnnou
- podmínkami, které musí hledané řešení splnit

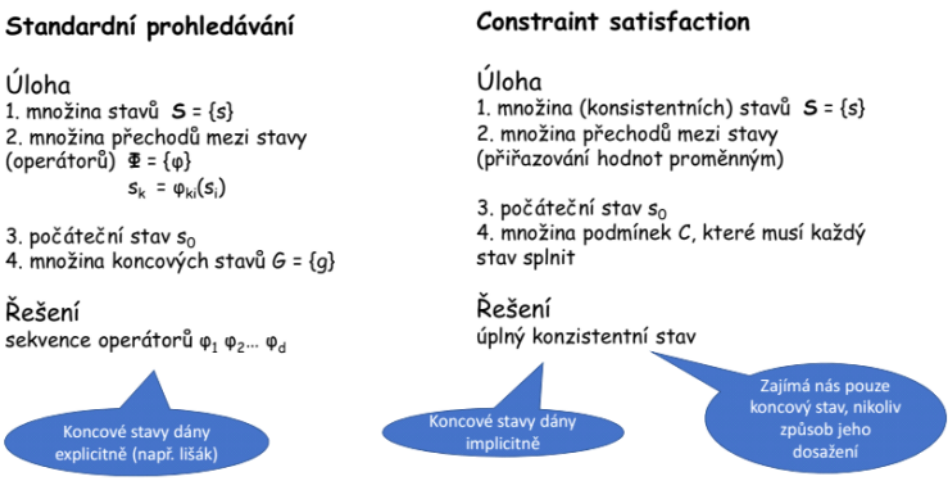
- řešení jsou pak takové hodnoty proměnných, aby všechny podmínky byly splněny
- př. - sudoku, 8 queens, barvení mapy, Einsteinova zebra, problém obchodního cestujícího
- reálnější úlohy - rozvrhování výroby, tvorba rozvrhů

Způsoby řešení

- Matematické metody
  - Operační výzkum
    - snaha najít optimální řešení daného problému při daných omezeních
    - používá se matematické modelování (rovnice, nerovnice apod.)

- Booleovská splnitelnost
  - → převod podmínek na logické formule
  - řešíme otázku nalezení interpretace (dosazení konstant za proměnné)
    - tak aby formule byla pravdivá
  - Metody:
    - DPLL algoritmus: slepé prohledávání do hloubky
    - Tablová metoda
    - Rezoluční princip

- Využití stavového prostoru
  - Stav: částečné přiřazení hodnot proměnným
  - Konzistentní stav: stav, který neporušuje žádnou podmínku
  - Úplný stav: stav u kterého jsou přiřazeny hodnoty všem proměnným
  - Koncový stav: stav, který je úplný a konzistentní



05 - Teorie her

čtvrtek 24. října 2024 14:27

Matematická teorie her
- disciplína aplikované matematiky
- analyzuje široké spektrum konfliktních rozhodovacích situací, které mohou nastat kdekoliv, kde dochází ke střetu zájmů.

Základní úlohy
- Které rozhodnutí (strategie) je optimální ?
- Jak nalézt optimální strategii ?

Předpoklad racionality
- každý hráč chce maximalizovat svůj zisk (vyhrát)

Typy her
- podle počtu hráčů
- podle počtu strategií
- podle typu výhry - hry s konstantím součtem / hry s nekonstantním součtem
- podle počte tahů - hrači hrají najednou / střídají se po tazích
- podle míry informace - s úplnou informací (šachy) / s neúplnou informací (karty)
- podle možnosti spolupráce
- deterministické / s prvkem náhody

Hra je formálně popsaná:
- množinou hráčů
- množinami strategií
- výhrami i-tého hráče při použití jednotlivých strategií

Hra jako úloha prohledávání
- Úloha je tvořena:
  - Počátečním stavem
  - Funkcí, která přiřazuje stavům následníky
  - Testem, zda hra skončila
  - Užitkovou funkcí pro skončenou hru (výhra, remíza, prohra)
- Řešení je tvořeno sekvencí akcí vedoucích k dosažení cíle

MinMax strategie
- každý hráč volí takový tah, aby následný nejlepší tah protihráče byl pro něj nejméně nebezpečný

Alfa-beta prožezávání
- vylepšení minmax algoritmu
  - neprocházíme celý strom; větve které nejsou perspektivní odřízneme
- Využívá dvě proměnné:
  - alfa - nejlepší hodnota, kterou může maximalizující hráč zaručeně dosáhnout
  - beta - nejlepší hodnota, které může minimalizující hráč dosáhnout
  - → pokud algoritmus zjistí, že hodnota uzlu je horší než aktuální alfa nebo beta, danou větev už neprochází

- princip:
  - **Prohledávání stromu:** Alfa-beta algoritmus prohledává herní strom stejně jako minimax, ale zároveň si udržuje α a β, aby rozhodl, kdy přestat procházet další uzly.
  - **Maximalizující hráč (např. X):**
    - Snaží se maximalizovat hodnotu.

        - Pokud zjistí, že hodnota uzlu≥β, ukončí procházení této větve, protože minimalizující hráč (O) by tuto větev stejně nezvolil.
    - **Minimalizující hráč (např. O):**
        - Snaží se minimalizovat hodnotu.
        - Pokud zjistí, že hodnota uzlu≤α, ukončí procházení této větve, protože maximalizující hráč (X) by tuto větev stejně nezvolil.
    - **Prořezávání větví:**
        - Pokud α≥β, větev stromu je prořezána (tj. už není potřeba ji dále zkoumat).

06 - Plánování (a rozvrhování)

čtvrtek 31. října 2024 14:31

Plánování <--> rozvrhování <--> provádění

Plánování
- rozhodovací proces - stanovení cílů, výběr prostředků, způsob jejich dosažení
- Jaké akce jsou potřeba pro dosažení cílů

Rozvrhování
- proces rozdělení úloh na zdroje → minimalizace času vytíženosti zdrojů
- Rozvrhování jak zpracovat vzhledem k omezeným zdrojům a času

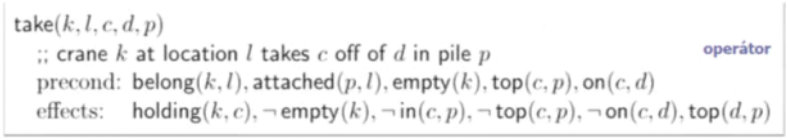
Úloha plánování
- Je tvořena:
  - Počátečním stavem světa
  - Popisem akcí schopných měnit stav světa
  - Požadovaným stavem světa
- Řešením je seznam (sekvence) akcí - plán
  - Bez ohledu na čas a zdroje
- Zabývá se kauzálními vztahy mezi akcemi a otázkou výběru akcí

Úloha rozvrhování
- Je tvořena:
  - Skupinou částečně uspořádaných aktivit
  - Dostupnými zdroji
- Řešením je alokace aktivit na zdroje a v čase
- Soustředí se na alokaci naplánovaných úloh

Plánování → Rozvrhování
- plánovaní řeší hlavně kauzální vztahy mezi akcemi a otázkou výběru akcí
- rozvrhování řeší alokaci naplánovaných akcí v čase a prostoru

- někdy je dobrý obě úlohy řešit najednou
  - třeba když existuje hodně plánů, ale jen málo z nich má přípustný rozvrh

Konceptuální model plánování
- Systém Σ modelující stavy a přechody:
  - Množina stavů S
  - Množina akcí A
  - Množina událostí E
  - přechodová funkce y: SxAxE -> P(S)
- Účelem plánování je zjistit jaké akce a na které stavy se mají aplikovat, abychom z dané situace dosáhli požadovaných cílů

Plánování jako booleovská splnitelnost
- 

- take(crane1,loc1,c3,c1,p1) — akce
  ;; crane crane1 at location loc1 takes c3 off c1 in pile p1
  precond: belong(crane1,loc1), attached(p1,loc1), empty(crane1), top(c3,p1), on(c3,c1)
  effects: holding(crane1,c3), ¬empty(crane1), ¬in(c3,p1), ¬top(c3,p1), ¬on(c3,c1), top(c1,p1)

Plánování jako splňování podmínek

- **Splňování podmínek**

  Úloha
  1. množina (konsistentních) stavů $S = \{s\}$
  2. množina přechodů mezi stavy (přiřazování hodnot proměnným)

  3. počáteční stav s_0
  4. množina podmínek C, které musí každý stav splnit

  Řešení
  úplný konzistentní stav

  Koncové stavy dány implicitně

  Zajímá nás pouze koncový stav, nikoliv způsob jeho dosažení

- **Plánování**

  Úloha
  1. množina stavů S
  2. množina akcí A

  3. počáteční stav s_0
  4. charakterizace cílového stavu

  Řešení
  Sekvence akcí vedoucích k dosažení cíle

  Koncové stavy dány explicitně i implicitně

  Zajímá nás způsob dosažení cíle

Plánování jako prohledávání

- **Standardní prohledávání**

  Úloha
  1. množina stavů $S = \{s\}$
  2. množina přechodů mezi stavy (operátorů) $\Phi = \{\varphi\}$
  $s_k = \varphi_{ki}(s_i)$

  3. počáteční stav s_0
  4. množina koncových stavů $G = \{g\}$

  Řešení
  sekvence operátorů $\varphi_1\ \varphi_2 \ldots\ \varphi_d$

  Koncové stavy dány explicitně

- **Plánování**

  Úloha
  1. množina stavů S
  2. množina akcí A

  3. počáteční stav s_0
  4. charakterizace cílového stavu

  Řešení
  Sekvence akcí vedoucích k dosažení cíle

  Koncové stavy dány explicitně i implicitně

Plánování ve stavovém prostoru

- uzly odpovídají stavům
- hrany odpovídají přechodům mezi stavy pomocí akcí
- cílem je nalézt cestu z počátečního stavu do některého z koncových

- Dopředné prohledávání ve stavovém prostoru
  - Začínáme v počátečním stavu a jdeme k některému stavu cílovému
  - Je potřeba umět:
    - rozhodnout, zda je daný stav cílový nebo ne
    - najít množinu akcí aplikovatelných na daný stav
    - vypočítat stav, do kterého se dostaneme aplikací akce

- Zpětné prohledávání ve stavovém prostoru
  - začínáme cílem a jdeme přes podcíle k počátečnímu stavu
  - Je potřeba umět:
    - zjistit, zda daný stav odpovídá cíli

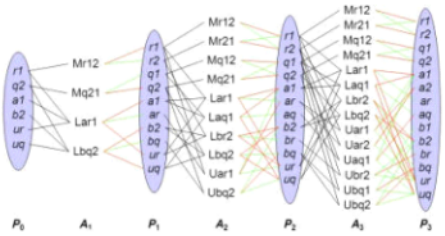
- pro daný cíl najít relevantní akce
- vypočítat podcíl umožňující aplikovat danou akci

Plánování v prostoru plánů

- uzly odpovídají částečným (částečně instanciovaným) plánům
- hrany odpovídají přechodům mezi částečnými plány
- cílem je nalézt úplný plán

Plánovací graf

- orientovaný vrstvený graf pro plánování ve stavovém prostoru
  - Stavová vrstva S_i (P_i) obsahuje atomy instanciované v čase i
  - Akční vrstva A_i obsahuje akce, jejichž předpoklady mohou být splněny v čase i
  - Stavové a akční vrstvy se střídají
  - Nultá vrstva popisuje počáteční stav
-
  - Následuje vrstva popisující akce použitelné na počáteční stav
  - Poslední vrstva je stavová
  - Hrany vedou z atomů do akce, pro kterou tyto atomy tvoří předpoklady
  - Hrany vedou z akce do atomů popisujících efekty dané akce



- Plánovací algoritmus založený na plánovacím grafu.
  - Střídá se fáze expanze grafu a extrakce plánu
    - Expanze
      - □ Nejprve vytvoří plánovací graf až do vrstvy, kde jsou všechny cílové atomy a žádná dvojice se vzájemně nevylučuje
      - □ Pokud fáze extrakce neuspěje, přidá další vrstvu
    - Extrakce
      - □ Z plánovacího grafu se pokusí vybrat plán vedoucí k cílovým atomům

07 - Strojové učení

čtvrtek 14. listopadu 2024 14:37

Metody učení:

- učení se zapamatováním
- učení se z instrukcí
- učení se z analogie
- učení se na základě vysvětlení
- učení se z příkladů
- učení se pozorováním a objevováním

Zpětná vazba v procesu učení

- příklady jsou klasifikované (učení s učitelem)
- malý počet příkladů klasifikovaných, zbytek neklasifikovaný (částečné učení s učitelem)
- algoritmus se "ptá" učitele na zařazení příkladů (aktivní učení)
- nepřímé náznaky odvozené s chování učitele
- žádné (učení bez učitele)

Úloha empirického učení z dat

- Analyzovaná data:
  - řádky = objekty, instance, příklady
  - sloupce = atributy, veličiny

$$\mathbf{D}_{TR} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} & y_1 \\ x_{21} & x_{22} & \cdots & x_{2m} & y_2 \\ \vdots & \vdots & & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} & y_n \end{bmatrix}$$

- Klasifikační/predikční úloha: hledáme znalosti (reprezentované hypotézou h) h: $x \rightarrow y$, která pro hodnoty vstupních atributů x nějakého objektu odvodí hodnotu cílového atributu $\hat{y} = h(x)$.
- Cílem je nalézt takové znalosti, aby se minimalizovala chyba klasifikace
- Pak očekáváme, že tyto znalosti budou dobře korespondovat i s dalšími příklady, které nebyly použity v procesu učení.
  - předpoklad stacionarity konceptu

Učení na základě podobnosti

- Objekty, patřící do téže třídy mají podobné charakteristiky
- nebezpečí garbage-in, garbage-out

Induktivní učení

- Hledáme hypotézu h, která bude konsistentní s (neznámou) funkcí f na trénovacích datech

Bias vyjadřuje kvalitu hypotézy, variance vyjadřuje robustnost hypotézy

Overfitting

- situace, kdy se model v průběhu učení soustředí na trénovací data a nemá dostatečnou schopnost generalizovat
- chyba na trénovacích datech klesé
- → chyba na testovacích datech začíná růst
- složité algoritmy

Underfitting

- situace, ve které se model nedokáže naučit tak dobře souvislost mezi daty, a tak je začne predikovat špatně
- "jednoduché" algoritmy

Chyba klasifikace:

- Příklad, který patří do třídy y zařadíme do třídy ŷ
- Spočítáme počet chybně zařazených příkladů

	Klasifikace systémem	
Správné zařazení	+	-
+	TP	FN
-	FP	TN

$$Err = \frac{FP + FN}{TP + TN + FP + FN} \qquad Err(h, D_{TR}) = \frac{1}{n}\sum_{i=1}^{n} Q_h(\boldsymbol{o}_i, \hat{y}_i)$$

Ztráta:

- Různé chyby klasifikace jsou různě závažné (např. půjčit peníze někomu, kdo je nevrátí je horší chyba než nepůjčit peníze někomu, kdo by je vrátil i s úrokem)

EmpLoss = FP * L(-,h(x)=+) + FN * L(+, h(x)=-)

$$EmpLoss_{L,E}(h) = \frac{1}{N}\sum_{(x,y)\in E} L(y, h(x))$$

Regularizace:

- Při hodnocení modelů penalizujeme jejich složitost

$$Cost(h) = EmpLoss(h) + \lambda\, Complexity(h)$$

Učení jako prohledávání

- hledáme struktury i parametry modelu
- hledání optimální hypotézy v prostoru možných modelů
- př. rozhodovací stromy

Učení jako aproximace

- hledáme "jen" parametry modelu
- na základě konečného počtu bodů [xi ,yi] se snažíme určit parametry předpokládané hypotézy y = h(x)
- aproximace neznámé cílové funkce f(x), která mapuje vstup x na odpovídající výstupy y
- př. neuronové sítě

Neuronová síť

- představuje výpočetní systém inspirovaný biologickými neuronovými sítěmi
- n-tice (U, W, A, O, net, ex)
  - U je konečná množina výpočetních jednotek (neuronů),
  - W je struktura sítě, tedy zobrazení z U X U do R (vazby mezi neurony),
  - A je zobrazení definující aktivační funkci
    Au : R → R pro každou jednotku u,
  - O je zobrazení definující výstupní funkci
    Ou : R → R pro každou jednotku u,
  - net je zobrazení, definující vstupní funkci (přenosovou funkci)
    Iu : (R X R)U → R pro každou jednotku u,
  - ex je externí vstupní funkce ex: U → R , která každé jednotce u přiřadí externí vstup jakožto reálnou hodnotu exU = ex(u) ∈ R.
-

PAC teorie (Probably Aproximately Correct)

- slouží k analýze, zda se model strojového učení dokáže efektivně naučit generalizovat z trénovacích dat na neviděná data.
- Poskytuje formální záruky, že model s dostatečným počtem dat dokáže dosáhnout chyby menší než zvolená hodnota ϵ pravděpodobností alespoň 1−δ.
- PAC teorie také odhaduje minimální počet trénovacích vzorků potřebných k dosažení těchto záruk a zavádí pojmy, jako je VC dimenze, která měří kapacitu modelu.
- Díky tomu pomáhá vyhnout se přeučení a nalézt rovnováhu mezi komplexitou modelu a požadavky na data.

No free lunch

- Pro libovolné dva algoritmy A a B platí, že je-li algoritmus A lepší (ve smyslu chyby) než algoritmus B na nějakých k datových souborech (úlohách), pak existuje jiných k datových souborů (úloh), na kterých je algoritmus B lepší než algoritmus A.
- neexistuje univerzálně nejlepší algoritmus

Ošklivé kačátko

- Počet atributů, ve kterých se popisy libovolných dvou příkladů shodují, je konstantní.
- Klasifikace předpokládá nějaký bias

08 - Použití znalostí v učení

čtvrtek 21. listopadu 2024 14:37

Klasifikace znalosti

- Podle formalizovatelnosti
  - explicitní - formalizované, artikulované
  - implicitní - primárně skryté (v datech)
  - tacitní - nevědomé a nesdělitelné znalosti skryté v myslích expertů
- podle obsahu
  - deklarativní - zachycují co platí
  - procedurální - zachycují jak postupovat

Požadavky na znalosti

- transparentnost
- modulárnost
- modifikovatelnost
- užitečnost

Reprezentace znalosti v AI

- Predikátová logika
  - rozšíření výrokové logiky
  - predikáty, funkce, logické spojky, kvantifikátory
- Sémantické sítě
  - popisuje realitu jako objekty (uzly), které jsou v nějaké relaci (hrany)
- rámce
  - datová struktura
  - obsahuje:
    - data
    - meta-data
    - procedury
  - staly se inspirací pro OOP
- pravidla
  - IF-THEN
  - použito v expertních systémech (rule-based)
- Případy
  - mají podobu vyřešených problémů z dané aplikační oblasti
  - použití v systémech případového usuzování

09 - Zpětnovazební učení

čtvrtek 28. listopadu 2024 14:41

- Učení založené na interakci s prostředím:
  - je aktivní spíše než pasivní
  - interakce mají obvykle sekvenční charakter
  - je orientované na dosažení cíle
  - probíhá bez příkladů optimálního chování
  - Cílem je optimalizovat souhrnnou odměnu plynoucí z interakce s prostředím.

- S každým stavem s je spojena odměna za jeho dosažení
  - Agent se učí na základě odměn a trestů, které mu dává prostředí

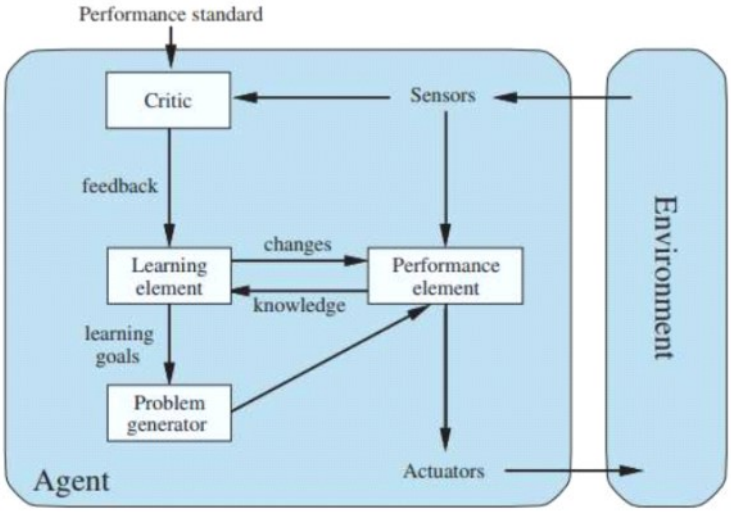
- U nějakých problémů je složité "oštítkovat" učitelem všechny možné inputy
  - Pak se hodí učení se zpětnou vazbou

Zpětnovazební učení

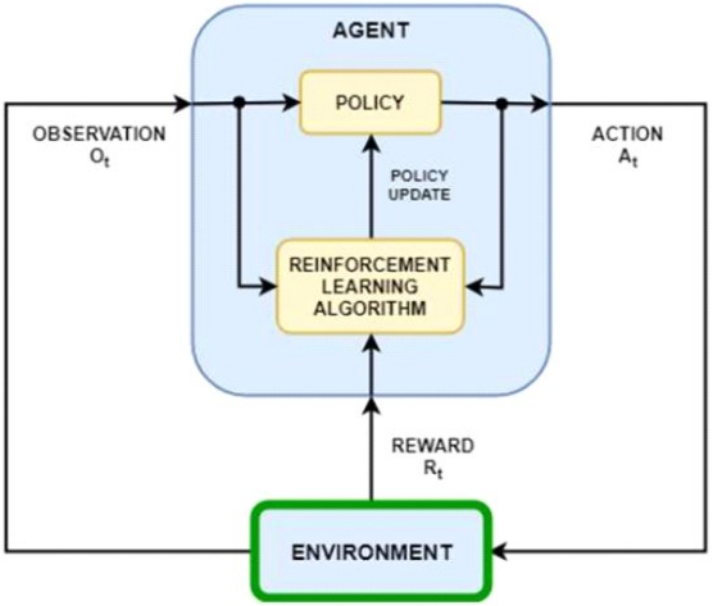
- Pasivní: cílem agenta je naučit se hodnoty užitku pro jednotlivé stavy U(s) za předpokladu fixní strategie π
- Aktivní: cílem agenta je naučit se i strategii

Obecný učící se agent

- učící se komponenta
  - slouží ke zdokonalování agenta
- výkonná komponenta
  - provádí příslušné akce agenta
- kritika postupu učení
  - zpětnovazební řízení procesu učení
- generátor
  - návrh akcí, které může agent provádět
-



Úloha posilovaného učení

- 

Markovský rozhodovací proces (MDP)

- matematický model používaný pro rozhodování v prostředí, kde výsledky akcí jsou částečně náhodné a částečně pod kontrolou agenta
- pravděpodobnost přechodu do stavu s' závisí jen na aktuálním stavu s a akci a, ne na historii
- Cílem agenta je najít optimální **politiku** π(s), což je pravidlo, které určuje, jakou akci a provést v každém stavu s, aby maximalizoval **očekávaný kumulativní zisk**.
- MDP je sekvenční rozhodovací proces (diskrétní ve stavech i v čase) s plně pozorovatelným stochastickým prostředím, markovskou přechodovou funkcí a aditivní užitkovou funkcí.

10 - Počítačové vidění

čtvrtek 5. prosince 2024 14:37

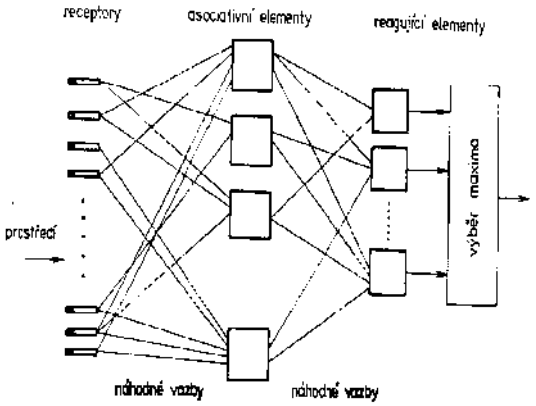
- věda i technologie usilující o vytváření "strojů schopných vidět a vnímat"

Typické úlohy:
- rozpoznávání
- analýza pohybu
- rekonstrukce scény
- restaurace obrazu
- snímání, digitalizace (OCR)
- předzpracování
- segmentace obrazu na objekty
- popis objektů
- porozumění obsahu obrazu

Základní charakteristiky ukládání obrazu:
- vzorkován (rozlišení)
- kvantování: počet možných hodnot pro reprezentaci informace v jednom bodu

Detekce hran
- Základní myšlenka: hledat místa největšího gradientu jasu na základě derivace.
- Problém se šumem, který může vést k falešně pozitivním detekcím
  - průběh intenzity jasu je třeba vyhladit (zprůměrovat)
  - → Konvoluce
    - Nová hodnota je spočítána jako lineární kombinace hodnot obrazové funkce v okolí
    - Příspěvek jednotlivých pixelů v okolí O je vážen v lineární kombinaci koeficienty h podle:
    - $$g(x,y) = \sum_{(m,n)} \sum_{\in \mathcal{O}} h(x-m,\, y-n)\, f(m,n)$$

Rosenblattův perceptron
- navržen Frankem Rosenblattem v roce 1958
- klasický model umělé neuronové sítě
- inspirována lidským okem
- jeho úkolem bylo pomocí optických snímačů uspořádaných do pole 20x20 rozpoznávat jednotlivé znaky
- algoritmus pro **binární klasifikaci**, který rozhoduje, zda vstupní data patří do jedné ze dvou tříd, a to na základě lineární kombinace vstupních hodnot a vážených koeficientů.
- Tvořený třemi úrovněmi:
  - receptory (výstupy 0, 1)
  - asociativní elementy (pevné váhy +1, -1)
  - reagující elementy (vážený součet $\sum_i(W_iX_i)$)



Konvoluční neuronová síť

- hluboká (vícevrstvá), dopředná (feed-forward) neuronová síť
- používá se pro analýzu obrázků neurony z jedné vrstvy jsou spojeny jen s malou oblastí neuronů v následující vrstvě
- výstupem je vektor pravděpodobností

11 - Agenti a roboti

čtvrtek 12. prosince 2024 14:29

Robotika - disciplína o vytváření inteligentních strojů

Typy robotů
- Průmyslové roboty (manipulátory)
  - Pevně umístěné na pracovních pozicích
- Mobilní roboty
  - Roboty pohybující se v prostředí
- Roboty humanoidní
  - vzhledem napodobující člověka

Autonomní robot je inteligentní stroj schopen vykonávat úkoly samostatně bez lidské pomoci
- nejdůležitější vlastnost je schopnost reagovat na změny prostředí
- základní části:
  - senzory
  - řídící jednotka
  - efektory (aktuátory)

Prostředí agenta
- pozorovatelné plně / částečně
- deterministické / stochastické
- epizodické / sekvenční
- statické / dynamické
- diskrétní / spojité
- jednoagentové / multiagentní

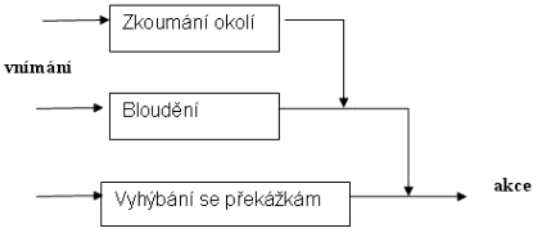
Sensorický subsystém
- sensory
  - pasivní (využívají fyzikální vlastnosti okolí)
    - kamera, gyroskop, tlačítka
  - aktivní (vysílají signál a detekují jeho návrat)
    - dálkoměrové senzory, GPS
- sensory
  - lokální (součástí robota)
  - distribuované

Příklady robotů:
- shakey - první univerzální mobilní robot
  - vedlejším produktem vývoje byl algoritmus A*
- TJ2 - robot schopný komunikovat v přirozeném jazyce
- Genghis - chodící robot
  - záporná zpětná vazba když se dotkl země tělem
  - kladná zpě5tná vazba když se pohyboval vpřed nebo vzad
- Hannibal - vylepšená verze zamýšlená jako planetární mobilní robot
- Cog - Humanoidní robot aproximující vnímání a dynamiku lidského těla
- Kismet - humanoidní robot simulující sociální interakce
- Coco - humanoidní robot, který se dokáže pohybovat
  - inspirací pro tvar těla bylo tělo gorily

Typy agentů

- Reaktivní agent
  - nejjednodušší typ agenta
  - jedná přímo na základě aktuálního stavu prostředí bez vytváření složitých plánů
  - př. hloupý robotický vysavač
- Deliberativní agent
  - na nedostatky reaktivních agentů navazuje agent deliberativní
  - plánuje své akce na základě modelu prostředí a svých cílů
  - dokáže plánovat cestu k množině svých cílů
  - plány tvoří na základě databáze svých znalostí a vnitřních stavů
  - V interakci s prostředím si tvoří jeho symbolickou reprezentaci, která je uložena ve formě jednotlivých tvrzení o světě
  - př. Robot, který plánuje trasu, aby doručil balík na specifické místo, a při tom zvažuje překážky a alternativní cesty.

Subsumpční architektura

- dekompozice agenta dle jeho aktivit do vrstev
- jednotlivé vrstvy spolu "soupeří" o řízení agenta
- Nejspodnější vrstva má přednost
- 

Celulární automat

- dynamický systém, diskrétní v prostoru a čase
- pravidelná struktura buněk v N-rozměrném prostoru (nejčastěji N=2)
- každá buňka se nachází v jednom z K stavů
- hodnoty stavů buňky se počítají na základě lokální přechodové funkce
  - funkce obsahuje aktuální stav a stav buněk v okolí
- využití:
  - simulace přírodních jevů
    - růst krystalů
    - šíření požáru
    - šíření nemoci
  - teoretická informatika
    - modelování výpočetních procesů
  - Fyzika
    - modelování plynových dynamik
    - simulace termodynamických procesů
  - Kryptografie
    - náhodnost generovaná CA může být použita v šifrovacích algoritmech nebo k tvorbě pseudonáhodných čísel