UMĚLÁ INTELIGENCE

URČENO PRO VZDĚLÁVÁNÍ V AKREDITOVANÝCH STUDIJNÍCH PROGRAMECH

EVA VOLNÁ

ČÍSLO OPERAČNÍHO PROGRAMU: CZ.1.07
NÁZEV OPERAČNÍHO PROGRAMU:
VZDĚLÁVÁNÍ PRO KONKURENCESCHOPNOST
OPATŘENÍ: 7.2
ČÍSLO OBLASTI PODPORY: 7.2.2

INOVACE VÝUKY INFORMATICKÝCH PŘEDMĚTŮ VE STUDIJNÍCH PROGRAMECH OSTRAVSKÉ UNIVERZITY

REGISTRAČNÍ ČÍSLO PROJEKTU: CZ.1.07/2.2.00/28.0245

OSTRAVA 2013

Tento projekt je spolufinancován Evropským sociálním fondem a státním rozpočtem České republiky

Recenzent: RNDr. Martin Kotyrba, Ph.D.

Název: Umělá inteligence
Autor: doc. RNDr. PaedDr. Eva Volná, PhD.
Vydání: první, 2013
Počet stran: 138

Jazyková korektura nebyla provedena, za jazykovou stránku odpovídá autor.

OBSAH

ÚVOD PRO PRÁCI S TEXTEM PRO DISTANČNÍ STUDIUM.....5
1. ÚVOD DO UMĚLÉ INTELIGENCE 6
1.1 CO JE TO UMĚLÁ INTELIGENCE 6
1.2 TŘI ZÁKLADNÍ PROUDY UMĚLÉ INTELIGENCE 8
1.3 TURINGŮV TEST 9
2 ŘEŠENÍ ÚLOH A PROHLEDÁVÁNÍ STAVOVÉHO PROSTORU13
2.1 STAVOVÝ PROSTOR A JEHO PROHLEDÁVÁNÍ 14
2.2 NEINFORMOVANÉ METODY PROHLEDÁVÁNÍ.....17
2.3 INFORMOVANÉ METODY PROHLEDÁVÁNÍ 19
2.4 PLÁNOVÁNÍ 22
2.5 PROGRAMOVÉ SYSTÉMY PRO ŘEŠENÍ ÚLOH 25
3 EXPERTNÍ SYSTÉMY 30
3.1 STRUKTURA EXPERTNÍHO SYSTÉMU 30
3.2 TYPY EXPERTNÍCH SYSTÉMŮ 33
3.3 ZNALOSTNÍ INŽENÝRSTVÍ 36
4 MULTIAGENTOVÉ SYSTÉMY 39
4.1 AGENT.....40
4.2 ARCHITEKTURA AGENTA.....41
4.3 MULTIAGENTOVÉ SYSTÉMY 45
5 UMĚLÝ ŽIVOT 49
5.1 CO JE TO UMĚLÝ ŽIVOT.....49
5.2 MODELY UMĚLÉHO ŽIVOTA.....50
6 DETERMINISTICKÝ CHAOS.....58
6.1 CO JE TO DETERMINISTICKÝ CHAOS.....59
6.2 NEJJEDNODUŠŠÍ MODEL DETERMINISTICKÉHO CHAOSU 62
6.3 ŘÍZENÍ DETERMINISTICKÉHO CHAOSU 64
7 FRAKTÁLY 69
7.1 CO JE TO FRAKTÁL.....69
7.2 GENEROVÁNÍ FRAKTÁLŮ.....74
7.3 FRAKTÁLNÍ DIMENZE 76
7.4 VÝSKYT FRAKTÁLŮ.....80
8 UMĚLÉ IMUNITNÍ SYSTÉMY 83
8.1 BIOLOGICKÝ IMUNITNÍ SYSTÉM.....83
8.2 UMĚLÉ IMUNITNÍ SYSTÉMY 87
8.3 APLIKACE UMĚLÝCH IMUNITNÍCH SYSTÉMŮ.....92
9 VÝPOČTY NA BÁZI DNA.....96
9.1 DEOXYRIBONUKLEOVÁ KYSELINA 96

9.2 VYUŽITÍ PRINCIPŮ DNA PŘI VÝPOČTECH 99
10 ROBOTIKA 108
10.1 CO JE TO ROBOT 109
10.2 ZPRACOVÁNÍ PŘIROZENÉHO JAZYKA 112
10.3 POČÍTAČOVÉ VIDĚNÍ 117
11 OCR 121
11.1 OPTICKÉ ROZPOZNÁVÁNÍ ZNAKŮ 121
11.2 OCR ALGORITMY 123
12 VYBRANÉ APLIKACE UMĚLÉ INTELIGENCE 130
LITERATURA 136

Úvod pro práci s textem pro distanční studium

Cíl předmětu

Tento text je inovací distanční studijní opory „Umělá inteligence“ (Habiballa 2004) a má sloužit pro potřeby výuky výběrového předmětu umělá inteligence na katedře informatiky a počítačů. Nepředpokládá se žádná předchozí znalost problematiky. Cílem tohoto textu je seznámit studenty se základními problémy umělé inteligence a vymezit umělou inteligenci jako obor i jako soubor unikátních problémů a přístupů k řešení. Absolvent by měl získat potřebný nadhled pro další studium v hlouběji zaměřených předmětech.

Po prostudování textu budete znát:

- základní problematiku z oblasti umělé inteligence,
- metody prohledávání stavového prostoru,
- úvod do problematiky expertních a multiagentových systémů,
- úvod do umělého života,
- úvod do robotiky a vybrané aplikace umělé inteligence.

V textu jsou dodržena následující pravidla:

- je specifikován cíl lekce
- výklad učiva
- důležité pojmy
- úkoly a otázky k textu
- korespondenční úkoly (mohou být sdruženy po více lekcích)

Úkoly

Vyberte si jeden korespondenční úkol a vypracujte na toto téma semestrální projekt, jehož obhajoba proběhne v dohodnutém termínu. Podrobné informace obdržíte na úvodním tutoriálu.

Pokud máte jakékoliv věcné nebo formální připomínky k textu, kontaktujte autora (eva.volna@osu.cz).

1. Úvod do umělé inteligence

V této kapitole se dozvíte:

- Co se rozumí pojmem umělá inteligence
- Jaký je princip Turingova testu.

Po jejím prostudování byste měli být schopni:

- Charakterizovat pojem umělá inteligence.
- Vysvětlit princip Turingova testu.

Klíčová slova této kapitoly:

Umělá inteligence, silná a slabá umělá inteligence, Turingův test, Loebnerova cena.

Průvodce studiem

Tento učební text by vám měl především otevřít obzor do celé škály problémů, postupů a otázek, které spadají do oblasti umělé inteligence. I když existuje mnoho definic umělé inteligence, je velice těžké vymezit ji jako disciplínu s přesnými hranicemi jako je teoretická informatika, numerická matematika a podobně.

Turingův test je typickou úlohou umělé inteligence. I přestože je definován poměrně neurčitě a těžko ho lze považovat za exaktní definici umělé inteligence, pokusy o jeho řešení jsou pro pochopení základních východisek umělé inteligence velice potřebné.

1.1 Co je to umělá inteligence

Zařazení umělé inteligence jako oboru je vcelku obtížné. Lze na ni pohlížet jako na matematickou disciplínu s aplikacemi, nebo také jako

na technický obor. Závratný růst tohoto oboru ovlivňovalo mnoho faktorů, jako např. zvyšování požadavků v oblastech automatizovaného řízení, průzkumu nedosažitelných míst a řada dalších činností, kde je přítomnost člověka z technických či zdravotních důvodů vyloučena.

Umělá inteligence (UI) jako vědní disciplína se postupně formuje jako průsečík několika disciplín, jakými jsou např. psychologie, neurologie, kybernetika, matematická logika, teorie rozhodování, informatika, teorie her, lingvistika atd. Její vývoj není zdaleka ukončen. Dnes jí však již nikdo neupírá právo na samostatnou existenci. Přesto má však mezi ostatními vědními disciplínami poněkud specifické postavení, a to ze dvou důvodů. Za prvé, dosud neexistuje všeobecně přijímaná definice umělé inteligence; za druhé, umělá inteligence neposkytuje jednotící teorie - spíše volně sdružuje různorodé teorie, metody a techniky, které lze úspěšně používat k počítačovému řešení některých složitých úloh rozhodování, plánování, diagnostiky apod. Počítačová UI byla ve větším měřítku použita poprvé v NASA, a to k řízení a kontrole letu dálkových družic. Asi všem je jasné, že základem UI je počítač. Bez počítače by nebylo UI a podle toho, jakou roli hraje počítač v existenci umělé inteligence, rozdělujeme UI na *slabou* - tam je počítač jen užitečným nástrojem a *silnou* - počítač není pouhým nástrojem, nýbrž přímo myslí a pomocí programu chápe. Možná někdy v budoucnu se objeví tak sofistikovaný počítač, který zvládne myslet sám o sobě.

V literatuře můžeme nalézt různé **definice umělé inteligence**. Pokusme se zde uvést alespoň nejdůležitější a nejvýstižnější z nich.

Marvin Minsky tvrdí, že „… *umělá inteligence je věda o vytváření strojů nebo systémů, které budou při řešení určitého úkolu užívat takového postupu, který – kdyby ho dělal člověk – bychom považovali za projev jeho inteligence*.“ (1967)

Tato definice vychází z Turingova testu a vyplývá z ní, že úlohy jsou tak složité, že i u člověka by vyžadovaly použití inteligence. Otázkou však je, jaké vlastnosti má složitost a „inteligentní“ řešení ? Složitost lze

ohodnotit počtem všech řešení, které připadají v úvahu. Ale hledání řešení pouhým prohledáváním stavového prostoru možných řešení není možné u složitějších úloh ani prostřednictvím superrychlých počítačů. Tento postup navíc není možno nazvat inteligentním. Je tedy nutno omezit velikost množiny prohledávaných řešení, což se děje na základě využívání znalostí.

E. Richová se domnívá, že „… *umělá inteligence se zabývá tím, jak počítačově řešit úlohy, které dnes zatím zvládají lidé lépe*." (1991)
Tato definice se bezprostředně váže na aktuální stav v oblasti počítačových věd a je možno očekávat, že se v budoucnosti bude ohnisko této vědy posouvat a měnit. Nevýhodou této – jinak velmi výstižné – definice je fakt, že nezahrnuje úlohy, které dosud neumí řešit počítače, ale ani člověk.

Alternativní definice

1 UI je označení uměle vytvořeného jevu, který dostatečně přesvědčivě připomíná přirozený fenomén lidské inteligence.
2 UI označuje tu oblast poznávání skutečnosti, která se zaobírá hledáním hranic a možností symbolické, znakové reprezentace poznatků a procesů jejich nabývání, udržování a využívání.
3 UI se zabývá problematikou postupů zpracování poznatků – osvojováním a způsobem použití poznatků při řešení problémů.

Umělá inteligence je interdisciplinární vědou, která nemá pevně vymezený předmět zkoumání ani teoretický základ – jde spíše o soubor metod, teoretických přístupů a algoritmů, sloužících k řešení velmi složitých úloh.

1.2 Tři základní proudy umělé inteligence

Symbolický funkcionalismus

Symbolický funkcionalismus je založen na dvou základních hypotézách: funkcionalistické hypotéze a hypotéze fyzikálního systému symbolů.

Zjednodušeně řečeno, základními předměty výzkumu symbolického funkcionalismu je problém reprezentace znalostí a inteligentní prohledávání stavového prostoru.
Se symbolickými způsoby náhledu na umělou inteligenci se při svém studiu setkáte poměrně zblízka. Jednak se budete hlouběji zabývat především typickou představitelkou symbolismu – logikou a také se blíže seznámíte ve vyšších kurzech s problematikou reprezentace znalostí různými metodami.

Konekcionismus

Konekcionismus, jako směr bádání v rámci umělé inteligence předpokládá, že inteligence plyne ze statického propojení velkého počtu jednoduchých výpočetních jednotek. Myšlenka je inspirována mozkem jako médiem, které zosobňuje inteligentní uvažování. Základní výpočetní jednotkou rozumíme model neuronu, který se na základě hodnoty součtu vážených vstupů excituje do aktivního stavu nebo nikoliv. Mezi hlavní odvětví, které se z řadí ke konekcionismu patří neuronové sítě, se kterými se rovněž můžete během svého studia blíže seznámit ve vyšších kurzech.

Robotický funkcionalismus

Klíčová filosofie robotického funkcionalismu je založena na implementaci chování jako psychologické školy. Raději než se zabývat reprezentací inteligence, robotičtí funkcionalisté se koncentrují na funkcionalitu modelovaného systému. Jako inteligentní chování je zde chápána jako rozumná interakce mezi třemi entitami: systém, prostředí, úloha. V případě že bude agent na danou úlohu a v daném prostředí reagovat inteligentně (jako by reagoval člověk), je považován za agenta disponující schopností vykazovat inteligentní chování.

1.3 Turingův test

Turing se konstrukcí svého testu snažil otázku "mohou stroje myslet" převést z oblasti filozofických spekulací na exaktnější úroveň. Za

myslící podle něj prohlásíme počítač tehdy, když jeho chovaní nebudeme schopni rozeznat od chování člověka.

Motivací pro vývoj umělé inteligence existovala už v polovině 20. století celá řada. Jedná se jistě o obrovskou teoretickou výzvu. Zkoumání inteligence umělé nám zřejmě dokáže mnoho říct o inteligenci vlastní. Dalším důvodem je prostá radost z konstrukce stále dokonalejších mechanismů. A neposlední řadě zde jsou motivy ryze praktické: Řadu úkolu musí plnit inteligentní bytosti, nicméně lidem se do nich příliš nechce (jsou např. stereotypní, nebo naopak nebezpečné).
Stochastické algoritmy pro globální optimalizaci heuristicky prohledávají prostor *D*. Heuristikou rozumíme postup, ve kterém se využívá náhoda, intuice, analogie a zkušenost. Rozdíl mezi heuristikou a deterministickým algoritmem je v tom, že na rozdíl od deterministického algoritmu heuristika nezajišťuje nalezení řešení. Heuristiky jsou v praktickém životě zcela samozřejmě užívané postupy, jako příklady můžeme uvést hledání hub, lov ryb na udici, pokus o složení zkoušky ve škole aj.

Ve své původní podobě vychází **Turingův test** z tzv. imitační hry, ve které jde o to odlišit dva lidi podle pohlaví. Pozorovatel, na jehož pohlaví nezáleží, má proti sobě např. ženu a muže, který předstírá, že je žena. Trojice lidí spolu nepřijde do fyzického kontaktu, sedí v oddělených místnostech a zprostředkovatel mezi nimi přenáší popsané lístky. Turingův test spočívá v tom, že imitátorem člověka by byl počítač. Vyvstává tedy otázka, zda dokáže počítač simulovat chování ženy stejně dobře jako muž. Test jako celek se však dosud žádnému programu splnit nepodařilo (Houser, 2003).

Rozsáhlá literatura existuje o tzv. paradoxu čínského pokoje, s nímž přišel filozof John Searl. Searlova námitka uvádí, že splnění Turingova testu ještě nemusí znamenat nějaké myšlení a uvědomování. V původní formulaci Searl ukazuje, že člověk může být např. schopný smysluplně třídit tabulky s čínskými znaky, aniž rozumí čínsky a chápe,

co text na tabulkách znamená. Podobně počítač, i když projde Turingovým testem, bude stále pouze něco imitovat.

Je možné, že v tuto chvíli ještě nevlastníme nějakou klíčovou znalost, která by nám umožnila konstruovat programy tak, aby prošly Turingovým testem. Američan Hugh Loebner v roce 1990 založil tzv. *Loebnerovu cenu*, která je každoročně udělována počítačovému programu za nejlepší konverzační schopnosti. Tato soutěž je variantou přísnějšího Turingova testu. Podle Turinga lze počítač považovat za "inteligentní", pokud jeho konverzační schopnosti nelze odlišit od schopností člověka. V Loebnerově soutěži skupina sudích hovoří s každým účastníkem soutěže a pokouší se určit, zda hovoří s člověkem nebo s počítačem. Zlatou a stříbrnou medaili obdrží ty programy, o nichž se nejméně polovina sudích domnívá, že hovořila s člověkem. Tyto medaile dosud nikdy nebyly uděleny. Bronzová medaile je udělována programu, o němž se největší počet sudích domníval, že hovoří s člověkem. Tuto medaili již dvakrát získal program *Alicebot*.

Kontrolní otázky:

1. Co se rozumí pod pojmem umělá inteligence?
2. Jaký je princip Turingova testu?

Úkoly k textu:

1. Najděte na Internetu více informací ohledně Loebnerovy ceny.
2. Najděte na Internetu další definice umělé inteligence.

Shrnutí obsahu kapitoly

Tato kapitola vymezuje pojem umělé inteligence. Existuje mnoho definic umělé inteligence, a proto je velice těžké ji vymezit jako disciplínu s přesnými hranicemi. Další část textu je věnován Turingovu testu, který je typickou úlohou umělé inteligence.

Pojmy k zapamatování

- umělá inteligence
- Turingův test
- Loebnerova cena

2 Řešení úloh a prohledávání stavového prostoru

V této kapitole se dozvíte:

- Co je to stavový prostor.
- Jaké jsou metody prohledávání stavového prostoru.
- Co je problém plánování.

Po jejím prostudování byste měli být schopni:

- Objasnit pojem stavový prostor.
- Charakterizovat neinformované (slepé) metody prohledávání stavového prostoru.
- Charakterizovat informované (heuristické) metody prohledávání stavového prostoru.
- Charakterizovat metody plánování.

Klíčová slova této kapitoly:

Stavový prostor, neinformované prohledávání, informované prohledávání, heuristika, algoritmus A-Star, rozklad úlohy na podproblémy, plánování, GPS, STRIPS, PLANNER.

Průvodce studiem

Stavový prostor si můžeme představit jako orientovaný graf (strom), jehož uzly představují jednotlivé stavy úloh a jehož hrany představují přechody mezi těmito stavy. Cesta z počátečního stavu do některého cílového stavu je řešením úlohy. Metody jak najít řešení ve stavovém prostoru jsou v principu nastíněny v této kapitole. Závěr kapitoly je věnován problematice plánování, a to hlavně programovým systémům pro řešení úloh (GSP, STRIPS a PLANNER).

2.1 Stavový prostor a jeho prohledávání

Stavový prostor je určen: ***SP*** = <***S***, ***O***>

- Konečnou neprázdnou množinou stavů ***S*** (ve zcela obecném případě i nekonečnou).
- Neprázdnou konečnou množinou operátorů ***O***, kde každý operátor je parciální funkcí na stavovém prostoru (nemusí být definován všude)
  $\alpha: S \rightarrow S$.

Úloha ve stavovém prostoru **SP** je <s_0, ***C***>, kde

- s_0 je počáteční stav,
- ***C*** je množina požadovaných cílových stavů.

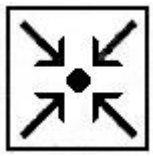
Plán (řešení) ***P*** pro danou úlohu ***U*** je taková posloupnost operátorů <α_1, α_2, ..., α_n>, která splňuje následující podmínky

$s_1 = \alpha_1 (s_0)$

$s_2 = \alpha_2 (s_1)$ ….

$s_n = \alpha_n (s_{n-1}) = \alpha_{n-1}(\alpha_{n-2}(\ldots (\alpha_2(\alpha_1(s_0)))))$,

přičemž s_n je definováno a je prvkem množiny cílových stavů ***C***.



Příklad (Habiballa 2004):

Jsou dány dvě nádoby, větší ***A*** o obsahu ***a*** litrů, a menší ***B*** obsahu ***b*** litrů:

- na začátku jsou obě prázdné (= *počáteční stav*)
- cílem je stav, kdy nádoba ***A*** je prázdná a v ***B*** je přesně $2\cdot(a - b)$ litrů vody

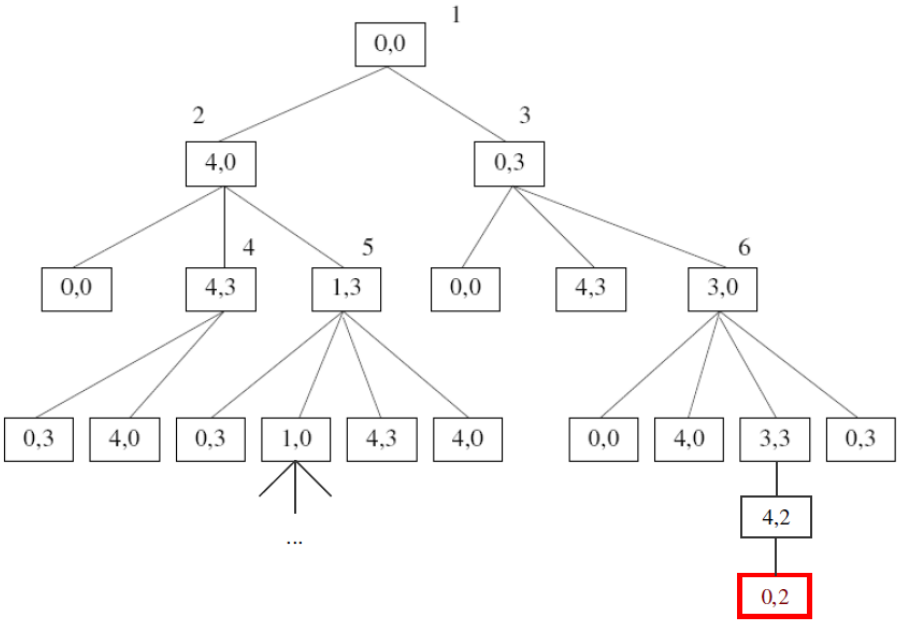
K dispozici je neomezený zdroj vody a nádoby nemají označené míry.

Naplňte nádobu B (= *nalezněte posloupnost akcí, že bude splněn daný cíl*).

Reprezentace stavů:

Zvolení vhodné reprezentace je jednou z důležitých podmínek efektivního vyřešení konkrétního problému. Stavy jsou v naprosté většině případů generovány v průběhu algoritmu

- Stav: dvojice <c_A, c_B> množství vody v nádobách ***A***, ***B***
  - počáteční stav <***0, 0***>
  - jediný cílový stav <***0***, ***2·(a − b)*** >
- přechody (= operátory):
  - vylití ***A***: <c_A, c_B > → <***0***, c_B >
  - vylití ***B***: <c_A, c_B > → <***0***, c_{BA}, ***0***>
  - napnění ***A***, ***B***
  - přelití ***A*** do ***B***
  - <c_A, c_B > → <max(c_A – (***b*** - c_B), 0), min(c_A + c_B, ***b***)>
  - možné jsou samozřejmě i další operátory



Obrázek 2.1: Stavový prostor úlohy přelévání vody (převzato z https://cw.felk.cvut.cz)

Prohledávání stavového prostoru

Stavový prostor lze reprezentovat orientovaným grafem G (stavovým grafem, stromem) ***G*** = (***V***; ***E***), ***V***: vrcholy (uzly), ***E***: hrany

- uzel reprezentuje stav,
- hrana reprezentuje přechod mezi stavy.

Řešení úloh pak lze formulovat jako hledání přijatelné cesty v orientovaném grafu z počátečního uzlu do některého z cílových uzlů (viz obr. 2.1).

Pozn.: k jednotlivým hranám je často přiřazena i cena vykonání daného přechodu.

Způsoby prohledávání stav. prostoru

1) Slepé (neinformované prohledávání):
   - do šířky – uzly k expanzi řadíme do fronty, náročné na paměť,
   - do hloubky – uzly řadíme do zásobníku; nutno ošetřit cykly.
2) Heuristické (informované prohledávání) - pořadí prohledávání na základě "další" informace, odhadu vzdálenosti stavu od cíle, vyjádřeného tzv. heuristikou ***h***(***uzel***).
   - paprskové (beam search),
   - gradientní (hill-climbing),
   - uspořádané (best-first),
   - A-Star (A*).

Dle čeho hodnotit jednotlivé algoritmy:

Pokud má být algoritmus výběru cesty úspěšný musí splňovat dvě základní vlastnosti:

- Musí vést k prohledávání, což znamená, že se musí procházet stavovým prostorem a přitom se vyhnout cyklům.
- Musí být systematický.

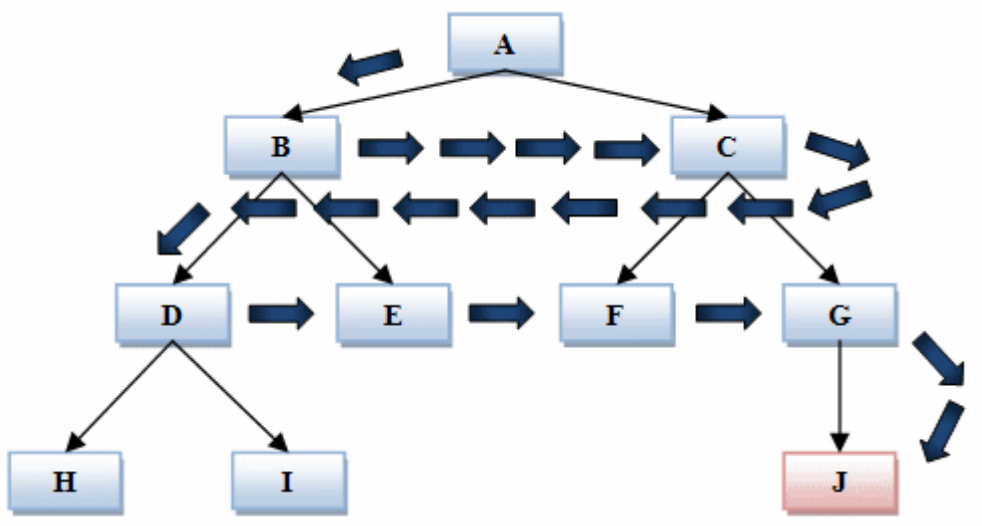
Při rozlišování mezi jednotlivými algoritmy se uplatňují zejména tyto kritéria:

- *Úplnost*: garantuje algoritmus nalezení řešení (pokud toto existuje).?
- *Časová složitost*: jak dlouho to potrvá – asymptotická reprezentace složitosti.
- *Paměťová náročnost*: kolik paměti je zapotřebí.

- *Optimalita*: nalezne algoritmus to nejlepší řešení v případě existence více řešení.

2.2 Neinformované metody prohledávání

Princip prohledávání do šířky spočívá v tom, že na každé úrovni vždy vygenerujeme všechny možnosti a ty teprve řešíme. Cena (ohodnocení) uzlu se rovná jeho hloubce. Uzly se expandují v pořadí, v jakém byly vygenerovány. Metoda nalezne vždy nejkratší cestu k cíli, pokud tato cesta existuje.



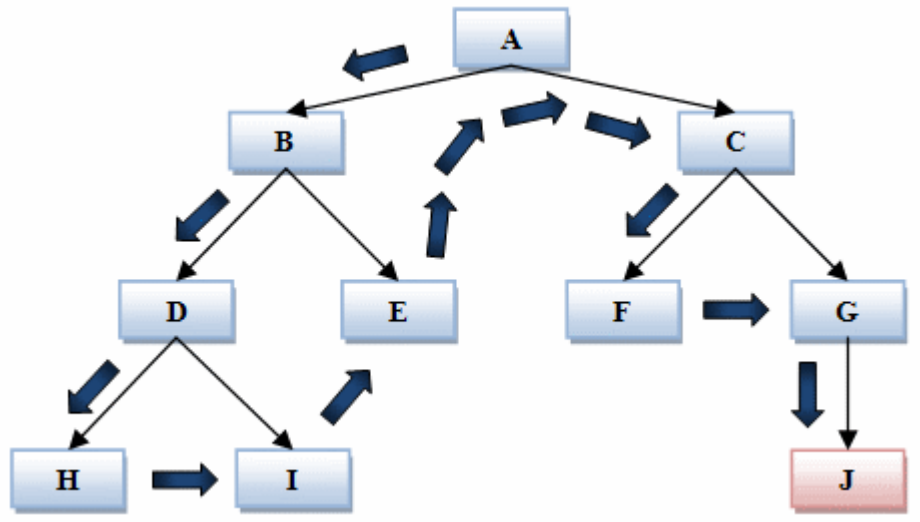
Obrázek 2.2: Prohledávání do šířky (převzato z http://faruzel.borec.cz/120.html)

Prohledávání do šířky (obr. 2.2)

1) CLOSED = { }, OPEN = {s_0}, kde s_0 je počáteční stav. Je-li s_0 současně i cílový stav, pak ukonči prohledávání.
2) Je-li OPEN prázdný, pak řešení neexistuje → ukonči prohledávání.
3) Vyber a současně vymaž první stav ze seznamu OPEN, označ jej s_i a zapiš jej do seznamu CLOSED.
4) Expanduj stav s_i. Pokud stav i nemá následovníky nebo všichni následovníci byli již expandováni (tj. jsou v seznamu CLOSED), jdi na krok 2.

5) Zapiš všechny následovníky stavu ***i***, kteří nejsou v seznamu CLOSED na konec seznamu OPEN.

6) Pokud některý z následovníků stavu je cílovým stavem, tj. řešení bylo právě nalezeno, ukonči prohledávání, jinak pokračuj krokem č. 2.



Obrázek 2.3: Prohledávání do hloubky (převzato z http://faruzel.borec.cz/120.html)

Prohledávání do hloubky (s omezením max na délku větve), obr. 2.3

1) CLOSED = { }, OPEN = { s_0}, kde s_0 je počáteční stav. Je-li s_0 současně i cílový stav, pak ukonči prohledávání.

2) Je-li OPEN prázdný, řešení neexistuje → ukonči prohledávání.

3) Vyber a současně vymaž první stav ze seznamu OPEN, označ jej s_i a zapiš jej do seznamu CLOSED

4) Pokud se hloubka uzlu i rovná maximální přípustné hloubce max, pokračuj krokem 2.

5) Expanduj stav s_i. Pokud stav s_i nemá následovníky nebo všichni následovníci byli již expandováni (tj. jsou v seznamu CLOSED), jdi na krok 2.

6) Zapiš všechny následovníky stavu si, kteří nejsou v seznamu CLOSED na začátek seznamu OPEN.

7) Pokud některý z následovníků stavu je cílovým stavem, tj. řešení bylo právě nalezeno, ukonči prohledávání, jinak pokračuj krokem č. 2

2.3 Informované metody prohledávání

Informované hledání je založeno na strategii, která využívá specifické znalosti pro daný problém, a nalezení cíle je mnohem efektivnější, včetně lepší možnosti dosáhnout optimálního řešení. Tyto znalosti se nazývají heuristické znalosti nebo také **heuristiky**.

Heuristické znalosti můžeme využít dvěma způsoby (Habiballa 2004):

- První způsob je zřejmější a spočívá v zahrnutí těchto znalostí do pravidel.
- Druhý způsob je založen na doplnění ohodnocující funkce o předpokládanou cenu k cíli:
  $\boldsymbol{f}^*(i) = \boldsymbol{g}^*(i) + \boldsymbol{h}^*(i)$,
  $\boldsymbol{f}^*(i)$ je ohodnocení uzlu i
  $\boldsymbol{g}^*(i)$ je dosavadní cena cesty z počátečního uzlu do uzlu i
  $\boldsymbol{h}^*(i)$ je předpokládaná cena cesty z uzlu i do cílového uzlu

Algoritmus A-Star (Russel, Norvig 2003)

Algoritmus A-Star je grafový algoritmus sloužící k nalezení „nejkratší“ cesty v ohodnoceném grafu vedoucí ze zadaného počátečního uzlu do uzlu cílového. Jeho vstupem je ohodnocený graf, počáteční uzel a cílový uzel, výstupem je nejkratší cesta z počátečního uzlu do cílového, nebo zpráva o tom, že žádná taková cesta neexistuje.

Algoritmus je v principu shodný s prohledáváním do šířky s tím rozdílem, že namísto obyčejné fronty používá frontu prioritní, ve které jsou cesty seřazeny podle hodnoty speciální funkce *f*. Tato funkce je definována pro každou cestu *p* a je součtem tzv. heuristické funkce (*h*) posledního uzlu cesty *p* a její délky (*g*). Čím je hodnota funkce *f(p)*

nižší, tím vyšší má daná cesta *p* prioritu. Dále se předpokládá, že cesty ve frontě neobsahují kružnice a pro každý cílový uzel se v ní nachází nanejvýš jedna cesta, a to ta nejkratší doposud nalezená.

Kroky algoritmu A-Star:

- Vytvoř prázdnou množinu cest *F*.
- Do množiny *F* vlož cestu nulové délky obsahující počáteční uzel *s*.
- Dokud není množina *F* prázdná, opakuj:
  - Z množiny *F* vyber nejkratší cestu *p* (s nejnižší hodnotou *f(p)*) a odeber ji.
  - Končí-li cesta v cílovém uzlu, vrať ji a ukonči výpočet.
  - Vytvoř nové cesty použitím všech možných operátorů na koncový uzel cesty *p*, které neobsahují smyčky.
  - Jestliže dvě cesty končí ve stejném uzlu, odstraň všechny kromě té nejkratší (s nejnižší hodnotou *f(x)*).
  - Přidej cestu *p* do množiny *F*.
- Je-li množina *F* prázdná, oznam, že žádná cesta z počátečního do cílového uzlu neexistuje.

Asymptotická složitost algoritmu A-Star silně závisí na použité heuristické funkci, která ovlivňuje množství cest uvažovaných během výpočtu. Asymptotická složitost vyjadřuje, jak roste náročnost algoritmu (doba výpočtu nebo potřebná paměť) s rostoucím množstvím vstupních dat. Obecně se však dá říci, že asymptotická složitost algoritmu A-Star nikdy nebude horší než složitost prohledávání do šířky.

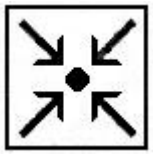
Rozklad úlohy na podproblémy

Rozklad úlohy na podproblémy lze opět znázornit grafem, ale na rozdíl od předcházející metody neodpovídají nyní uzly grafu stavům úlohy, ale podúlohám (podproblémům). Každý podproblém se opět rozkládá na jednodušší podproblémy až konečně listy grafu odpovídají buď elementárním úlohám nebo úlohám neřešitelným.

Druhý rozdíl od grafů stavových prostorů spočívá v tom, že rozeznáváme dva typy bezprostředních následníků – uzly AND a uzly OR (větve k následníkům typu AND se spojují obloučkem). Uzel je řešitelný pokud jsou řešitelní všichni jeho bezprostřední následníci typu AND, resp. je-li řešitelný alespoň jeden bezprostřední následník typu OR. V opačném případě je uzel neřešitelný.

Metody řešení rozkladem na podproblémy jsou vlastně metody prohledávání AND/OR grafů a můžeme je opět rozdělit na slepé (do šířky a do hloubky) a heuristické.

Příklad:



Rozklad problému na podproblémy ilustrujme na příkladu hry NIM.

Pravidla:

- hráč zadá počet zápalek (např. od 15 do 35)
- pak se střídá se strojem v odebírání; odebrat lze 1, 2 nebo 3 zápalky,
- prohraje ten, kdo odebere poslední zápalku.

Dílčí podproblémy:

- zadání počtu zápalek
- odebrání zápalek hráčem
- odebrání zápalek strojem

Hrubé řešení:

int *pocet*;
boolean *stroj* = false;
“*zadani_poctu_zapalek*“
do {
    if (stroj)
    “*odebrani_zapalek_strojem*“
    else
    “*odebrani_zapalek_hracem*“
    stroj = !*stroj*;
} while (*pocet*>0);
if (*stroj*) “vyhral_stroj“

else “vyhral_hrac“

Podproblémy „*zadani_poctu_zapalek*“, „*odebrani_zapalek_strojem*“ a „*odebrani_zapalek_hracem*“ budeme realizovat metodami. Proměnné *počet* a *stroj* pro ně budou statickými (čili nelokálními) proměnnými.

2.4 Plánování

Je-li dán počáteční model prostředí a požadovaný koncový model prostředí, pak je úkolem inteligentních strojů a systémů vyhledat vhodnou posloupnost akcí, jejichž aplikací lze přejít od počátečního modelu k cílovému. Taková posloupnost se nazývá *plán* a metody vytváření plánů se označují jako *metody řešení úloh*. Každému modelu odpovídá jistý stav prostředí, množina všech stavů tvoří *stavový prostor*.

Co vlastně pojem **plánování** představuje? Dalo by se říci, že každý z nás vědomě či nevědomě se s ním dnes a denně setkává. Představme si kupříkladu studenta, který se přihlašuje na zkoušku ve zkouškovém období. Musí brát na zřetel eventuální možné termíny, jejich obsazení, své splněné či nesplněné předpoklady k ukončení daného předmětu, stejně tak jako své vědomostní schopnosti a celkovou připravenost, aby věděl, kdy bez problémů dokáže danou zkoušku vykonat.

Takže na začátku je zapotřebí si důkladně promyslet (**naplánovat**) určitou posloupnost akcí (např. u našeho studenta splnění jeho úkolů, nastudování teorie apod.), abychom posléze mohli zdárně dospět k požadovanému výsledku.

Pro člověka je otázka **plánování** nepříliš složitá, samozřejmá, téměř každodenní činnost. Ale z hlediska **umělé inteligence** představuje jeden z problémů, které bychom mohli zařadit mezi tzv. *obtížné úlohy*.

Při řešení úloh, které představují vlastně klasické metody umělé inteligence, se používají různé způsoby prohledávání stavového prostoru. V něm každý bod odpovídá jedné možné situaci, která se

může ve skutečném reálném světě vyskytnout. Systémy umělé inteligence hledají vhodnou posloupnost akcí (nazvanou **plánem**), díky které lze přejít od počátečního stavu do stavu koncového.

Máme-li úlohu poněkud složitější, je důležité mít možnost si daný problém rozdělit na dílčí části. Pracujeme pak s menšími úlohami, jejichž výsledky nakonec zkombinujeme do jednoho celkového řešení. V případě, že nemáme možnost takového dílčího rozdělení, se naše úloha stane v podstatě neřešitelná. Získáme příliš velké množství dat, příp. podmínek, které je nutno v každém kroku uvažovat. Je tedy nutné úlohu určitým způsobem zjednodušit. K tomu nám dopomůžou dva způsoby.

První možností, jak toho dosáhnout, je přestat přepočítávat celý stav úlohy na stav nový a to při každém kroku ve stavovém prostoru. Namísto toho bychom měli jenom zaznamenávat změny stavu. Představme si příklad z praxe. Vstoupíme-li do kuchyně, abychom připravili oběd, pak touto naší akcí se nezmění rozmístění spotřebičů v kuchyni ani poloha kuchyně vůči bytu. Ale vstupem do kuchyně (pojmenování naší akce) může být změněna naše poloha např. vůči kuchyňské lince. Změn samozřejmě bude více, začneme-li vařit. Ty složky stavu řešené úlohy, které jsou příslušnou akcí změněny, přesně popisuje **rámec akce**. A právě s rostoucí složitostí stavu úlohy se zvyšuje důležitost mechanizmu rámců.

Vezměme si na pomoc další příklad. Takový obyčejný úklid domácnosti (např. vysávání koberců) rovněž představuje značně složitou situaci. Stav této úlohy musí zaznamenávat nejen polohu a vzájemné vztahy všech předmětů v blízkém okolí, ale i lokalizaci dotyčné osoby. Ovšem činnost tohoto člověka ovlivňuje pouze velmi malou část tohoto celkového stavu. Dá se předpokládat, že se nezmění poloha dveří, oken, skříní a větších kusů nábytku vůči celému bytu, ale třeba umístění vysavače (příp. židlí či jiných menších kusů nábytku) dozná jistě určitých změn. Při použití **rámců** se budeme snažit zachytit změny pouze těch složek stavu úlohy, které budou ovlivněny příslušnou akcí, přičemž ostatní složky stavu je možné považovat za neměnné.

Plánovací systém

U systémů pro řešení úloh, které jsou založeny na elementárních postupech, musíme nutně provádět každou z těchto funkcí :

- *Vybrat nejvhodnější pravidlo příštího kroku na základě dostupné heuristické informace.*

  Nejpoužívanější postup k výběru vhodného pravidla je postup, kdy se nejprve určí množina rozdílů mezi požadovaným stavem cílovým a stavem aktuálním. Potom se identifikují pravidla významná pro redukci těchto rozdílů. Bude-li nalezeno takových pravidel více, lze použít pro výběr mezi nimi dostupné heuristické informace. Ty sice nebývají podloženy hlubší teorií, ale velmi často pomáhají účinně nalézt řešení, ačkoli obecně jej negarantují.
- *Aplikovat vybrané pravidlo a vypočíst nový stav úlohy vycházející z použití tohoto pravidla*.

  U jednoduchých systémů popisujících omezený rozsah světa by byla aplikace pravidel snadná. Stačí specifikovat stav úlohy, který vyplynul z jeho použití. Chceme-li pracovat s pravidly, která specifikují jen malou část úplného stavu úlohy, je třeba použít jiný způsob, jak toho dosáhnout.
- *Zjistit, zda bylo nalezeno řešení.*

  Za úspěšné je považováno to řešení plánovací úlohy, kdy je nalezena posloupnost operátorů, která transformuje počáteční stav úlohy na stav cílový. Aby systém mohl tuto situaci rozpoznat, je zapotřebí přímá unifikace stavových popisů. Jsou-li však úplné stavové popisy popsány množinou požadovaných vlastností, je tento problém podstatně složitější.
- *Najít slepé cesty, které je možno vynechat, aby se systém mohl ubírat přijatelnějším směrem.*

  V tomto bodě se jedná o tentýž mechanismus usuzování jako pro detekci správného řešení. Plánovací systém musí být schopen zjistit, zda zkoumaná cesta není k nalezení řešení nemožná, resp. nepravděpodobná.

- *Detekovat, zda nebylo nalezeno téměř správné řešení, a pokud ano, tak použít zvláštních technik za účelem získání zcela správného řešení.*

Plánovací činnosti a rozvrhování jsou výsadou inteligentních živých bytostí. Samozřejmě kdo jiný než člověk je ze všech takových tvorů na tom nejvyšším stupni inteligence. I když jsou i takoví zástupci svého druhu, o kterých lze jistě hovořit jako o tvorech inteligentních (např. pes, delfín). Ale rozhodně nikdo jiný než člověk nedokáže natolik využít svých znalostí, aby uměl „vdechnout“ jistou dávku inteligence i neživému objektu. Opět se dostáváme k pojmu **umělá inteligence**, tedy něco, co se svým umem (myšleno v oblasti plánování) přibližuje člověku. Úloha plánování je tudíž považována jako netriviální problém **umělé inteligence**.

2.5 Programové systémy pro řešení úloh

GPS (General Problem Solver)

Cíle:

- ukázat, jak lze počítače využít pro řešení úloh vyžadujících inteligenci,
- přispět k poznání, jak člověk takovéto úlohy řeší.

Teoretickým přínosem systému GPS je metoda nazvaná *analýza prostředků a cílů*. Princip metody je založen na postupném rozkladu úloh na podúlohy metodou výpočtu diferencí. GPS vychází z diferencí mezi počátečním a cílovým stavem řešení úlohy a snaží se tyto diference postupně zmenšovat. Jednotlivé diference uspořádává podle důležitosti pro řešení, různým druhům diferencí je přiřazen různý význam a snahou systému je zmenšovat vždy nejvýznamnější diferenci. GPS se při své činnosti může dostat do stavu, *ve kterém se již nacházel*. Aby se nedostal do nekonečného cyklu, musí si vést seznam již vygenerovaných stavů, který je implementován jako *zásobník* - GPS může uskutečnit návrat k

předchozímu stavu. Vlastní řešení začíná procedurou, která se snaží vybrat *produkční pravidlo*, které by danou diferenci odstranilo nebo ji co nejvíce zmenšilo. Nejde-li to, určí proč a tuto informaci předá zpět proceduře hledající produkční pravidlo. Tento postup rekurzivně opakuje tak dlouho, až je buď *možné celou sekvenci hledaných pravidel aplikovat a tím odstranit diference*, nebo skončí *neúspěchem*, není-li úloha řešitelná.

STRIPS

STRIPS je programový systém, který byl vyvíjen jako následovník systému GPS. Vývoj systému STRIPS byl motivován aplikací v oblasti *navrhování a konstrukce inteligentních robotů*, avšak jeho základní algoritmus má *obecné využití*. Stavy řešené úlohy jsou popsány formulemi predikátového počtu prvního řádu a je specifikován počáteční a cílový stav úlohy. Pravidla pro přechody mezi stavy jsou popsána trojicemi (**C,D,A**), kde **C** je podmínka aplikovatelnosti pravidla, **D** je množina klausulí, které budou z popisu stavu vynechány, použije-li se toto pravidlo a **A** je množina klausulí, které budou v případě aplikace pravidla přidány. Zde však nehovoříme o odstraňování diferencí, ale o splňování cílů. Principiálně to znamená, že *objeví-li se nová diference, vznikl nový cíl*, který je třeba splnit. Řešení úlohy systémem STRIPS začíná tím, že se systém snaží splnit zadaný cíl řešení. Jestliže *cíl koresponduje* se specifikovaným cílovým stavem, je řešení úlohy úspěšně ukončeno. Jestliže *cíl nekoresponduje*, snaží se systém vybrat takové produkční pravidlo, v jehož seznamu se vyskytl fakt korespondující se zadaným cílem. Najde-li takové pravidlo, musí stejným způsobem pokračovat v plnění cílů uvedených v samostatném seznamu cílů pravidla. **Tento rekurzivní proces skončí tehdy, když jsou všechny cíle splněny**.

Tento postup nazýváme **plánem.** "Ladění" plánu provádí systém automaticky - postupně aplikuje produkční pravidla a jakmile zjistí nesrovnalost v jejich aplikovatelnosti na aktuální stav databáze, stanoví si jako nejbližší cíl jeho odstranění. Splňování nově stanoveného cíle

probíhá naprosto stejným postupem. STRIPS opět nejprve vytvoří náčrt plánu a potom postupně prověřuje jeho realizovatelnost (splnitelnost). Takovéto "etapy" se postupně vnořují do sebe do libovolné hloubky tak dlouho, až lze původní plán plně realizovat nebo se zjistí (dokáže), že zadaná úloha nemá řešení.

PLANNER

Programový systém *PLANNER* představoval ve své době významný pokrok především proto, že jako první využíval procedurální reprezentace znalostí. PLANNER reprezentuje konkrétní i obecné poznatky pro řešení úlohy obdobně jako STRIPS. Rozdíl je však v tom, že nepřipouští, aby fakta spojená s konkrétní situací obsahovala proměnné. Striktně rozlišuje specifické poznatky o konkrétní řešené úloze a obecné poznatky vztahující se k dané úloze. Obecné poznatky platí všeobecně pro celou třídu úloh a proto mají ve své reprezentaci proměnné, za které se podle potřeby dosazují konkrétní objekty. Specifické poznatky o konkrétních předmětech (objektech) se nazývají fakta a jsou uloženy v databázi (nesmí obsahovat proměnné). Fakta jsou v systému PLANNER reprezentována *deklarativně*, proto pomocí faktů nelze tudíž vyjádřit obecné poznatky. Ty se ukládají do báze znalostí v podobě pravidel. Pravidla obsahují informaci, za jakých podmínek mohou být použita a jak jejich provedení ovlivní obsah databáze. Pravidla jsou tedy nositeli procedurální reprezentace znalostí.

U pravidel systému PLANNER rozlišujeme *hlavu* pravidla a jeho *tělo*. Tělo obsahuje posloupnost příkazů realizujících operaci, hlava pravidla je jeho popisem umožňujícím jeho vyvolání. Pravidlo může být aplikováno jen na ty výrazy, které korespondují s jeho hlavou. Jestliže bychom systém PLANNER chtěli stručně charakterizovat, pak se jedná o pravidlový dedukční systém, který pracuje jak v přímém, tak i ve zpětném režimu, přičemž při dedukci využívá převážně režimu zpětného.

Kontrolní otázky:

1. Co je to stavový prostor?
2. Jaké jsou metody prohledávání stavového prostoru?
3. Jaký je princip A-Star algoritmu?
4. Charakterizujte rozklad úlohy na podproblémy.
5. Jaké znáte programové systémy pro řešení úloh.

Úkoly k textu

1. Nalezněte na Internetu další algoritmy prohledávání stavového prostoru a porovnejte je.
2. Nalezněte na Internetu další programové systémy pro řešení úloh a porovnejte je.

Korespondenční úkoly

1. Navrhněte, vytvořte a odlaďte program pro řešení jednoduchého problému pomocí zvoleného algoritmu prohledávání stavového prostoru.

   Příklady:

   - Mějme hrací plochu o 9 polích, obsahující kameny s čísly 1 až 8. Kameny jsou na hrací ploše náhodně rozmístěny a úkolem je posunem kamenů dosáhnout zvoleného cílového stavu.
   - Cesta bludištěm.
   - Problém obchodního cestujícího.
   - Hanojské věže.

   apod.

2. Najděte na Internetu vybraný systém pro plánování, stručně jej popište a vytvořte v něm řešení zvoleného plánovacího problému.

Shrnutí obsahu kapitoly

V této kapitole jsme se věnovali problematice řešení úloh a prohledávání stavového prostoru. Stavový prostor si můžeme představit jako orientovaný graf (strom), jehož uzly představují jednotlivé stavy úloh a jehož hrany představují přechody mezi těmito stavy. Cesta z počátečního stavu do některého cílového stavu je řešením úlohy. Dále byly popsány neinformované (slepé) a informované (heuristické) metody prohledávání stavového prostoru. Závěr kapitoly je věnován problematice plánování, a to hlavně programovým systémům pro řešení úloh (GSP, STRIPS a PLANNER).

Pojmy k zapamatování

- stavový prostor
- neinformované prohledávání
- informované prohledávání
- algoritmus A-Star
- programové systémy pro řešení úloh (GPS, STRIPS, PLANNER)

3 Expertní systémy

V této kapitole se dozvíte:

- Co je to expertní systém.
- Jaké jsou charakteristické rysy expertních systémů.
- Jaké jsou typy expertních systémů.
- Jaké typy úloh můžeme expertními systémy řešit.

Po jejím prostudování byste měli být schopni:

- Vysvětlit pojem expertní systém.
- Znát základní rozdělení a charakteristiku expertních systémů.
- Vědět, jaké typy úloh lze expertními systémy řešit.

Klíčová slova této kapitoly:

Expertní systém, inferenční mechanismus, báze znalostí, znalostní inženýrství.

Průvodce studiem

V této kapitole se podíváme na problematiku expertních systémů. Dozvíte se co je to expertní systém, kde se s ním můžete v praxi setkat, jaké typy úloh zpracovává, jaká je jeho vnitřní struktura a jaké typy expertních systémů existují. Jejich nasazení je ve velkém spektru problémů z průmyslu, zdravotnictví a dalších oblastí. V závěru kapitoly se zmíníme o znalostním inženýrství.

3.1 Struktura expertního systému

Expertní systémy jsou počítačové programy, simulující rozhodovací činnost experta při řešení složitých úloh a využívající vhodně

zakódovaných, explicitně vyjádřených znalostí, převzatých od experta, s cílem dosáhnout ve zvolené problémové oblasti kvality rozhodování na úrovni experta (Habiballa - 2004).

Charakteristické rysy expertních systémů

Oddělení znalostí a mechanismu pro jejich využívání

Znalosti experta jsou uloženy v bázi znalostí odděleně od inferenčního mechanismu. To umožňuje vytvářet problémově nezávislé (*prázdné*) expertní systémy (expert system shells), kde jeden inferenční mechanismus může pracovat s různými bázemi znalostí. Předem je dána pouze strategie využívání znalostí z této báze - řídící mechanismus neboli inferenční mechanismus.

1. *Neurčitost (nejistota) v bázi znalostí a neurčitost (nejistota) v datech*

   Způsob zpracování znalostí a dat v expertním systému musí mít některé rysy podobné způsoby uvažování experta. Expert pracuje velmi často s nejistými znalostmi - heuristikami - a s nejistými daty (např. se subjektivním popisem potíží pacienta, s výsledky přibližných metod vyhodnocování apod.). I expertní systém musí být schopen využívat nejistých znalostí, tj. znalostí s přidělenou mírou důvěry v jejich platnost - nejistota v bázi znalostí. Zde se pak objevují pojmy jako „často“ „většinou“, které je potřeba kvantifikovat (např. v nějaké škále od „určitě ano“ přes „nevím“ až k „určitě ne“). Takovou znalostí s neurčitostí může například být „jestliže má pacient teplotu, obvykle je mu předepsán acylpyrin"; pacient totiž „často" má teplotu v důsledku chřipky, ale „někdy“ může mít teplotu, protože má zlomenou nohu. Systém také musí umět využívat odpovědí, zahrnujících nejistotu uživatele, způsobenou nepřesně určenými hodnotami nebo subjektivním pohledem uživatele (odpovědi typu "nevím", "asi ano" apod.) - nejistota v bázi dat.

2. *Dialogový režim a báze dat*

   Expertní systémy jsou nejčastěji konstruovány jako tzv. konzultační systémy. Uživatel komunikuje se systémem způsobem

„dotaz systému - odpověď uživatele“ obdobně, jako s lidským expertem. Tento systém práce byl do značné míry usnadněn nástupem osobních počítačů a s tím souvisejícím přechodem od dávkového zpracování na sálech výpočetních středisek k interaktivnímu zpracování přímo na psacím stole uživatele. Báze znalostí popisuje znalosti z dané oblasti. Má tedy charakter obecného rozhodovacího pravidla (či systému rozhodovacích pravidel). Řešit konkrétní případ znamená "dosadit" data o daném případu do obecně formulovaných znalostí z báze znalostí. Data k danému konkrétnímu případu poskytuje obvykle uživatel sekvenčně, v dialogovém režimu s počítačem. Dialog uživatele s počítačem má charakter dialogu laika či méně zkušeného odborníka s expertem. Expertní systém se uživatele dotazuje na údaje, týkající se konzultovaného případu, a na základě odpovědí a svých obecných znalostí si postupně upřesňuje "představu" o případu a dochází k závěru, eventuálnímu řešení.

3. *Vysvětlovací činnost*

   Aby se zvýšila důvěra uživatelů v závěry a doporučení expertního systému, měl by systém poskytovat vysvětlení svého uvažování. Obvykle systém vysvětluje právě položený dotaz, znalosti relevantní k nějakému tvrzení, právě zkoumanou cílovou hypotézu, právě probíhající odvozování.

4. *Modularita a transparentnost báze znalostí*

   Pro účinnost expertního systému je rozhodující kvalita báze znalostí. Znalosti experta nemají statický charakter, nýbrž se postupně (a obvykle nikoliv pomalým tempem) vyvíjejí a rozrůstají. Modularita umožňuje snadnou aktualizaci báze znalostí, transparentnost umožňuje její snadnou čitelnost a srozumitelnost. Samotné vytváření báze znalostí probíhá iterativním způsobem při opakovaných konzultacích experta z dané problémové oblasti s odborníkem na tvorbu bází, tzv. znalostním inženýrem, kdy je báze znalostí postupně „laděna“', až chování expertního systému (alespoň při konzultaci pro vzorové příklady) odpovídá představám experta.

3.2 Typy expertních systémů

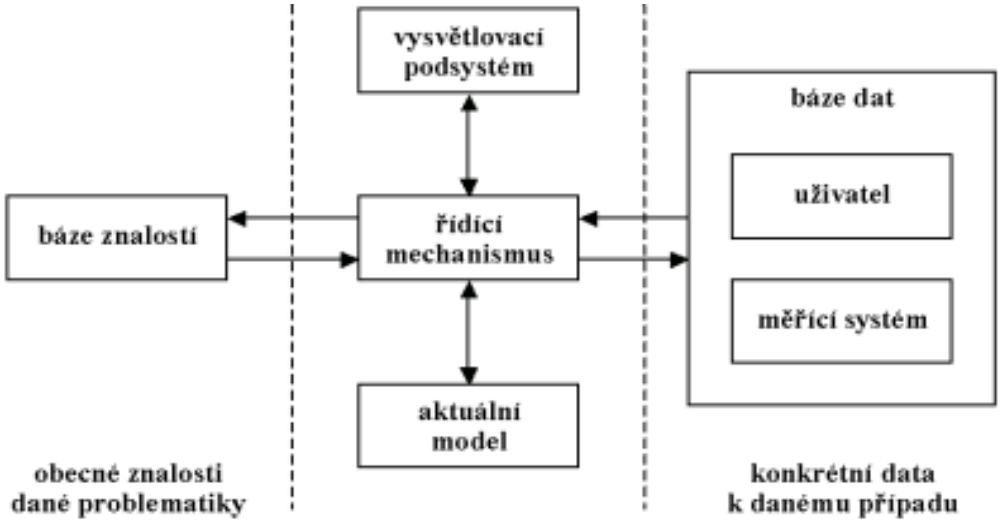
Podle charakteru řešené úlohy rozeznáváme expertní systémy (Pokorný 2004):

- diagnostické
- plánovací
- hybridní
- prázdné ES - ať již diagnostické či plánovací – bez problémově závislých částí, tj. bez báze znalostí a bez báze dat. Doplněním báze znalostí k prázdnému systému se systém teprve orientuje na řešení příslušné problematiky. Dodáním báze dat je pak vždy řešen konkrétní případ.

Diagnostické expertní systémy

Úlohou diagnostických expertních systémů je provádět efektivní interpretaci dat s cílem určit, která z hypotéz z předem stanovené konečné množiny cílových hypotéz nejlépe koresponduje s reálnými daty týkající se daného konkrétního případu. U diagnostických systémů tedy řešení případu probíhá formou postupného ohodnocování a přehodnocování dílčích a cílových hypotéz v rámci víceméně pevně daného vnitřního modelu řešeného problému, který je předem navržen expertem.

Struktura diagnostického expertního systému je uvedeno na obrázku 3.1. Jádrem tohoto systému je řídící mechanismus, který využitím báze znalostí a báze dat po každé odpovědi upřesňuje *aktuální model* konzultovaného případu. *Řídící mechanismus* je odpovědný za výběr dotazu, od jehož zodpovězení očekává největší přínos k upřesnění aktuálního modelu, a za úpravu aktuálního modelu po obdržení odpovědi. Báze dat může být obecně tvořena jak přímými odpověďmi uživatele, tak i hodnotami automaticky odečtenými z měřících přístrojů. *Vysvětlovací mechanizmus* sděluje uživateli informace o průběhu konzultace a především závěr celé konzultace. Jde tedy pouze o oddělení uživatelského rozhraní od vlastního řídícího mechanizmu.

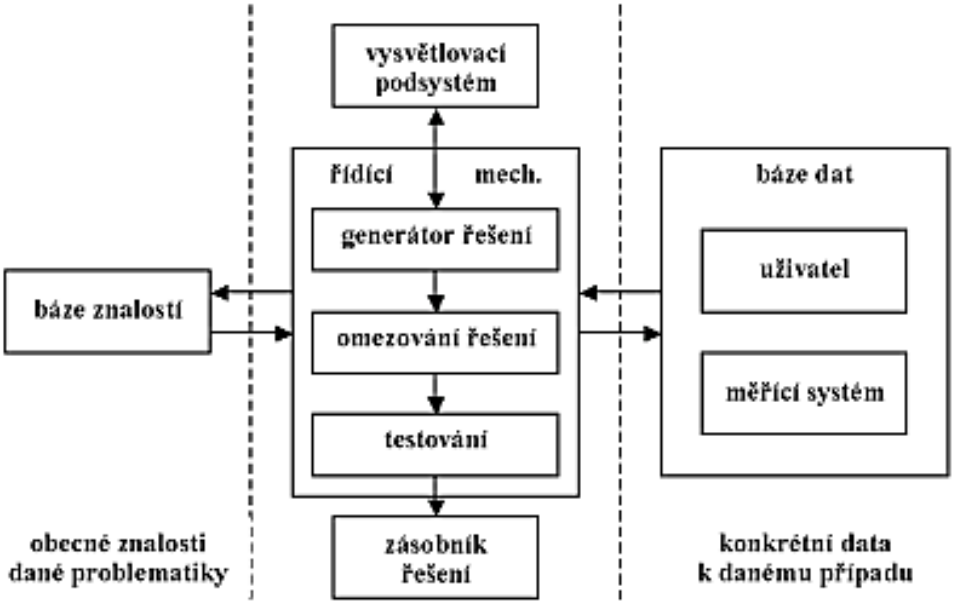


Obrázek 3.1: Struktura diagnostického expertního systému

Plánovací expertní systémy

Plánovacími expertními systémy jsou obvykle řešeny takové úlohy, kdy je znám cíl řešení a počáteční stav a systém má využitím dat o konkrétním řešeném případu nalézt posloupnost kroků (operátorů), kterými lze cíle dosáhnout.

Struktura plánovacího expertního systému je podobná jako v případě diagnostického, odlišné je pouze uspořádání střední, inferenční části. Struktura plánovacího expertního systému je uvedeno na obrázku 3.2. Podstatnou částí takovýchto expertních systémů je *generátor možných řešení*, který automaticky kombinuje posloupnost kroků. Lze ukázat, že s rostoucím počtem operátorů, a především s nutným počtem kroků řešení, roste velmi rychle počet kombinací při vytváření posloupnosti kroků – hovoří se o „kombinatorické explozi“. Znalost experta i data o reálném případu jsou v plánovacích expertních systémech užívány k výraznému omezení kombinatorické exploze zkoumaných řešení, navrhovaných generátorem. Výsledkem činnosti plánovacího expertního systému je nabídka ohodnocených přípustných řešení, z nichž si uživatel podle svých kritérií vybere jednu dominantní. Tento seznam je uchováván v *zásobníku vhodných řešení*, který se v průběhu řešení úlohy dynamicky mění.



Obrázek 3.2: Struktura plánovacího expertního systému

Hybridní systémy

Vyznačují se kombinovanou architekturou, poněvadž částečně využívají principů diagnostických, částečně plánovacích systémů. K hybridním systémům řadíme například inteligentní výukové systémy či systémy monitorovací. Od samého počátku rozvoje expertních systémů byla zdůrazňována opakovatelná využitelnost vyvíjených expertních systémů či jejich komponent. Vysoká znovupoužitelnost je patrná zejména u obecných problémově nezávislých částí expertních systémů, tj. u řídících a vysvětlovacích mechanismů.

Prázdné expertní systémy

Expertní systémy – ať již diagnostické či plánovací – bez problémově závislých částí, tj. bez báze znalostí a bez báze dat. Doplněním báze znalostí k prázdnému systému se systém teprve orientuje na řešení příslušné problematiky. Dodáním báze dat je pak vždy řešen konkrétní případ.

Expertní systémy se hodí pro řešení úloh různých typů (Dvořák 2004):

1. *Diagnostické úlohy*

   Diagnostický proces lze chápat jako získávání a interpretaci

informací relevantních pro potvrzení přítomnosti nebo nepřítomnosti nějaké závady v systému. Cyklus stanovení diagnózy je tvořen třemi kroky: formulování hypotézy, testování hypotézy a přijetí nebo zamítnutí hypotézy.

2. *Generativní úlohy (plánování, návrh, predikce aj.)*
Řešení je u generativních úloh vytvářeno (poskládáno) z dílčích komponent, kdy se hypotézy generují dynamicky až v průběhu konzultace. Cílem je sestavit sekvenci akcí, která povede k řešení požadovaného úkolu. Plánování je spíše chápáno jako splňování omezení, kladených na požadované řešení úlohy. Když je k dispozici dostatek znalostí pro stanovení, co se má v dané chvíli provést, se postupuje způsobem „navrhni a aplikuj“. Když jsou dostupné znalosti neúplné, postupuje se způsobem „navrhni a reviduj“, který umožňuje navrácení k předcházejícím variantám řešení.

3.3 Znalostní inženýrství

Znalostní inženýrství je ta část umělé inteligence, která se zabývá metodami a technikami získávání, formalizace, kódování, uchovávání, testování a udržování znalostí. Zabývá se tedy veškerými činnostmi, jejichž cílem je naplnění znalostních systémů tím nejdůležitějším - znalostmi. V literatuře se můžeme setkat také s pojmem znalostní systém (*knowledge-based system*), který je podle staršího pojetí obecnější než pojem expertní systém. Expertní systém tedy lze chápat jako zvláštní typ znalostního systému, který se vyznačuje používáním expertních znalostí a některými dalšími rysy, jako je např. vysvětlovací mechanismus. V poslední době však dochází ke stírání rozdílů mezi těmito pojmy.

Znalostní inženýrství je považováno za klíčovou oblast umělé inteligence, protože kvalita znalostí má rozhodující vliv na efektivitu znalostního (expertního) systému a proto musí být tvorbě báze znalostí

věnována mimořádná pozornost. Znalostní inženýrství má mnoho společných rysů se softwarovým inženýrstvím. Odlišnosti se týkají typu, povahy a množství reprezentovaných znalostí. U softwarového inženýrství se jedná o dobře definované algoritmické znalosti. Povahu a množství těchto znalostí potřebných pro řešení daného problému lze předem dobře odhadnout. U znalostního inženýrství se jedná o extenzivní, nepřesné a špatně definované znalosti, jejichž povahu a množství lze předem velmi špatně odhadnout. To způsobuje potíže v počátečních etapách vývoje expertních systémů při odhadu potřebného úsilí a při tvorbě jeho návrhu.

Kontrolní otázky:

1. Co je to expertní systém?
2. Jaké jsou charakteristické rysy expertních systémů?
3. Jaké jsou typy expertních systémů?
4. Jaké typy úloh můžeme expertními systémy řešit?

Úkoly k zamyšlení:

Přemýšlejte, kde by jste mohli využít expertní systém, k čemu by sloužil a jakého typu by expertní systém byl?

Korespondenční úkoly

1. Vyberte si libovolný expertní systém (nejlépe volně šiřitelný), vytvořte a otestujte vlastní bázi znalostí na zvolený problém.
2. Nalezněte na Internetu možnosti využití expertních systémů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran

Shrnutí obsahu kapitoly

Tato kapitola představuje stručný úvod do problematiky expertních systémů. Dozvěděli jste se co je to expertní systém, jaké jsou jeho charakteristické rysy a jaké typy úloh zpracovává. V závěru kapitoly jsme se zmínili o znalostním inženýrství.

Pojmy k zapamatování

- Expertní systém,
- inferenční mechanismus,
- báze znalostí,
- znalostní inženýrství

4 Multiagentové systémy

V této kapitole se dozvíte:

- Co se rozumí pod pojmem agent.
- Jaké typy agentů rozlišujeme.
- Jak probíhá koordinace, kooperace a komunikace agentů v multiagentovém systému.

Po jejím prostudování byste měli být schopni:

- Vysvětlit pojmy agent a multiagentový systém.
- Charakterizovat jednotlivé typy agentů a jejich interakce v multiagentovém systému.

Klíčová slova této kapitoly:

Distribuovaná umělá inteligence, multiagentové systémy, koordinace, kooperace a komunikace agentů.

Průvodce studiem

Tato kapitola představuje úvod do problematiky multiagentových systémů. Seznámíte se zde s pojmem agent, s architekturou agenta a se základní typologií agentů. Zatímco problematiku jednotlivých agentů bylo možné zjednodušeně vnímat jako problém strojového učení a řešení úloh, problematika multiagentových systémů přinesla do umělé inteligence sociální témata týkající se vytváření skupin na základě společných zájmů. Aby agenti byli schopni spolupráce, musí mít prostředky pro vzájemnou komunikaci. To znamená, že musí mít společný jazyk, společné vnímaní pojmů a musí mít dána pravidla vzájemné komunikace.

Problematika agentů a multiagentových systémů vychází z oblasti zvané distribuovaná umělá inteligence, ze které se postupně vyděluje jako samostatná disciplína, opírající se o výsledky výzkumu v oblasti počítačových věd i z ostatních částí umělé inteligence. Distribuované řešení úloh reprezentuje tu třídu problémů, která se zabývá hledáním možností, jak pro pevně zvolenou úlohu vhodně rozdělit práci na jejím řešení mezi více modulů (uzlů) a jak následně mohou jednotlivé moduly sdílet informace o postupu vzniku cílového řešení. Studiem chování skupin volně propojených autonomních systémů (agentů), které spolupracují v zájmu nějakého společného cíle se zabývají multiagentové systémy.

4.1 Agent

Agent je entita zkonstruovaná za účelem kontinuálně a do jisté míry autonomně plnit své cíle v adekvátním prostředí na základě vnímání prostřednictvím senzorů a prováděním akcí prostřednictvím aktuátorů. Agent zároveň ovlivňuje podmínky prostředí tak, aby se přibližoval k plnění cílů (Kubík 2000).

O agentovi obecně platí, že:

- může posílat a přijímat informace od jiných agentů za použití vhodných protokolů.
- může zpracovávat přijaté informace a uvažovat o nich (tj. provádět odvozování, syntézu i analýzu).
- má soubor schopností provádět akce, které se mohou i dynamicky měnit (tj. akce charakterizují úkoly, které dokáže agent provádět).

Inteligentní agent je entita, která je zodpovědná za rozhodování, zda a jak reagovat na externí podněty.

Inteligentní agent má schopnost plnit cíle s využitím logické dedukce, tj. navíc umí:

- uvažovat o svých schopnostech a schopnostech ostatních agentů.
- generovat cíle nebo plány pro sebe a jiné agenty.

- účastnit se složitých interakcí s ostatními (např. za účelem vyjednávání a delegování úkolů), dynamicky se zapojovat do skupin či organizací, které mohou například sdružovat agenty s podobnou funkcí, nebo také skupiny opouštět.
- získávat informace a používat jejich zdroje a udržovat explicitní modely důvěry pro sebe a ostatní agenty.

Agenty lze rozdělit následovně (Kubík 2000):

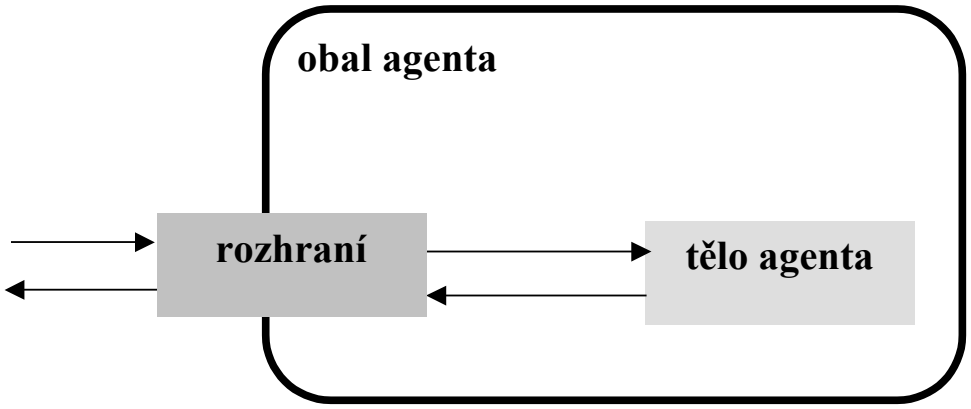
1. Biologičtí agenti – lidé.
2. Techničtí agenti – roboti.
3. Programoví agenti – softboti
   - agenti v počítačových hrách,
   - počítačové viry,
   - agenti pro specifické úkoly,
   - agenti – entity umělého života.

Funkcionalita agenta má svá ohraničení a každý agent je schopen jenom takového chování, které lze dosáhnout přizpůsobením jeho senzorů (např. kamery u robotů nebo příkazy pro zjištění obsahu adresáře u softwarových agentů) a aktuátorů (např. kolečka nebo ramena u robotů nebo příkaz na smazání souborů u softwarových agentů) prostředí.

4.2 Architektura agenta

Agent má obvykle následující strukturu (viz obr. 4.1):

- *Obal*, který je zodpovědný za plánování a realizaci sociálních interakcí a tvoří jej:
  - komunikační vrstva;
  - model sociálního chování.
- *Vlastní tělo*, jež nemá informace o komunitě.



Obrázek 4.1: Architektura agenta (Netrvalová 2004).

Individuální agent může být charakterizován podle toho, zda a na jaké úrovni je schopen zvažovat rozličné varianty řešení svého cíle.
Základní charakteristiky agenta jsou (Kubík 2000):

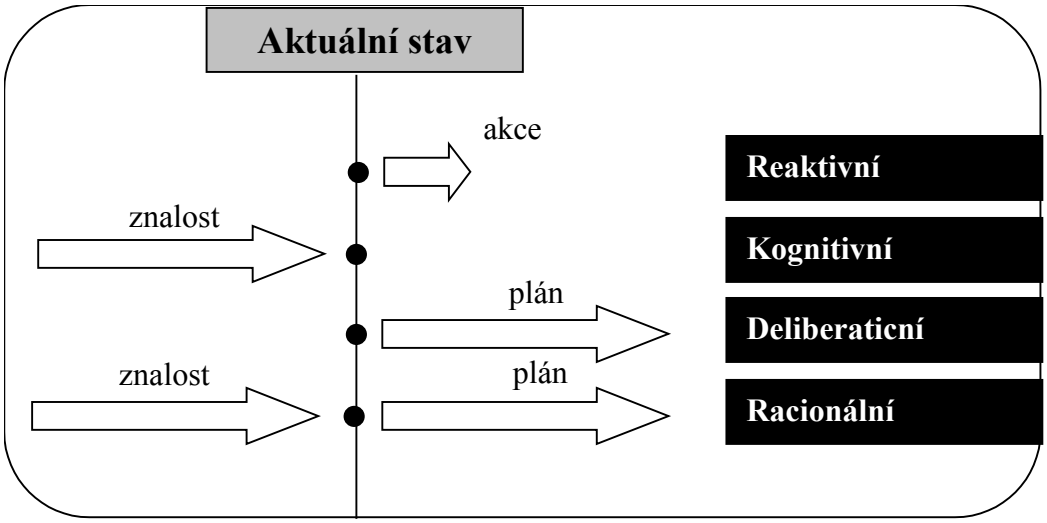
- *autonomie*

  tj. vlastnost agenta, spočívající v samostatnosti rozhodování o svém chování v rámci daného systému, bez implicitní závislosti na jakýchkoliv jiných prvcích tohoto systému. Autonomie je pravděpodobně jediným styčným bodem mezi mnoha přístupy k pojmu agent.
- *reaktivita*

  tj. vlastnost agenta, spočívající v jeho reakci na změny prostředí tak, aby dosáhl cíle, pro který byl navržen.
- *intencionalita*

  tj. agenti jsou schopni mít uloženy v paměti dlouhodobé cíle, organizace chování k dosahování těchto cílů, formulace vlastních plánů a využití svých úsudků.
- *schopnost sociálního chování*

  tj. agenti jsou schopni spolupráce pro dosažení společných cílů, udržování informace o jiných agentech a vytváření úsudků o nich, sdružování do koalic a týmů, od nichž očekávají vzájemný prospěch.



Obrázek 4.2: Základní rozdělení agentů (Netrvalová 2004).

Agenty lze dále rozlišit na následující typy (viz obr. 4.2):

- *reaktivní*

  Tento agent bezprostředně reaguje na jisté změny prostředí (nebo své změny vůči prostředí), aniž by měl vnitřní reprezentaci znalostí o tomto prostředí. Jeho reakce nejsou výsledkem výpočtů či dedukcí na základě znalostí, ale pouze reakcemi na podněty. Reaktivita je vedle autonomity druhou významnou vlastností agentů.

- *deliberativní (rozvážný)*

  Na rozdíl od reaktivního agenta má deliberativní (rozvážný) agent schopnost plánovat postup svých akcí vedoucích k dosažení zvolených nebo zadaných záměrů/cílů. To znamená, že agent musí mít schopnost různých výpočtů, což bude později v textu chápáno jako druh vnitřní činnosti agenta. K dosažení svých záměrů pak agent ovlivňuje okolní prostředí tak, aby získal nějakou výhodu. Toto jednání je další z často uváděných vlastností agentů a nazývá se proaktivita.

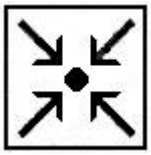
- *kognitivní*

  Kognitivní agent má schopnost vyvozovat logické závěry ze svých pozorování okolního prostředí. Takový agent musí především být schopen se učit a vytvářet si svou vlastní bázi znalostí. Do ní si během svého působení ukládá informace získané interakcí s

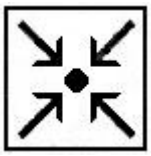
okolím nebo znalosti získané dedukcí. Kognitivní agent nemusí mít nutně deliberativní schopnosti. Pak provádí pouze vnitřní akce, například analyzuje scénu, provádí překlad, nebo získává znalosti (dolování znalostí z dat).

- *racionální*

  Racionální agent má všechny výše uvedené vlastnosti a jeho struktura obsahuje jak plánovací jednotku, tak i kognitivní jednotku včetně báze znalostí. Je to agent, který je na základě svých poznatků schopen se učit a pak plánovat svoji činnost tak, aby dosáhl svých cílů racionálním způsobem. Stojí nejvýše na pomyslné hierarchii uvedených agentů (viz obr. 4.2).



Příkladem *reaktivního* agenta je mechanická beruška, která má senzory – tykadla a aktuátory – kolečka. Prostředím berušky je rovná plocha stolu. Jediným relevantním stimulem je zachycení okraje desky stolu, kdy se poloha jednoho z tykadel sníží, čímž se malé kolečko dostane na plochu stolu a způsobí otočení berušky doprava. Beruška se tak vyhne pádu přes okraj stolu. Toto chování je v jejím jednoduchém prostředí racionální, i když o tom sama beruška nemá „představu“ a nevyužívá žádné reprezentace prostředí k jeho dosažení.



Problematikou budování inteligentních systémů zdola nahoru, tj. od jednodušších agentů ke stále komplikovanějším se jako první zabývala M. Matric (Mataric 1994). Nejprve sestrojila jednoduchého robotnického agenta, vybaveného sonary rozloženými po obvodu jeho válcovitého těla pro vnímání prostředí a s přiměřenou motorikou, který se dokázal pohybovat podél stěn a vyhýbal se při svém pohybu překážkám. Takto vzniklý systém doplnila tak, aby byl schopen zapamatovat si určité charakteristiky stavů prostředí, ve kterých se právě nacházel a následně rozeznal body, které již dříve navštívil. Postupně by měl agent z takovýchto bodů sestavovat mapu svého prostředí, kterou by si měl pamatovat, tj. postupně si vytvářet vnitřní reprezentaci svého prostředí, přičemž využíval tuto mapu k efektivnější navigaci. Robot, který byl vybudován na základě této architektury byl nazván Toto. Robot Toto byl

svým chováním o krok blíže k autonomnímu chování lidí. Dokázal cíleně sledovat vytyčenou cestu na základě zaznamenání historie rozložení objektů v prostoru a také její dynamické modifikace. Tuto reprezentaci si vytvářel sám, avšak její jednoduchost na úrovni symbolických značek záchytných bodů v místnostech mu neumožňovala projevovat se jinak, než bezkolizním pohybem v komplexu místnosti. Dále nebylo ani vyřešeno, jak ale zabezpečíme, aby se agent dokázal rozhodnout, kterou část úkolu bude právě plnit, jaké prostředky na splnění cílů použije a jak bude postupovat při plnění cílů. Nelze také *deliberativního* agenta naučit zlepšovat své chování a funkcionalitu.

4.3 Multiagentové systémy

Multiagentové systémy (MAS Multi-Agent Systems,) jsou takové systémy, kde se v prostředí pohybuje více než jeden agent. Musí tudíž nutně docházet ke spolupráci agentů a je proto zapotřebí vytvořit prostředky pro vzájemnou komunikaci.
V multiagentových systémech musí mít agenti společné následující schopnosti a zájmy:

- vnímání pojmů,
- pravidla vzájemné komunikace,
- jazyk.

Agenti mohou navzájem ovlivňovat svoje chování, tj. svou činnost mohou provádět v souladu s ostatními nebo pouze nenarušovat činnost ostatních nebo působit proti ostatním. Dále můžeme agenty dělit na (Netrvalová 2004):

- *kooperativní agenty*
  mající společné cíle, tj. každá akce, která je vykonána jedním agentem je v souladu se záměry druhého agenta.
- *kompetitivní agenty*
  mající protichůdné cíle, tj. snaha jednoho agenta k dosažení jeho cíle jde přímo proti snaze agenta druhého.

- *kolaborativní agenty*
  tj. navzájem spolupracující agenty.

Častější jsou však případy, kdy agenti mají své partikulární zájmy, které sice nejsou stejné, ale nejsou ani v přímém rozporu se zájmy ostatních agentů. Během činnosti agentů pak může docházet k souladu nebo kolizím jejich záměrů.

Koordinace je proces probíhající v multiagentových systémech, kterým se dosahuje propojení jednotlivých komponent v systému za účelem řešení problému. Koordinační procesy lze rozdělit do dvou základních kategorií na procesy s centralizovaným a decentralizovaným řízením. Pokud je společenství agentů řízeno jedním centrálním prvkem nebo malou skupinou částečně decentralizovaných komponent, vzniká v něm *kooperace*. Každý agent má přesně určenou roli, kterou musí plnit a také vztahy s ostatními agenty k tomu, aby bylo dosaženo globálního cíle. Říkáme, že agenti mají protokol pro kooperaci. Protokolu se využívá za účelem efektivní *komunikace*, ke které v rámci kooperace dochází. Jedná se o přesně stanovené sledy zpráv, které určují, jak lze odpovědět na přijatou zprávu. Agenty mohou komunikovat i bez takto specifikovaného protokolu, což je sice mnohem sofistikovanější, ale také mnohem složitější způsob komunikace.
Kooperace je vyšší proces než koordinace, protože kooperaci nelze bez procesu koordinace dosáhnout, ale zároveň koordinace mezi agenty nemusí nutně vést ke kooperaci. Komunikace je nutným předpokladem kooperace, jejímž speciálním případem je kolaborace s pevnějšími vazbami a závazky mezi agenty.

Strategie agenta udává, která akce z báze akcí bude vykonána jako reakce na aktuální stav prostředí. U reaktivního agenta vyplývala následná akce automaticky z ohodnocení prostředí, kdežto v případě multiagentového systému je strategie agenta volena z množiny strategií podle jeho vlastního rozhodnutí. Strategie skupiny agentů je množina strategií jednotlivých agentů. Nejjednodušší situace nastává, když si

každý agent volí svou strategii bez ohledu na strategie, které si zvolili ostatní agenti. Pak je zřejmě jeho záměrem zvolit takovou strategii, která je pro něho optimální. Pokud existuje strategie, která je pro agenta nejlepší nezávisle na strategiích zvolených ostatními agenty, je tato strategie nazývána dominantní strategií. Racionální agent pak vždy volí tuto dominantní strategii. Dominantní strategii lze nalézt například v problému známém pod názvem vězňovo dilema (Prisoner's dilema).

Příklad (Netrvalová 2004)**.**

Dva agenti A (např. Alice) a B (např. Bob) byli chyceni nedaleko místa loupeže a oběma hrozí za tento čin deset let vězení. Oba jsou vyslýcháni zvlášť a je jim řečeno, že když se přiznají, bude jim přiznána polehčující okolnost a dostanou jen pět let. Když se nepřiznají, dostanou stejně rok vězení za to, že u nich byl nalezen ukradený prsten. Pokud se však přizná jen jeden z nich a bude svědčit proti druhému, bude za to propuštěn. Následující tabulka udává možnosti pro oba agenty.

Tabulka 4.1: Vězňovo dilema.

	Alice se přizná	**Alice se nepřizná**
Bob se přizná	A = -5; B = -5	A = -10; B =0
Bob se nepřizná	A = 0; B = -10	A = -1; B = -1

Je zřejmé, že výsledek strategie jednoho agenta nezávisí na zvolené strategii druhého agenta (pro Boba je výhodnější se přiznat, nezávisle na tom, co udělá Alice a totéž platí pro Alici) a proto v tomto případě existuje dominantní strategie pro oba obviněné.

Kontrolní otázky

1. Co rozumíme pod pojmem agent?
2. Jaké typy agentů rozlišujeme?
3. Jak probíhá koordinace, kooperace a komunikace agentů v multiagentovém systému?

Korespondenční úkoly

1. Nalezněte na Internetu další možné charakteristiky a typy agentů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran
2. Nalezněte např. na Internetu možné příklady použití multiagentových systémů při řešení úloh. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

V této kapitole jste se seznámili s pojmem agent, s architekturou agenta a se základní typologií agentů. Dále jsme se věnovali multiagentovým systémům. Aby agenti byli schopni kooperace, musí svá jednání koordinovat, a proto spolu musí komunikovat a mít i vůli ve prospěch celé skupiny. Kooperace je vyšší proces než koordinace, protože kooperaci nelze bez procesu koordinace dosáhnout a zároveň nutným předpokladem kooperace je komunikace.

Pojmy k zapamatování

- distribuovaná umělá inteligence
- agent
- multiagentové systémy
- koordinace, kooperace a komunikace agentů

5 Umělý život

V této kapitole se dozvíte:

- Co je umělý život.
- Jaké jsou projevy umělého života.
- Jak lze umělý život modelovat.

Po jejím prostudování byste měli být schopni:

- Objasnit, co je to umělý život.
- Objasnit, jaké jsou projevy umělého života.
- *Objasnit, jak lze umělý život modelovat.*

Klíčová slova této kapitoly:

Umělý život, celulární automat, life, Wolframův celulární automat.

Průvodce studiem

Stejně tak jako definice umělé inteligence naráží na problém neexistence všeobecně přijatelné definice inteligence přirozené, naráží i definice umělého života na nejednotnost definice života existujícího na Zemi. Navíc otázky života vyvolávají v porovnaní s otázkami inteligence ještě více pochybností etických, filozofických i náboženských. V této kapitole se pokusíme ozřejmit, co považujeme za umělý život a jaké jsou jeho projevy.

5.1 Co je to umělý život

Pro potřebu této kapitoly se asi nejlépe hodí definice (Bonabeau 1995): Umělý život je všeobecná metoda, podstatou které je generovat z jednoduchých mikroskopických spolupracujících prvků takové chování

na úrovni makroskopické, které je možno interpretovat jako projev života.

Základní vlastnosti umělého života jsou následující:

- Podstatou umělého života je *informace* a ne materiální forma, která slouží k jejímu uchovávání a zpracování. Umělý život v podstatě představuje systém pro zpracování informací.
- Život vyžaduje jistou míru *složitosti*. Struktura po dosažení určité složitosti je schopna vytvářet svoje identické kopie a také složitější potomky.
- Informace má dvojí podobu:
- neinterpretovaná - *genotyp* - slouží zejména k rozmnožování; interpretovaná - *fenotyp* - podle něj se vytváří struktura nového jedince.
- Prostředkem *evoluce* tj. vývoje směrem ke složitějším a dokonalejším strukturám je samoreprodukce, mutace a selekce.
- Syntetický proces vývoje života probíhá zásadně směrem zdola nahoru, tj. od elementárních primitiv k složitým strukturám vykazujícím složité chování.
- Vzájemné lokální spolupůsobení primitiv vyvolává na globální úrovni úplně nové fenomény – tento jev je označován jako *emergence*. Vše přitom probíhá bez jakéhokoliv centrálního řízení.

Jednou z podmínek emergence je nelineární chování elementárních primitiv, tedy neplatnost principu superpozice. *Princip superpozice* předpokládá že chování celku lze odvodit složením dílčích chování primitiv.

5.2 Modely umělého života

Kinematický model

První koncept emulace umělého života vytvořil von Neumann a ve svém kinematickém modelu (Neumann 1966) se soustřeďoval

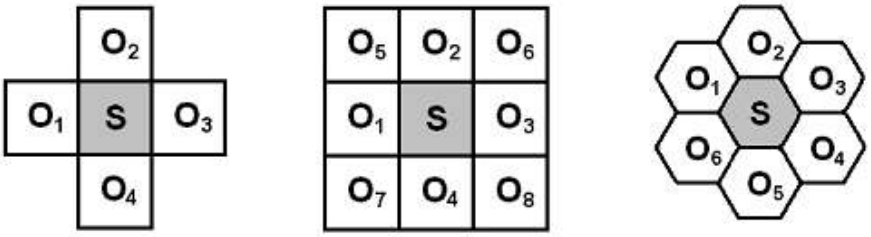
především na *samoreprodukci*, která je vlastní životu. Vytvořil koncept samoreprodukujícího se automatu složeného z několika prvků:

- *Manipulátor* – analogie ruky řízené z centra.
- *Oddělovač* – rozpojuje dva prvky (řízen z centra).
- *Spojovač* – spojuje dva prvky (řízen z centra).
- *Senzor* – rozpoznává součástky a informuje centrum.
- *Nosníky* – kostra struktury a paměťové médium.

Automat žije v prostředí náhodně rozložených součástek a na příkaz centra shromažďuje součástky a skládá z nich svou kopii. Na konci sestavený automat oživí nahráním vlastní kopie své paměti do nového jedince. Tomuto navrženému modelu se říká *kinematický model*. Problémem tohoto modelu je jeho přílišná obecnost. Známe sice jeho funkce, ale neznáme vnitřní stavbu jeho součástek (jde o tzv. black-box problém).

Celulární automaty

Celulární automat (dále jen CA) je dynamický systém, který je diskrétní v prostoru i čase. Je tvořený pravidelnou strukturou buněk v *n*-rozměrném prostoru. Nejčastěji je *n=2*, tj. buňky tvoří čtvercovou mřížku. Každá buňka může nabývat jeden z *K* možných *stavů*. Často se jedná pouze o dva stavy: *0* - mrtvá buňka, *1* - živá buňka; v tomto případě se občas stav *1* označuje jako buňka a *0* jako prázdné políčko (mřížky). Hodnoty stavů buněk v dalším časovém kroku (v následující generaci) se vypočítávají *paralelně* na základě *lokální přechodové funkce* (stejné pro všechny buňky). Argumenty této funkce jsou aktuální hodnoty stavů uvažované buňky a všech sousedních buněk v jejím okolí. V případě, že *n=1,* je *okolí* tvořené tzv. poloměrem - počtem sousedů po obou stranách uvažované buňky; v případě, že *n=2*, tvoří okolí čtyři přilehlé buňky k dané buňce (tzv. *neumannovské okolí*) nebo se do něj zahrnou i čtyři další sousední buňky, které se dotýkají vyšetřované buňky jen v rozích (tzv. *úplné okolí*). Mezi nejpoužívanější okolí můžeme proto zařadit následující (viz obr. 5.1):



neumannovské okolí úplné okolí šestiúhelníkové okolí

Obrázek 5.1: Okolí vyšetřované buňky S.

Lokální přechodová funkce *f* definuje stav buňky v čase *t+1* pro okolí buňky ***S*** následovně: *S*(*t+1*) = *f*(*S*(*t*), *O1*(*t*), *O2*(*t*), *O3*(*t*), ...).

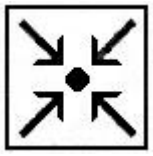
Lokální přechodová funkce bývá často definovaná sadou pravidel, jež mohou být zadané slovně (například v případe hry LIFE), nebo graficky (Mařík 2000). Zpravidla předpokládáme, že struktura buněk je nekonečná. V praktických aplikacích se krajní buňky uvažují buď identicky za nulové (prázdné), nebo předpokládáme, že jejich okraje jsou "propojené" a tvoří v případě *n=1* smyčku a v případě *n=2* anuloid. Některé z *K* možných stavů jsou označované jako *klidové*, tj. pokud je vyšetřovaná buňka v klidovém stavu, má ve svém okolí pouze buňky v klidovém stavu, potom se hodnota jejího stavu v dalších generacích nemění.

Celulární automat charakterizují tři klíčové vlastnosti (Mařík 2000):

- *Paralelizmus*
  výpočet hodnot nových stavů všech jeho prvků probíhá současně (na běžných sériových počítačích se musí tento postup simulovat).
- *Lokalita*
  nový stav prvku závisí pouze na jeho původním stavu a na původních stavech prvků z jeho okolí.
- *Homogenita*
  pro všechny prvky platí stejná přechodová funkce.

Celulární automaty se mohou používat jako modely pro biologické, fyzikální a společenské procesy, neboť každá živá buňka (element, jedinec apod.) mění svůj stav souběžně s ostatními buňkami (*paralelizmus*), v závislosti na stavu svého okolí (*lokalita*) a na základě stejných zákonitostí (*homogenita*).

Příklad: Hra LIFE



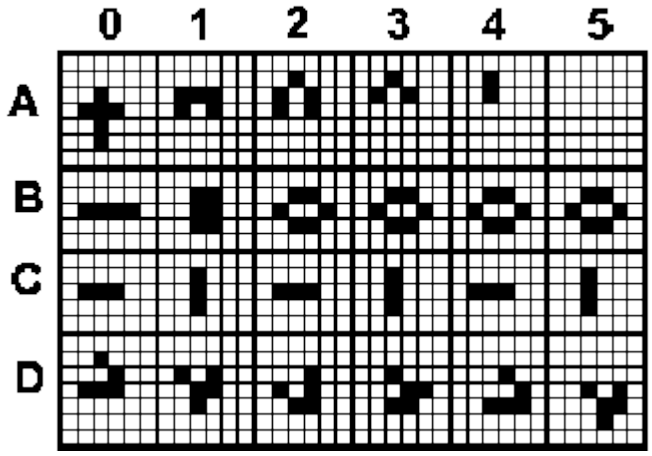
John Horton Conway pracoval na sestavení neumannovského *2D* CA v mnohem jednodušší podobě. Pracoval pouze se dvěma stavy (dále budou označované jako buňka a prázdné políčko) a s úplným okolím (tj. buňka s osmi sousedy umožňuje vygenerovat $2^9 = 512$ pravidel). Dlouho experimentoval s vhodnou lokální přechodovou funkcí, definující pravidla pro zrod, přežití a uhynutí buňky. Nakonec našel pravidla, která zaručovala dynamiku obrazců, tvořených populacemi buněk:

- *Zrod*
  tj. v okolí prázdného políčka jsou právě tři buňky - "trojpohlavní" rozmnožování.
- *Přežití*
  tj. v okolí buňky jsou dvě nebo tri další buňky.
- *Uhynutí*
  tj. v okolí buňky je 0, 1, 4, 5, 6, 7 nebo 8 dalších buněk.

Výsledná sada pravidel poskytovala také přijatelnou biologickou interpretaci, např. uhynutí příliš osamocené buňky, ale také buňky v přehuštěné populaci. Navržený celulární automat byl nazvaný hrou života - LIFE a ta se stala po svém zveřejnění velmi populární. Výchozí obrazce, tvořené populacemi buněk, směřovaly obvykle po několika generacích k jedné z následujících situací:

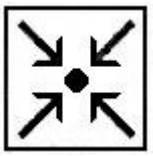
- *Zánik* (struktura A na obr. 5.2).
- *Stabilní obrazec,* tj. v dalších krocích neměnný (struktura B na obr. 5.2).
- *Cyklicky se opakující obrazec* (struktura C na obr. 5.2).

- *Cyklicky se opakující, ale posunutý obrazec.* Struktura D na obr. 5.2 je tzv. *kluzák* - ten se v každé další čtvrté generaci objeví ve stejném tvaru, ale posunutý o jedno políčko po úhlopříčce. Avšak cyklicky se opakujících posunutých obrazců je v LIFE mnohem více (Gardner 1970): kombajn, loď apod.



Obrázek 5.2: Struktury ve hře LIFE.

William Gosper našel tzv. *kluzákové dělo*, které produkuje v každé 30-té generaci jeden kluzák. Kluzáky jsou nejmenší pohyblivé struktury, které mohou být generovány jinými strukturami a jejichž kombinace jsou schopné vytvářet zajímavé procesy (např. kombinace osmi kluzáků vytvoří kluzákové dělo). Celý počítač v LIFE nebyl nikdy sestaven, ale byla v něm sestrojena funkční sčítačka. Největším problémem však bylo, že se v LIFE nepodařilo nalézt konečnou samoreprodukující se strukturu. Odhaduje se však, že je možné na mřížce obsahující 106 políček vytvořit strukturu srovnatelnou s jednobuněčným živým organizmem.



Příklad: Wolframův celulární automat

Nový impulz do výzkumu celulárních automatů přinesl Stephen Wolfram (Wolfram 1984). Wolfram studoval vlastnosti jednorozměrných celulárních automatů (*1D CA*): jejich výhodou byl poměrně malý počet možných pravidel a názorná reprezentace posloupností generací v řádcích pod sebou. Za nejjednodušší případ lze považovat dvojstavový systém, kde okolí tvoří pouze dvě sousední buňky. Nová

hodnota buňky je pak určena trojicí starých hodnot a může mít osm podob, můžeme proto této osmici přiřadit *28* výstupních kombinací. Výsledný počet všech možných kombinací pravidel je zde *256*. Na obrázku 5.3 jsou uvedena pravidla ve své grafické podobě (trojice buněk tvoří okolí a pod ní je uvedena nová hodnota buňky). Obvykle se udává číslo sady pravidel jako dekadický ekvivalent binárně interpretovaného vektoru nových hodnot.

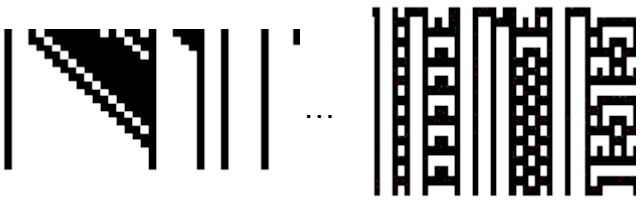
Obrázek 5.3: Pravidla Wolframova automatu v grafické podobě.

Dynamiku *1D* automatu je možné dobře znázornit tak, že v prvním řádku je výchozí stav automatu (čas *t=0*) a pod ním se postupně zakreslují stavy v časech *t=1, 2, 3,...* Wolfram rozdělil těchto *256* možných kombinací pravidel celulárního automatu do čtyř skupin podle složitosti chování:

1. *CA1* (obr. 5.4)
   celý řádek se rychle vyprázdní (sada 40) nebo zaplní.

Obrázek 5.4: CA1.

2. *CA2* (obr. 5.5)
   počáteční aktivita se postupně utlumuje a začínají převládat stabilní shluky (sada 228), viz obr. 5.5(a); případně jednoduché, cyklicky se opakující struktury (sada 109), viz obr. 5.5(b).

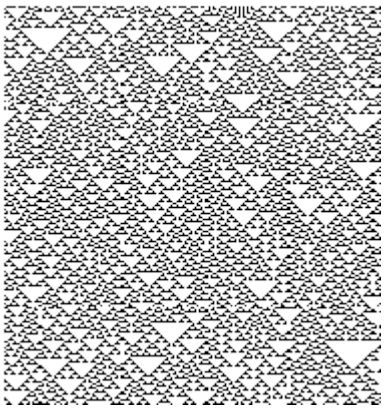


(a) (b)

Obrázek 5.5: CA2.

3. *CA3* (obr. 5.6)
   převládá zdánlivě chaotický vývoj, kde pouhým okem nelze

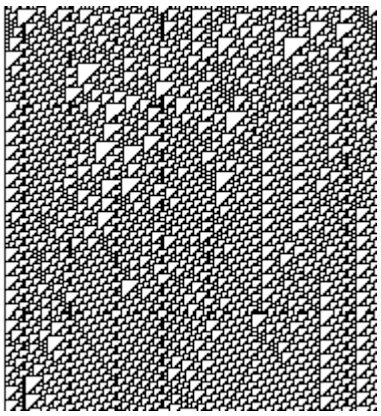
pozorovat nějakou pravidelnost. Obrazce zde působí jako náhodný šum (sada 22).



Obrázek 5.6: CA3.

4. **CA4** (obr. 5.7)

   vykazují složitou pravidelnost. Generují se nové - obvykle *posuvné* struktury (např. kluzáky v LIFE), které „žijí“ poměrně dlouho (sada 110). Tyto celulární automaty jsou schopné realizovat univerzální počítač.



Obrázek 5.7: CA4.

Ve všech uvedených příkladech proces začínal náhodně generovaným řádkem. Experimenty byly zaměřeny na schopnost celulárního automatu přenášet informaci, protože přenos a uchovávání informace jsou základními rysy života. První dva režimy lze charakterizovat jako nepříznivé pro existenci života. Třetí režim pak reprezentuje „existenci života na pokraji chaosu“. Navazuje tak na názor von Neumanna, že existuje jistá kritická hranice složitosti, pod kterou jsou procesy syntézy

degenerativní, ale nad kterou má syntéza při správné organizaci explozivní charakter.

Kontrolní otázky:

1. Co je umělý život.
2. Jaké jsou projevy umělého života.
3. Jak lze umělý život modelovat.

Korespondenční úkol:

1. Nalezněte na Internetu simulátory umělého života a vyzkoušejte si je na vlastní aplikaci.
2. Nalezněte např. na Internetu možné aplikace umělého života. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran

Shrnutí obsahu kapitoly

Stejně tak jako definice umělé inteligence naráží na problém neexistence všeobecně přijatelné definice inteligence přirozené, naráží i definice umělého života na nejednotnost definice života existujícího na Zemi. V této kapitole jsme se pokusili ozřejmit, co považujeme za umělý život a jaké jsou jeho projevy.

Pojmy k zapamatování

- umělý život
- celulární automat
- life
- Wolframův celulární automat

6 Deterministický chaos

V této kapitole se dozvíte:

- Co se rozumí pod pojmem „deterministický chaos“.
- Jaké jsou modely deterministického chaosu.
- Zda lze deterministický chaos řídit.

Po jejím prostudování byste měli být schopni:

- Vysvětlit co se rozumí pod pojmem „deterministický chaos“.
- Vysvětlit, zda lze deterministický chaos řídit a jaké jsou jeho modely.

Klíčová slova této kapitoly:

Deterministický chaos, atraktor, bifurkační diagram, samoorganizace.

Průvodce studiem

Když se vysloví slovo „chaos“, obvykle si vybavíme proces, který je čistě nahodilý a bez vnitřních zákonitostí. Málokdo si uvědomuje že „být chaotický“, znamená striktně se řídit přesně danými pravidly, v nichž mnohdy není pro náhodu místo. Chaos je disciplínou, která dostala své jméno až ve dvacátém století, kdy bylo zjištěno, že už i jednoduché problémy generují velmi složité a nepředpověditelné chování. V poslední době vznikl v souvislosti s deterministickým chaosem směr zvaný „řízení deterministického chaosu“. Je to relativně nový vědecký interdisciplinární směr, v němž dochází k symbióze více oborů jako je např. fyzika, chemie, biologie či elektronika. Vzhledem k šíři dané problematiky je kapitola zaměřena spíše do úrovně informativní s cílem podat studentovi alespoň nástin základních principů chaosu a jeho řízení.

6.1 Co je to deterministický chaos

Deterministický chaos představuje proces *samoorganizace* chování složitých systémů v souladu s přírodními zákony. Z hlediska jedné určité struktury obsahuje sice vývoj systému jisté prvky náhodnosti, avšak systém jako celek se vyvíjí zcela zákonitě a tedy deterministicky. Typickým příkladem relativně dobře prozkoumaného systému chovajícího se podle zákonů deterministického chaosu je *turbulentní proudění*, jehož struktura je charakterizována vírovými koherentními strukturami. Jejich velikost je sice dána zákonitostmi, ale okamžitá poloha a orientace konkrétního víru v prostoru je náhodná. Matematický model proudící tekutiny ukazuje, že za určitých podmínek může dojít k extrémnímu zesilování poruch určitého charakteru v proudovém poli. Systém tedy funguje jako filtr, který některé poruchy potlačuje a jiné zesiluje. Toto je obecná vlastnost dynamických systémů popsaných nelineárním matematickým modelem. Chaotické chování je tedy charakterizováno stavem, *kdy velmi malé, prakticky neměřitelné podněty vyvolávají velké změny v chování systému*. Pokud nejsme schopni tyto podněty indikovat, pak je chování systému bez zjevné příčiny, tj. chaotické. Při chaotickém chování systému roste řádově jeho komplexita (počet stupňů volnosti). Teorie chaosu paradoxně vede k tomu, že chaotické jevy v sobě ukrývají určité prvky řádu. Ten je obsažen především v matematických funkcích, které používáme k modelování a popisu. Matematická úroveň popisu těchto jevů pracuje např. s pojmy *fraktál* a *atraktor*. Slovo atraktor je odvozeno z anglického slovesa "to attract" (přitahovat), jedná se tedy o popis systému, který směřuje do určitého stabilního stavu. Fraktály a atraktory jeví podivuhodnou pravidelnost, kdy stejný matematický popis můžeme použít i na zcela nepříbuzné jevy. Teorie chaosu obecně popisuje systémy, které jsou charakterizovány tzv. nelineárními diferenciálními rovnicemi. Proto se někdy označují také jako "nelineární systémy". Podstatné však je především zjištění, že v rámci chaosu existuje současně jakási podivná organizovanost, přičemž v jistém úhlu je

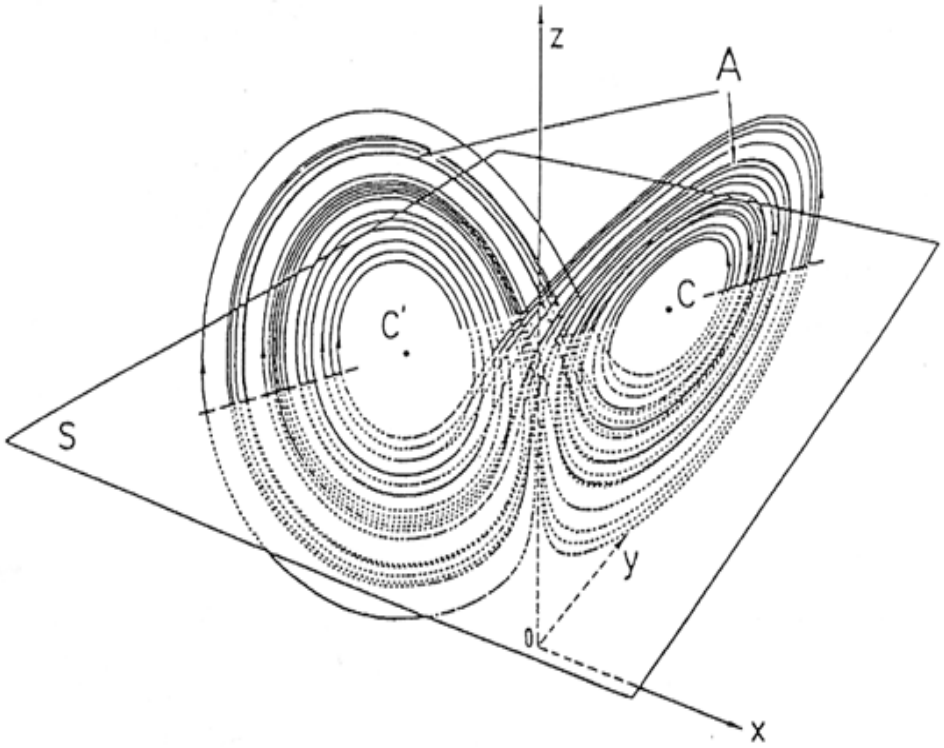
možné pohlížet i na živé organismy jako velmi složité nelineární systémy.

Za otce chaosu se považuje Edward Lorenz, jež se proslavil objevem tzv. "*motýlího efektu*", který prezentoval v 60. letech dvacátého století. Motýlí efekt poukazuje na nemožnost dlouhodobé předpovědi počasí (max. 2-3 dny) a tím zdůrazňuje jeho chaotičnost. Základní tezí motýlího efektu je „*zda nepatrné mávnutí motýlích křídel nad Tokiem může způsobit uragán nad New Yorkem“*. Lorenz vytvořil jednoduchý matematický model zemské atmosféry, na kterém se pokoušel studovat počasí, tzv. *Lorenzův systém*. Postupně svůj matematický model zjednodušil tak, že z původně dvanáctidimenzionálního modelu nakonec vznikl známý třírozměrný Lorenzův systém z roku 1963:

$$\frac{dx}{dt} = \sigma(y - x),$$
$$\frac{dy}{dt} = -xz + rx - y,$$
$$\frac{dz}{dt} = xy - bz.$$

Tento matematický model zachycuje základní vlastnosti konvektivního proudění atmosféry, která je zahřívána povrchem ze spodu a ochlazována svrchu. Vzniká tak rotační pohyb částic vzduchu, kdy ohřátá částice stoupá, tím se ochlazuje a začne klesat, aby se opět zahřála a stoupala. Proměnné *x*, *y* a *z* v uvedené rovnici nejsou souřadnicemi v prostoru, jejich fyzikální význam je poněkud abstraktní. Proměnná *x* představuje rychlost rotace pohybu částice, kladná hodnota je ve směru hodinových ručiček. Proměnná *y* reprezentuje rozdíl teplot stoupající a klesající tekutiny. Proměnná *z* charakterizuje odchylku svislého profilu teploty od lineárního průběhu. Parametry *r* je Rayleigho číslo, σ je Prandtlovo číslo a *b* představuje štíhlost válce tekutiny při konvekci, tedy poměr jeho délky a průměru. Lorenz prováděl matematickou simulaci systému, kdy pro hodnoty parametru $r \leq 1$ dospěje řešení k ustálené hodnotě $x = y = z = 0$. Obvyklé parametry pro atmosférické podmínky $\sigma = 10$, $r = 28$ a $b = 8/3$ způsobují

chaotické chování systému, kdy se směr rotace náhodně mění. Limitní stav - *atraktor*, který zde nastává po určitém přechodovém čase, zcela závisí na počátečních podmínkách. Tento stav může být zobrazen ve fázovém prostoru buďto jako bod - konečný stav klidu ke kterému systém spěje nebo jako limitní cyklus - uzavřená křivka, který odpovídá periodickému pohybu. Atraktor příslušející Lorenzovu systému za určitých podmínek vybočuje z tohoto konceptu, nedojde k jeho ustálení ani po velmi dlouhém čase a vzniká nekonvergující křivka, viz obr. 6.1.



Obrázek 6.1: Lorenzův podivný atraktor

Tento atraktor (obr. 6.1) je jedním z tzv. "*podivných atraktorů*" charakterizujících chaotické chování dynamického systému, který byl podroben zevrubnému systematickému zkoumání. Tento atraktor má některé podivné vlastnosti (Zelinka 2006):

- Je tvořen spojitou křivkou v prostoru, která obecně začíná v jistém počátečním bodě, může však mít nekonečně velkou délku. Přitom vyplňuje jistý přesně vymezený podprostor ve fázovém prostoru, ze kterého nikdy nevybíhá.
- Nikdy neprotíná sám sebe, nekříží se ani se neopakuje.

- Má vlastnost fraktálů, tj. jeho struktura se opakuje v různých měřítkách.
- Její průběh v prostoru je náhodný, chaotický, nepředpověditelný.

Kvalitativně podobným způsobem se chovají i jiné nelineární systémy, např. turbulentní proudění vazké tekutiny. Podmínky proudící tekutiny lze z hlediska stability řešení charakterizovat bezrozměrnou střední rychlostí proudění tzv. Reynoldsovým číslem. Pro Reynoldsova čísla vyšší než je kritická hodnota nastává chaotické chování proudící tekutiny a říkáme, že nastává přechod z laminárního stavu proudění do turbulentního. Výsledné chaotické chování a vývoj takového složitého systému je však podstatně komplexnější a složitější než u jednoduchého Lorenzova systému. Dochází zde ke vzniku chaosu na různých místech a v různých časových okamžicích. Výsledné koherentní struktury se potom bouřlivě vyvíjejí a interagují navzájem.

6.2 Nejjednodušší model deterministického chaosu

Jeden z velmi jednoduchých modelů chaotického chování je tzv. *logistická rovnice*

$$x_{n+1} = Ax_n\left(1 - x_n\right),$$

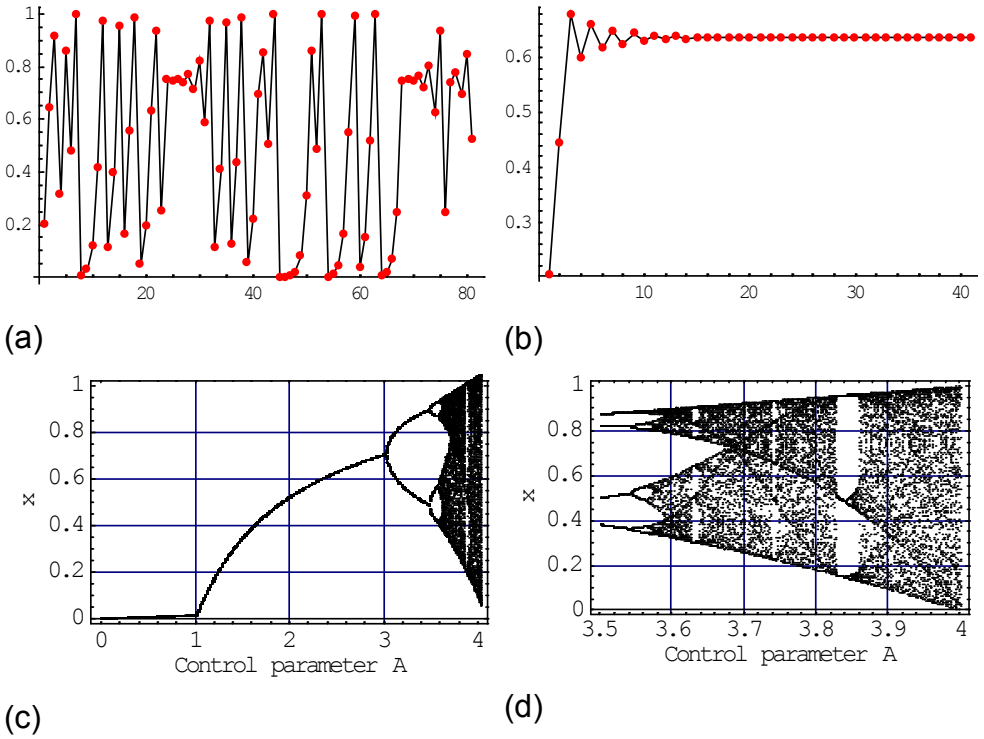
jež vznikla jako odezva na potřebu simulace biologických systémů, tj. popisuje chování druhu v jeho přirozeném prostředí. Tento popis byl založen na existenci nějakého druhu v "uzavřeném" prostředí (většina druhů má omezené migrační schopnosti), které mu poskytovalo obživu.

Nejjednodušším příkladem je např. chování kaprů a štik v rybníce. Pokud se do rybníku nasadí určité množství kaprů a štik, pak samozřejmě dojde k jejich množení a růstu. Ten bude mít nároky na kapacitu potravy (kaprů) v rybníku. Při rostoucím počtu jedinců bude klesat množství potravy v rybníce, což se projeví zpomalením růstu počtu kaprů. Od jisté hranice pro nedostatek potravy začnou štiky vymírat hlady, zatímco díky jejich poklesu se množství kaprů začne

zvyšovat. To od určité hranice množství kaprů způsobí jejich opětovný nárůst. Prostým citem tedy lze očekávat, že populace bude asi "periodicky" oscilovat, nebo se ustálí na nějaké hodnotě. Jak již jednoduché simulace ukázaly, může systém popsaný touto rovnicí vykazovat velmi komplikované chování od ustáleného přes periodické až po chaotické.

Toto chování lze znázornit pomocí tzv. *bifurkačního diagramu* (obr. 6.2), což je graf, který ukazuje závislost chování modelu systému na řídícím parametru. Bifurkační diagram (Zelinka 2006) lze "číst" tak, že plná křivka představuje ustálený stav daného systému na různých hodnotách řídícího parametru (obr. 6.2, A = 2.75). To je dobře vidět na obr. 6.2 (c) v intervalu <1, 3>, kde každý bod křivky představuje ustálený stav systému s různou hodnotou. Soubor těchto hodnot pak v závislosti na parametru A vytváří onu hladkou křivku. Při dalším nárůstu však dochází k zajímavému jevu, a to je tzv. *zdvojení periody* (x_{start} = 0.54278, A = 3.13), což znamená, že se stav populace vrací do původního stavu až po dalším stavu (0.5, 0.3, 0.5, 0.3,…). Při dalším nárůstu pak dochází k dalšímu zdvojení již zdvojené periody (x_{start} = 0.381, A = 3.51) a to se děje tak dlouho, až vývoj přejde do chaotického průběhu. Zajímavostí je, že daný průběh vykazuje fraktální charakter - daná zdvojení se „de facto“ v přesném měřítku opakují (a to i u jiných rovnic nežli je tato). Z obr. 6.2 (c) a (d) je rovněž vidět, že v bifurkačních diagramech jsou oblasti, které ukazují kdy se systém chová ustáleně <1, 3>, kdy dochází ke zdvojování periody <3, 3.5>, ale také kdy o ustáleném stavu či periodě nemůže být ani řeči. To je dobře vidět např. v místech, viz obr. 6.2 (d) cca A = 3.7, 3.8, či 3.9 a více. Toto „zrnění“, můžeme-li to tak nazvat, je deterministický chaos, který se v časovém rozvoji projeví jako křivka, která se nikdy přesně neopakuje. Z výše uvedeného bifurkačního diagramu plyne rovněž ještě další zajímavý fakt: Systém, který přešel do chaotického režimu v něm nemusí zůstat navždy a to i navzdory rostoucímu řídicímu parametru, díky jehož nárůstu se do něj dostal. Pokud tedy vykazuje daný systém

režim chaotického chování, pak je šance, že změnou řídicího parametru přejde systém opět do stabilního režimu.



Obrázek. 6.2*: Logistická rovnice; (a) chaotické chování rovnice při vhodné hodnotě řídicího parametru, (b) deterministické chování téže rovnice pro jiné nastavení , (c) bifurkační diagram, (d) detail.*

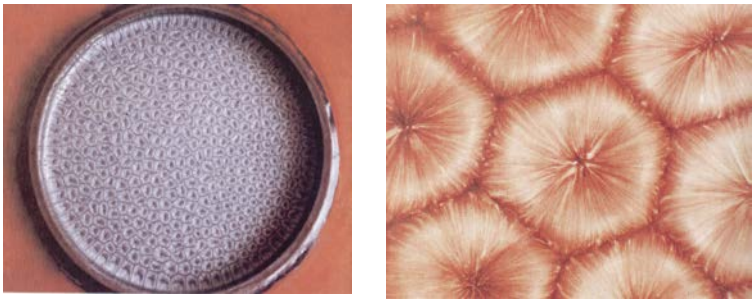
6.3 Řízení deterministického chaosu

Řízením deterministického chaosu se rozumí *takové působení na daný chaotický systém, aby se z režimu chaotického dostal zpět do režimu periodického či neperiodického ustáleného stavu*. Toto je základní myšlenka řízení deterministického chaosu (Zelinka, 2003). Existují samozřejmě i opačné trendy, tzn. takové řízení, jenž by daný systém přivedlo právě do chaotického režimu. Ačkoliv to zní absurdně, je faktem, že některé děje mají optimální průběh právě při chaotickém chování (např. chemické reakce). Možnost řídit deterministický chaos, dává rovněž možnost dopravit systém do jiného stavu s minimálními nároky na energii. Principem řízení deterministického chaosu je tedy ovlivňování aktuálního stavu systému, který vykazuje chaos, malými perturbacemi za účelem jeho stabilizace do periodického chování či

ustáleného stavu. Tato perturbace může ovlivnit chaotické chování jen tehdy, pokud je aktuální stav systému dostatečně „blízko“ režimu, do kterého má být systém převeden. Základní princip řízení chaosu snad nejlépe objasní následující příklad, jenž je založen na logistické rovnici (2). Z bifurkačního diagramu na obr. 6.2(c) je vidět, že při různých hodnotách řídicího parametru vykazuje rovnice různé typy chování deterministickým počínaje a chaotickým konče. Chaos nastává přibližně při hodnotě $A > 3.57$ a v případě že $A = 3.8$, je chování chaotické. Nechť je cílem řízení periodická trajektorie o periodě „*m*“, tzn. $x(m+1) = x(1)$ a zároveň existuje možnost jemně ladit parametr A v intervalu $\langle A - \delta,\ A + \delta \rangle$. Díky již zmíněným vlastnostem chaotického atraktoru tak dostaneme stav systému do blízkosti periodické trajektorie. Bez zásahu by samozřejmě stav systému pokračoval na svém chaotickém pohybu stavovým prostorem. Proto je nutno v okamžiku, kdy je systém ve vhodném stavu, provést řídicí zásah. V tomto případě změnit parametr A o ΔA, neboli perturbovat jeho hodnotu. Nutno podotknout, že míra perturbace se musí neustále přepočítávat. Aplikováním této perturbace se systém dostane do režimu *m*-periodické trajektorie a zůstane v ní dokud bude existovat toto řízení. Pokud bude ΔA nastaveno na 0, systém opět přejde do chaotického režimu. Nutno poznamenat, že čas potřebný k tomu, aby stav systému dospěl do dostatečné blízkosti je relativně dlouhý a souvisí s definováním okolí „ε“ okolo *m*-periodické trajektorie. Perturbace „δ“ je tedy aplikována, pokud je momentálně stav systému v okolí $u < \varepsilon$. Jinými slovy musí platit $x_n - x(i) << 1$ což znamená, že ΔA je rovněž velmi malé. Řízení deterministického chaosu daného systému je zaměřeno s ohledem na jeho časový vývoj. V současné době teorie řízení deterministického chaosu zahrnuje také řízení vícerozměrných systémů nebo systémů s dopravním zpožděním, což jsou v podstatě klasické oblasti řízení.

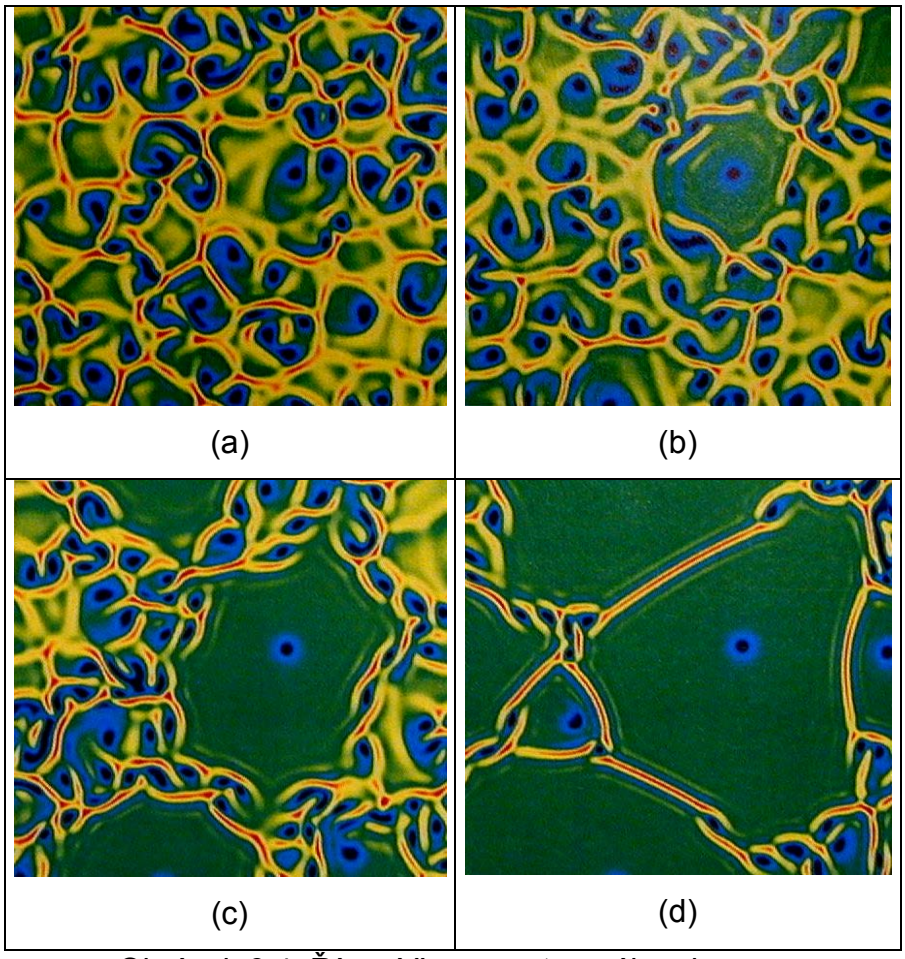
V teorii řízení deterministického chaosu lze objevit i studie, které se zaměřují na řízení časoprostorového chaosu. Chování systémů, které vykazují časoprostorový chaos už lze mnohdy definovat jako *samoorganizaci*, což je další speciální třída chování, při níž se

v chování systému objevuje nová kvalita (obr. 6.3), obvykle geometrické uspořádání, která nebyla do systému importována. Samoorganizace (Zelinka, 2003) je jev, který je intenzivně studován teprve od minulého století. Je velmi dobře propracován po teoretické stránce např. v chemii, kde se při vhodných podmínkách začnou milióny molekul chovat jako inteligentní jedinci (např. v Bělousov – Žabotinského reakci začnou cyklicky vytvářet okem viditelné barevné vzory v tekutině s geometrickým motivem). Samoorganizaci lze pochopit díky *entropii*, která vyjadřuje míru neuspořádanosti daného systému. Čím více je daný systém neuspořádaný, tím větší je entropie a naopak. Samoorganizace pak vzniká v systému, který se skládá z velikého množství samostatných jednotek (molekuly, lidé…), kteří tvoří menší systémy, mezi nimiž dochází k „tokům“ entropie. Jinými slovy, vzniká mezi částmi tvořícími celek, které se navzájem snaží ovlivnit si své entropie. Dochází tak ke vzniku tzv. *disipativních struktur* (tj. struktur, kde dochází k tokům energie „z“ a „do“ okolí systému), které mohou za určitých podmínek vykazovat chaotické chování (Zelinka 2003).



Obrázek 6.3: Samoorganizace – samovolně se tvořící šestihrany v zahřívané kapalině, celkový pohled (vlevo) a detail (vpravo).

Pro řízení časoprostorového chaosu byla vypracována metoda využívající řízení daného systému topologických defektů. Na obr. 6.4 (a)-(d) je postupně vidět jak samotný časoprostorový chaos, tak i jeho postupné ustálení (Zelinka 2003).



Obrázek 6.4: Řízení časoprostorového chaosu.

Kontrolní otázky:

1. Co se rozumí pod pojmem „deterministický chaos“.
2. Jaké jsou modely deterministického chaosu.
3. Lze deterministický chaos řídit.

Korespondenční úkol:

Nalezněte např. na Internetu další možné aplikace deterministického chaosu. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Tato kapitola představuje úvod do problematiky deterministického chaosu. Vzhledem k šíři dané problematiky je kapitola zaměřena spíše

do úrovně informativní s cílem podat studentovi alespoň nástin nejjednoduššího modelu deterministického chaosu a možnosti jeho řízení.

Pojmy k zapamatování

- deterministický chaos
- atraktor
- bifurkační diagram
- samoorganizace

7 Fraktály

V této kapitole se dozvíte:

- Co je to fraktál.
- Jak lze fraktály generovat.
- Co je to fraktální (Hausdorffova) dimenze.
- Kde lze fraktály nalézt.

Po jejím prostudování byste měli být schopni:

- Charakterizovat fraktál.
- Popsat, jak lze fraktály generovat.
- Popsat rozdíl mezi topologickou a fraktální (Hausdorffovou) dimenzí.
- Uvést, kde lze fraktály nalézt.

Klíčová slova této kapitoly:

Fraktál, soběpodobnost, soběpříbuznost, systémy iterovaných funkcí IFS, L-systémy, topologická dimenze, fraktální (Hausdorffova) dimenze, power law.

Průvodce studiem

Tato kapitola představuje úvod do problematiky fraktální geometrie. Nejprve se seznámíte s pojmem fraktál a se základním dělením fraktálů. Dále si uvedeme, jak lze fraktály generovat, co je to fraktální dimenze a kde se můžeme s fraktály v přírodě setkat.

7.1 Co je to fraktál

Protože velká část fraktálů je využívána v počítačové grafice a fraktály lze nejlépe popsat jako geometrické objekty. Můžeme na ně nahlížet

jako na jako nekonečně členitý útvar. Opakem nekonečně členitého útvaru je geometricky hladký útvar, který lze popsat klasickou Euklidovskou geometrií.
Slovo *fraktál* pochází z latinského slova „fractus“ a znamená „rozlámaný“. Jednoznačná definice fraktálu sice neexistuje, ale lze jej charakterizovat následovně.

Fraktál je geometrický objekt, který po rozdělení na menší části vykazuje tvarovou podobnost s těmito částmi (*Zelinka, 2006*).

Fraktál je množina či geometrický útvar, jejíž Hausdorffova dimenze je (ostře) větší než dimenze topologická. (*Mandelbrot, 1982*)

Fraktálními objekty se zabývá samostatná vědní disciplína nazývaná *fraktální geometrie*. Fraktální geometrie je poměrně mladé odvětví matematiky, jehož oficiální vznik lze datovat do 70. let minulého století. Za jejího zakladatele je považován matematik Benoit B. Mandelbrot, který jako první vymezil pojem fraktál. I před zavedením pojmů fraktál a fraktální geometrie se vědci i umělci zabývali geometrickými útvary, které dnes nazýváme fraktály, jako jsou například sněhová vločka Helge von Kocha nebo Sierpinského trojúhelník apod.

Charakteristickými vlastnostmi fraktálů jsou *soběpodobnost* nebo *soběpříbuznost*. Podle této vlastnosti rozlišujeme dvě základní skupiny fraktálů.

Soběpodobné fraktály

Fraktály jsou soběpodobné, pokud je kterákoliv jejich část přesnou kopií původního objektu bez ohledu na změnu vůči měřítku. Vyskytuje se jen u čistě matematických struktur, protože jsme v přírodě jednak omezeni velikostí částic (některé se zdají být nedělitelné) a dále těžko v přírodě vznikne takto dokonalý fraktál.

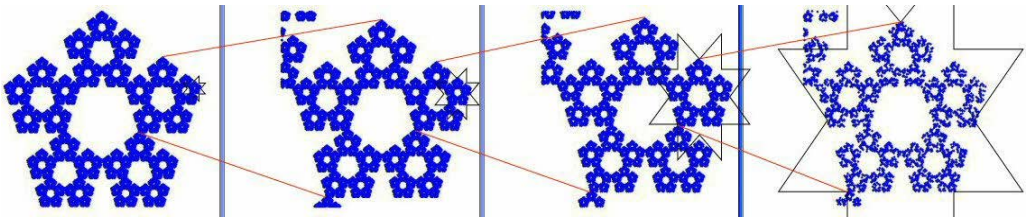
Soběpodobnost je jednou z hlavních vlastností fraktálu. Dá se říci, že soběpodobnost znamená opakování sebe sama, např. se změnou měřítka, posunutím, rotací nebo zkosením. V matematice se setkáváme s označením invariance vůči změně měřítka. Tato vlastnost také napomáhá identifikaci fraktálů v přírodě. Například kapradinu můžeme považovat za složeninu nekonečného počtu větviček, nebo lístků, které po zvětšení vypadají opět jako kapradina.

Soběpříbuzné fraktály

Do skupiny soběpříbuzných fraktálů patří takové fraktály, u nichž je libovolný výsek objektu podobný objektu původnímu, není však zcela shodný s původním motivem. Soběpříbuznost lze nalézt v přírodě téměř na každém kroku, je základní vlastností všech přírodních struktur a systémů, vesmírem počínajíc, strukturou listu a tvarem mraků či profilem krajiny pokračujíc a větvením žil v lidském těle konče. Členitost při zvětšování měřítka zůstává stále podobná struktuře z menších zvětšení. Tato takzvaná soběpodobnost (obr. 7.1), nikoliv stejnost! fraktálních útvarů je jejich hlavním znakem a většinou je také považována za jejich definici. Tuto přítomnost podobných útvarů při zvětšování detailů nenajdeme i na klasických, elementárních útvarech, jako jsou krychle, koule, čtverce apod.

Pokud nějakou kouli zvětšíme, pak v ní sice neuvidíme menší kulové struktury, ale lze říci, že pomocí těchto elementárních tvarů (jako jsou úsečky, trojúhelníky, čtverce, koule apod.) jejich nekonečným zmenšováním a vnořováním můžeme fraktální útvary "uměle vyrobit" a napodobit tak přírodu, která je už vlastně dávno zná. Princip opakování podobných tvarů ve zmenšené podobě je vidět nejen u neživých útvarů, ale i u živých organizmů a jejich skupin, prakticky u jakékoliv komplexní, složité struktury, která je vytvářena i pomocí velmi jednoduchých pravidel. Způsob, jakým probíhá větvení stromů (silnější větve se rozbíhají ve stále menší a tenčí větvičky) či cév a žil v tělech živočichů nebo hromadění např. baktérií a řas v koloniích a také tučňáků na ostrovech v polárních mořích se dá popsat jediným nástrojem: *fraktální*

geometrií. Ta je však schopna vysvětlit ještě více: slouží nejen k pochopení morfologie složitých statických prostorových struktur, ale je schopna pojmout i složité děje, které se odehrávají v čase, tedy jevy dynamické, neboť princip koexistence jednoduchých výchozích pravidel a komplexního, složitého výsledného tvaru platí i zde. Lze jej chápat i jako princip koexistence řádu a chaosu v jediném objektu. (Zelinka 2003)

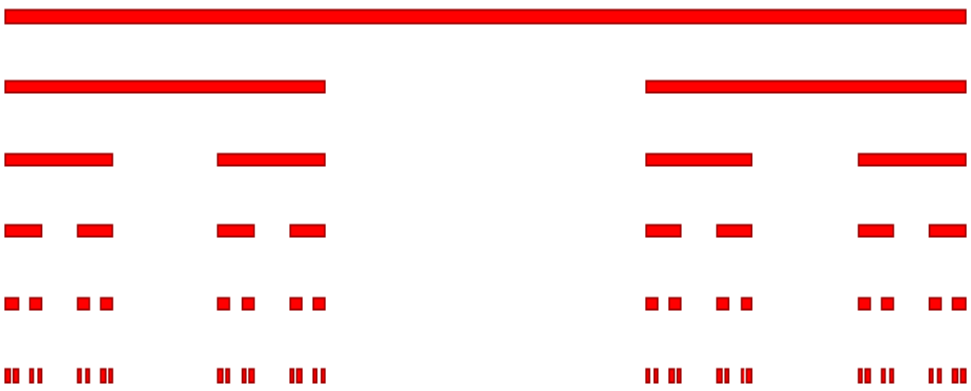


Obrázek 7.1: Ukázka soběpodobnosti krystalů

Klasické fraktály

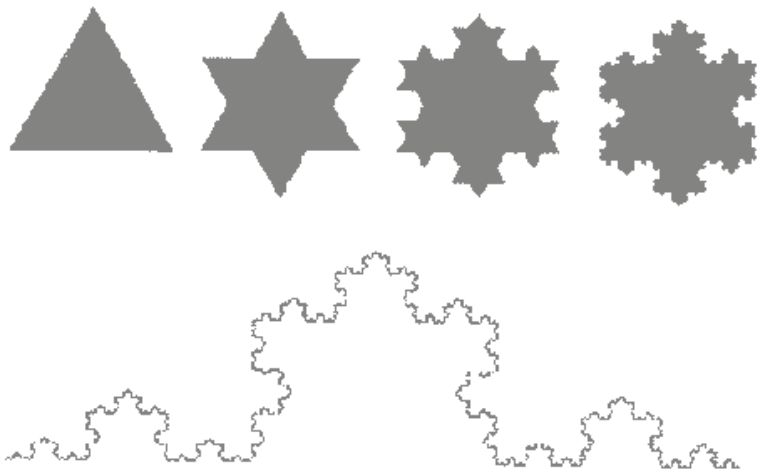
Uveďme některé z těchto objektů.

Cantorova množina je množina bodů z uzavřeného intervalu <0; 1>. Nejjednodušší definicí této množiny je popis, jak ji získat. Interval <0; 1> rozdělíme na tři shodné a otevřený interval (1/3; 2 /3) vyjmeme. Čísla 1/3 a 2/3 nám tedy zůstanou v množině. Takto jsme získali dva uzavřené intervaly <0; 1/3> a <2/3; 1> o délce 1/ 3 . Nyní opakujeme tento postup na nové intervaly, tj. z obou intervalů vyjmeme prostředek. To provádíme až do nekonečna. Body, které zůstanou, nám definují Cantorovu množinu (obr. 7.2).



Obrázek 7.2: Konstrukce Cantorovy množiny

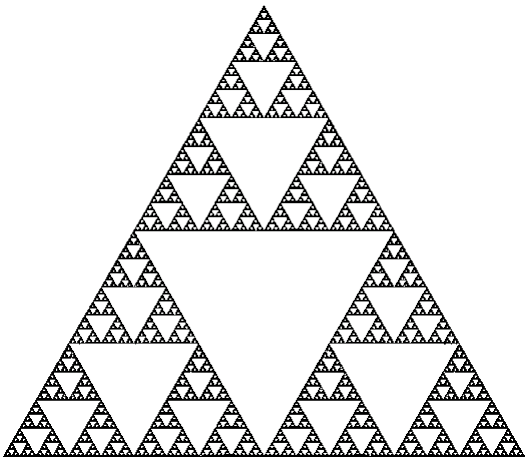
Křivka Kochové, někdy je v upraveném tvaru nazývána též Kochova vločka, nebo Kochův ostrov. Tato křivka má zajímavé matematické a geometrické vlastnosti: i když je v celém svém rozsahu spojitá, v žádném bodě nemá konečnou derivaci. Každý bod na křivce je totiž po nekonečně mnoha iteracích průnikem dvou „nekonečně malých“ úseček, které tvoří strany trojúhelníka, který je taktéž „nekonečně malý“. Každá iterace křivku o malý kousek prodlouží, ale plocha zůstává na rozdíl od křivky konečná, je proto nekonečně dlouhá, i když zabírá jen omezenou část plochy, jak je ostatně patrné z obrázku 7.3.



Obrázek 7.3: Křivka Kochové

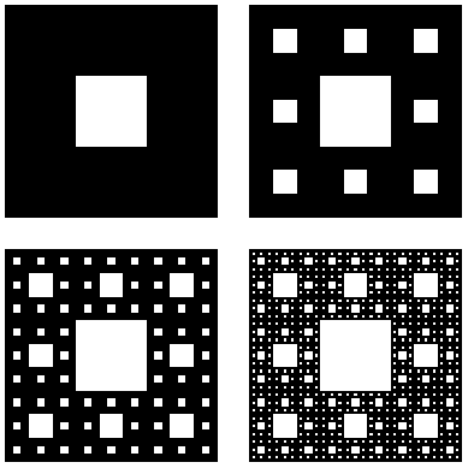
Sierpinského trojúhelník (obr.7 .4). Zkonstruujeme jej aplikací následujících tří jednoduchých kroků:

1. Narýsujeme libovolný trojúhelník.
2. Vyznačíme jeho střední příčky.
3. Vyjmeme prostřední trojúhelník a na zbylé tři aplikujeme rekurzivně kroky 2 a 3.



Obrázek 7.4: Sierpinskéhoho trojúhelník

Sierpinského koberec je uveden na obrázku 7.5. Sierpinského koberec je jistou modifikací Sierpinského trojúhelníku. Postup jeho konstrukce je proto podobný konstrukci Sierpinského trojúhelníku, avšak místo trojúhelníku zde použijeme čtverec, a to následovně. Vezmeme jednotkový čtverec, ten pak rozdělíme na 9 čtverců o straně 1/3 a prostřední vyjmeme, což rekurzivně opakujeme.



Obrázek 7.5: Sierpinského koberec

7.2 Generování fraktálů

Podobně jako v ostatních vědních disciplínách, i ve fraktální geometrii se po určitém čase začaly rozlišovat jednotlivé typy fraktálů, kdy fraktály stejného typu mají shodné své nejvýznamnější charakteristiky. Toto rozlišení není zavedeno jen kvůli systematičnosti, ale i proto, že

jednotlivé typy fraktálů jsou vhodné pro řešení určitého okruhu problémů, které se vyskytují v různých oborech. Také způsob jejich vytváření (generování) se liší podle jejich typu, přičemž fraktály stejného typu se většinou generují s využitím podobných algoritmů.

Základní typy generování fraktálů jsou následující (*Zelinka, 2006*):

- L-systémy
- Systémy iterovaných funkcí IFS
- Dynamické systémy
- Nepravidelné fraktály

L-systémy (Lindenmayerovy systémy) jsou skupinou fraktálů definovaných pomocí přepisovacích gramatik. Podstatou tvorby L-systémů je přepisování řetězců podle určitých pravidel. Každý symbol v řetězci má přiřazen jistý geometrický význam, například transformaci či generování objektu. S pomocí L-systémů lze generovat fraktály, které se podobají rostlinám, stromům a dalším přírodním útvarům. Některé aplikace směřují k využití těchto fraktálů při generování textur. Těmto fraktálům se také někdy říká graftály.

Systémy IFS jsou to generativní metody pro tvorbu fraktálů. Tyto metody jsou vhodné jak pro generování fraktálů, tak i pro kompresi bitmapových obrazů, zavedeme-li tzv. inverzní úlohu. Zajímavé je, že tato metoda může být jak *deterministická*, tak i *náhodná*. Oboje však paradoxně vede ke stejnému výslednému fraktálu, pokud použijeme dostatečný počet iterací.

Dynamické systémy jsou pravděpodobně tím typem generování fraktálů, který má v technické praxi nejširší uplatnění. Dynamický systém je sytém, který je ve většině případů závislý na čase. Dynamický systém vychází z počátečních podmínek a je jimi v čase determinován. Existují i dynamické systémy, které se po určitém čase neustálí v pevném stavu, ale ani nedivergují. Tento případ má většinou fraktální dynamiku a označuje se termínem deterministický chaos. Za

dynamický systém v komplexní rovině lze považovat *Mandelbrotovu množinu*. (Mandelbrot 2003)

Nepravidelné fraktály - další skupinou fraktálů jsou nepravidelné fraktály. Zatímco všechny předchozí skupiny fraktálů byly v určitém smyslu symetrické, nepravidelné fraktály vnáší při generování do algoritmu náhodu. Tento typ fraktálů také umožňuje nejvěrohodnější popis přírodních objektů.

7.3 Fraktální dimenze

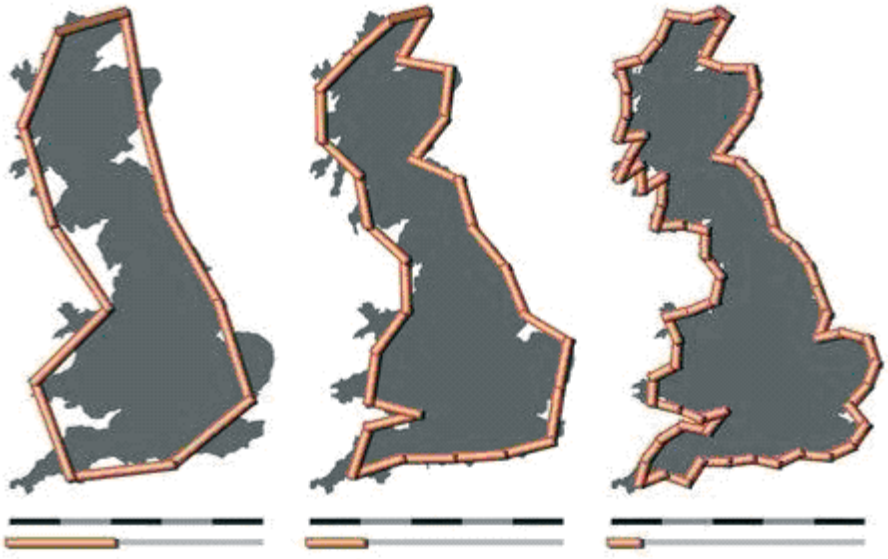
Ústřední pojem při zkoumání fraktálů hraje jejich dimenze, a to jak dimenze topologická, tak i dimenze fraktální (někdy také nazývaná dimenze Hausdorffova).

Topologická dimenze

Geometricky hladké objekty, které je možné popsat klasickou Euklidovskou geometrií, mají celočíselnou dimenzi, nazývanou také *topologická dimenze*. Pokud si celou problematiku zjednodušíme, můžeme říci, že topologická dimenze určuje počet parametrů (nezávislých proměnných), jakým lze dané těleso (resp. každý bod na tělese) popsat. Například bod má nulovou dimenzi, jelikož je jeho poloha popsána vztahem $P=X$ (tj. konstantním vektorem) a úsečka má dimenzi rovnu jedné, neboť ji lze popsat vztahem $y_t=y_0+kt$, kde t je jediný parametr (nezávislá proměnná). Jakákoliv hladká plocha (kruh, trojúhelník, n-úhelník) má dimenzi rovnu dvěma, to znamená, že poloha bodu musí být určena pomocí dvou parametrů. Krychle, koule, válec nebo celý běžný prostor kolem nás mají dimenzi rovnu třem, protože poloha jakéhokoli bodu je v nich jednoznačně určena třemi parametry. Tímto způsobem je samozřejmě možné pokračovat dále, ale s dalšími dimenzemi nemáme osobní zkušenosti - v reálném světě za obvyklých podmínek prakticky neexistují. Ve všech případech jsme hovořili o dimenzi, která je specifikována celým číslem. Tato dimenze se nazývá *dimenze topologická*.

Power law, univerzální zákon přírody

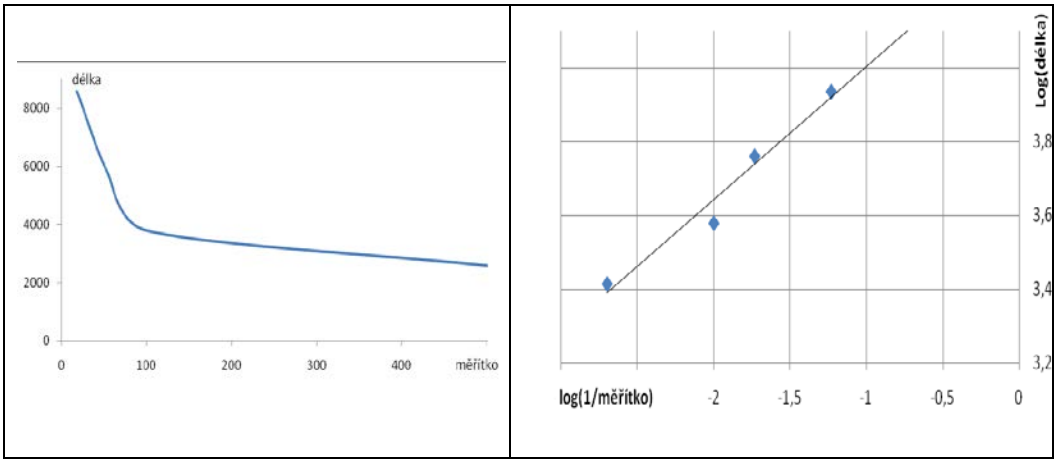
Mocninné zákony zejména dobře charakterizují fraktály. Není to náhoda, neboť pravidelné fraktály se vyznačují soběpodobností. U náhodných fraktálů, které jsou v přírodě mnohem častější, není soběpodobnost přesně geometrická, ale musí se chápat v pravděpodobnostním smyslu: změníme-li měřítko zobrazení, objekt zůstává v pravděpodobnostním smyslu totožný. To se vyznačuje pojmem *škálová invariance*, a právě tato absence charakteristické délky, je matematicky dobře vyjádřena mocninným rozdělením. Rozdělovací funkce *P(s)* udávající pravděpodobnost, že ve fraktálu nalezneme objekt velikosti *s*, má tvar mocniny $P(s) = s^{-\tau}$. Změna škály znamená vynásobení všech velikostí konstantou $s \rightarrow bs$. Přitom se tvar rozdělovací funkce nemění $P(s) = s^{-\tau} \rightarrow (bs)^{-\tau} = b^{-\tau}s^{-\tau} \approx s^{-\tau}$, jelikož konstanta $b^{-\tau}$ je pohlcena požadavkem na normalizaci, $\int P(s)ds = 1$.
Matematická podstata mocninného zákona (*power law*) je nejlépe vidět, znázorní-li se graficky (obr. 7.6).



Obrázek 7.6: Měření délky ostrova

Ve zjednodušené formě představuje přímku. Čím je přímka strmější, tím je „power“, z anglického „exponent“, větší. Na obr. 7.7(a) je vyjádřena závislost naměřené délky pobřeží na měřítku (Kukal, 2009). Na obr.

7.7(b) je tato závislost znázorněna v logaritmických souřadnicích, což není náhoda, neboť tak lze mocninný zákon znázornit jako přímku, jejíž sklon je zároveň jejím exponentem. Jednotlivý mocninný zákon je pak reprezentován skupinou bodů více či méně se přimykajících k této úsečce.



(a) (b)

Obrázek 7.7: (a) Závislost naměřené délky pobřeží na měřítku
(b) Mocninný zákon pro délku pobřeží v závislosti na měřítku
v logaritmických souřadnicích.

Z obrázku 7.7(b) je zřejmé, že se hodnoty na ose „x“ pohybují v rozmezí -2,7 až -1,2 a máme-li například zjistit, jaká bude naměřená délka pobřeží při hodnotě měřítka -1, pak je řešení tohoto problému jednoduché. Vzhledem k tomu, že závislost je lineární, lze přímku prodloužit a provést odečet závisle proměnné – v tomto případě délky pobřeží. Tím se vlastně získá odhad – *predikce*, jak velkou délku při daném měřítku naměříme.

Fraktální dimenze

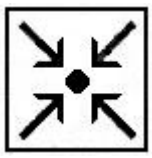
Pro fraktální objekty je číselná hodnota této dimenze větší než hodnota dimenze topologické. Ostatní (ne-fraktální) objekty mají tu vlastnost, že zmenšováním délky měřidla se přibližuje délka objektu (obvod) k nějaké limitní hodnotě, tj. pokud budeme měřit délku geometricky hladké křivky a budeme-li při tom měnit měřítko, dostaneme vždy stejný výsledek. Fraktální dimenze umožňuje popsat stupeň složitosti objektu podle

toho, jak rychle roste jeho délka, obsah či objem v závislosti na velikosti měřítka, při kterém měříme. Proto můžeme mocninný zákon zobecnit i na objekty neceločíselné dimenze, kdy změna měřítka u fraktálního objektu vede k růstu jeho zobecněného objemu (v prostoru s nějakou obecnou i ne-Euklidovskou metrikou), který je vyjádřen $N(r) = r^D$, kde r je měřítko, $N(r)$ můžeme chápat jako počet zobecněných jednotkových objemů, které jsou potřeba k jeho pokrytí a D je fraktální dimenze. Fraktální objekty si nemusí být na různých škálách zcela podobné a jejich části mohou být organizovány náhodně, musí však splňovat podmínku, že D je stejné pro všechna měřítka. V reálných systémech samozřejmě existují omezení. Čím více se hodnota Hausdorffovy dimenze liší od topologické, tím je daný objekt více členitý a naopak. Mezním příkladem je hodnota Hausdorffovy dimenze, která je o jedničku větší než dimenze topologická, což je vlastnost hranice Mandelbrotovy množiny.

Na jednoduchém příkladu si nyní ukážeme rozdíl mezi Hausdorffovou dimenzí pro úsečku a známou křivku Helge von Kocha, což jsou oba objekty s topologickou dimenzí 1. Nejprve vytvoříme úsečku, která má jednotkovou délku, pak tuto úsečku rozdělíme na N dílů. Délka jednoho dílku bude $s=1/N$. Kdybychom místo úsečky dělili čtverec na N dílků, délka jednoho dílku by byla $s=1/N^{1/2}$. Podobně u krychle by byla délka jednoho dílku $s=1/N^{1/3}$. Obecně můžeme zapsat $s=1/N^{1/D}$, kde D je fraktální (Hausdorffova) dimenze objektu (*Zelinka, 2006*).

Příklad: fraktální dimenze úsečky

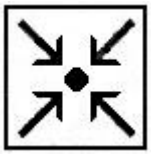
U geometricky pravidelných objektů je výpočet fraktální dimenze jasný. V případě úsečky dostaneme po dosazení za $s=1/N$ hodnotu fraktální dimenze $D = \frac{\log N}{\log\left(\frac{1}{s}\right)} = \frac{\log N}{\log N} = 1$.



Topologickou dimenzi úsečky samozřejmě známe z Euklidovské geometrie: je rovna jedné. Hausdorffovu dimenzi jsme nyní vypočítali a

je také rovna jedné. Hausdorffova dimenze se tedy rovná dimenzi topologické. Z definice fraktálu je zřejmé, že úsečka není fraktál, protože pro fraktál musí být Hausdorffova dimenze ostře větší než dimenze topologická.

Příklad: fraktální dimenze křivky Kochové



Postup výpočtu fraktální dimenze nyní aplikujme na nejjednodušší fraktál na ploše - křivku Kochové, pro niž platí, že se každá úsečka předchozího útvaru nahradí dvěma úsečkami se třetinovou délkou a rovnostranným trojúhelníkem sestrojeným uprostřed mezi nimi. Při každé iteraci se tak délka úsečky *s* zmenší na 1/3 své původní hodnoty a počet soběpodobných úseků vzroste na *N* = 4. Při trojnásobném zjemnění se délka křivky zvětší čtyřikrát, proto Hausdorffova dimenze není celé číslo. Pro *N*=4 se měřítko musí zmenšit na třetinu *s*=1/3. Hausdorffova dimenze křivky Helge von Kocha se pak vypočítá následovně:

$$D = \frac{\log N}{\log\left(\frac{1}{s}\right)} = \frac{\log 4}{\log 3} = 1{,}261895.$$

Topologická dimenze této křivky je rovna jedné, Hausdorffova dimenze je však větší než jedna. Z toho vyplývá, že křivka Helge von Kocha je fraktálem.

7.4 Výskyt fraktálů

Přírodní fraktály oproti matematickým nejsou nikdy striktně soběpodobné, proto přírodní útvary nejsou při zvětšení přesně identické. S fraktály se můžeme setkat ve fyzice, chemii, biologii a mnoha dalších vědních disciplínách, které nemají se studiem geometrických objektů zdánlivě nic společného. Dnes již klasickým případem jsou pobřeží ostrovů a břehy říčních toků, které je možné modelovat pomocí stochastických fraktálů. Dalšími příklady přírodních fraktálů jsou pohoří, oblaka, blesky, sněhové vločky, stromy i listí, cévní

systémy živočichů, komůrky v plicích, DNA, buněčné kolonie, dále rozložení hmoty v galaxiích, hvězdokupy, ale i povrch, tvar trhlin a vad v obrobku či nástroji, časový vývoj akcií na burzách, časové změny inflace, zadlužení států i vývoj kurzu měny. Dá se říci, že nás fraktály obklopují a jsou všude, aniž bychom to tušili.

U přírodních fraktálů neznáme „algoritmus generování“ útvaru. Vlastnosti těchto fraktálů můžeme jen odhadovat. Přírodní fraktály jsou často multifraktály. Tento pojem značí, že má fraktál v určitých měřítcích odlišnou složitost. To je dáno různou váhou vlivů působících na vznik fraktálu v různých měřítcích

Kontrolní otázky:

1. Co je to fraktál?
2. Jak lze fraktály generovat?
3. Co je to fraktální (Hausdorffova) dimenze?
4. Kde lze fraktály nalézt?

Korespondenční úkol:

1. Nalezněte např. na Internetu možné využití fraktálů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.
2. Nalezněte např. na Internetu další možné výskyty fraktálů, a to nejen v přírodě. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Tato kapitola představuje úvod do problematiky fraktální geometrie. Nejprve jste se seznámili s pojmem fraktál, se základním dělením fraktálů a jak lze fraktály generovat. Dále jsme si vysvětlili rozdíl mezi topologickou a fraktální (Hausdorffovou) dimenzí. Závěr kapitoly se věnuje výskytům fraktálů.

Pojmy k zapamatování

- fraktál,
- soběpodobnost,
- soběpříbuznost,
- systémy iterovaných funkcí IFS,
- L-systémy,
- topologická dimenze,
- fraktální (Hausdorffova) dimenze,
- power law.

8 Umělé imunitní systémy

V této kapitole se dozvíte:

- Co se rozumí pod pojmem „umělý imunitní systém“.
- Jak lze umělé imunitní systémy modelovat.
- Jaké jsou aplikace umělých imunitních systémů.

Po jejím prostudování byste měli být schopni:

- Charakterizovat, co je to umělý imunitní systém.
- Vysvětlit, jak lze umělé imunitní systémy modelovat.
- Uvést, jaké jsou aplikace umělých imunitních systémů.

Klíčová slova této kapitoly:

Biologický imunitní systém, umělý imunitní systém, antigen, afinita, protilátka.

Průvodce studiem

Tato kapitola představuje především úvod do problematiky umělých imunitních systémů. Imunitní systém vykazuje velmi zajímavé vlastnosti, nejen z lékařského hlediska, např. přizpůsobivost, robustnost a schopnost učení se. Seznámíme se zde především s využitím vlastností imunitních systémů v technicky zaměřených oblastech.

8.1 Biologický imunitní systém

Pojmem *biologická imunita* zde bude označován *imunitní systém* člověka. Právě tyto systémy jsou objektem výzkumu *imunologie*, které jsou základem umělých imunitních systémů.

Imunitní systém lidského těla funguje na základě vzájemného působení velkého množství buněk a molekul pokrývající celý organizmus člověka. Tento systém vytváří ochranný mechanizmus. Z pohledu funkčnosti systému je hlavním výkonným prvkem právě buňka, která komunikuje s jinými buňkami, množí se, aktivuje se, eliminuje infikované buňky a umírá. Imunitní systém patří k nejdůležitějším a nejsložitějším systémům v lidském organismu. Úkolem imunitního systému je zabezpečit obranu organismu před různými cizorodými látkami, tj. baktériemi, viry apod. Imunitní systém je schopen *detekovat* a *ničit* patogeny (tj. živé původce nemocí). Důležitou vlastností imunitního systému je *paměť*. Díky ní je schopen se rychleji a efektivněji vypořádat s cizorodou látkou, se kterou se již setkal. Tento jev vnímáme v běžném životě jako odolnost na již překonané nemoci. Schopnost *učení se*, je základem adaptability (schopnost přizpůsobení), která zabezpečuje dlouhodobou efektivitu systému.

Imunitní odpověď systému vzniká po objevení cizorodé látky v organismu. Takovou látku pak označujeme jako *antigen*. Antigeny vyvolávající imunitní odpověď musí mít následující vlastnosti:

1. *Cizorodost*

   V průběhu vývoje imunitních buněk se lymfocyty učí poznávat vlastní struktury od nevlastních. Vlastní antigeny tolerují, zatímco na cizí si ponechávají potenciální schopnost reagovat, tj. indukovat imunitní reakci. Všeobecně platí: čím je větší rozdíl v genetické výbavě mezi jedincem, který na antigen odpovídá a jedincem z kterého antigen pochází, tím je i reakce rychlejší a silnější. Např. transplantovanou ledvinu jednovaječného dvojčete přijme organismus lehce, zatímco ledvina od nepříbuzného dárce, pokud se nepoužijí látky tlumící imunitní reakci, může být odvrhnutá.

2. *Degradovatelnost*

   Imunitní buňky rozpoznávají buď určitý fragment pocházející z molekuly antigenu, nebo rozpustný antigen. Pokud molekula nemůže být rozložena nebo rozpuštěna, nemůže představovat

antigen. To je i důvodem, proč organizmus přijímá např. umělé klouby bez problémů.

3. *Biochemická struktura*

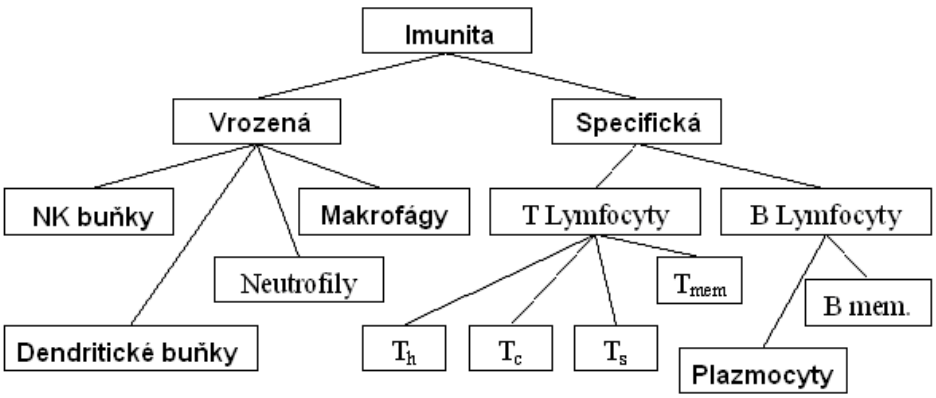
   Bílkoviny jsou výborné antigeny pro svou velikost a komplexnost na rozdíl od sacharidů, lipidů nebo nukleových kyselin, jež představují tzv. slabé antigeny. Slabé antigeny však mohou vytvářet společné vazby.

4. *Molekulová hmotnost*

   Všeobecně jsou velké molekuly lepší antigeny něž menší molekuly. Pod jistou hranicí molekulové hmotnosti dokonce imunitní systém na molekuly vůbec nereaguje. Velikost molekuly není úplně postačující, důležité je, aby molekula byla i dostatečně smíšená a komplexní.

5. *Dávka a cesta vniknutí antigenu do organizmu*

   Po vniknutí antigenu do organizmu antigen indukuje imunitní reakci jen v případě, pokud se podá v určité optimální *dávce*. Nedostatečná dávka nevyvolá imunitní odpověď, protože nedokáže stimulovat dostatek lymfocytů. *Cesta* vniknutí antigenu do organizmu rozhoduje o tom, který lymfatický orgán a které populace buněk se aktivují. Např. antigen podaný nitrožilně se dostává do sleziny a vyvolá systémovou odpověď, antigen podaný podkožně se dostane do oblastních lymfatických uzlin a vyvolá včas lokální odpověď.



Obrázek 8.1: Rozdělení imunitních buněk.

Buňky imunitního systému (obr. 8.1)

Počet imunitních buněk je asi 10^{12} (Buc 1998). Tyto buňky kolují po celém těle v rámci krevního a lymfatického oběhu. Díky tomu mají přístup do každého místa v organizmu. Existuje několik typů buněk, které tvoří imunitu a každý typ buňky má svoje specifické funkce. Na základě různých enzymů a komplexů bílkovin probíhá velmi důležitá komunikace mezi buňkami. Podle schopnosti reakce na cizí prvky se imunitní buňky dělí na dvě skupiny: nespecifické buňky a specifické buňky. Obě tyto skupiny tvoří základ dvou linií imunitní obrany, tj. *vrozená* (nespecifická) imunita a *získaná* (specifická) imunita.

Vrozená imunita je založena na receptorech, schopných rozpoznat molekulární vzory spojené s mikrobiologickými patogeny, které se nikdy nevyskytují u vlastního organizmu. Představuje první linii obrany organizmu a na daný patogen se reakce objevují již během několika minut. Reagují však stereotypně, nemají schopnost rozpoznávat, učit se a cíleně likvidovat daný patogen.

Nespecifické imunitní buňky jsou přitahované k místu infekce chemickými látkami, které vypouštějí poškozené buňky. Tato prvotní imunitní reakce způsobí zvýšení tělesné teploty, která vede k zvýšenému průtoku krve a k intenzivnější činnosti bílých krvinek v postihnuté oblasti. Klíčovou činností pro rozpoznání a eliminaci škodlivých buněk je *fagocytace*, tj. *neutrofily* (velmi početný typ bílých krvinek, které jsou vysoce destrukční pro mikroorganizmy) a *makrofágy* (velké bílé krvinky mající schopnost rozpoznat a požírat mikroorganizmy) požírají cizí látky a mrtvé či poškozené buňky, ale NE ZDRAVÉ BUŇKY! Tato schopnost rozpoznávání dobrých a poškozených buněk je rozhodující i pro aktivaci specifické obrany.

Buňky a mechanizmy, které patří do této skupiny reagují na určitý konkrétní antigen, který se musí naučit rozpoznávat. Specifická imunita nastupuje po aktivaci vrozené imunity. Používá organizmem produkované antigenové receptory, které jsou pak dále do organizmu rozšiřovány. Receptory jsou generované pomocí náhodných procesů jako je např. mutace. Tím je zabezpečena obranyschopnost organizmu

před mikroorganizmy, se kterými nikdy předtím nepřišel do styku. Tento typ imunity je pomalejší a má pozdější nástup než mechanizmy vrozené imunity, ale je o mnoho efektivnější právě díky schopnosti reakce na konkrétní antigen a schopnosti vytvářet paměťové buňky, v nichž uložené informace jsou pak využívány při střetnutí s tímto typem infekce.

8.2 Umělé imunitní systémy

V imunitních systémech je možné identifikovat několik vlastností kybernetických systémů. Jeho klíčové vlastnosti se dají shrnout do následujících bodů (de Castro 2002):

- *Rozpoznávání vzorů*
  jednou z vlastností umělého systému imunity je schopnost rozlišovat vlastní vzory od cizích (resp. nebezpečné látky od neškodlivých).
- *Jedinečnost a vlastní identita*
  Každý organizmus má jedinečný imunitní systém. Jedinečnost imunity umožňuje rozpoznat jakoukoliv buňku, molekulu nebo tkáň, která organizmu nepatří a následně ji eliminovat.
- *Funkční samostatnost, rozšiřitelnost a decentralizace*
  Imunitní systém je rozložený po celém organizmu a tvořený velkým množstvím buněk s různými funkcemi (neexistuje centrální prvek, který by řídil chod imunity). Činnost celého systému je výsledkem koordinované aktivity celého systému na buněčné úrovni.
- *Paralelní zpracování informací*
  Paralelní zpracování množství rozdílných informací probíhá přímou komunikací buněk, tj. shromažďováním buněk na specifických místech v systému.
- *Detekce anomálií*
  tj. schopnost systému rozpoznat patogeny a reagovat na ně. Systém je schopen reagovat i na takové patogeny, se kterými se organizmus dosud nesetkal.

- *Robustnost*

  Vysoká robustnost systému je zabezpečena vlastnostmi jakými jsou např. paralelnost a různorodost buněk a jejich vícevrstvé uspořádání. Existuje zde tedy několik současně fungujících mechanizmů, které spolu spolupracují, ale i navzájem soutěží, což vede k vysoké odolnosti systému vůči poruchám.
- *Adaptace (tj. schopnost učení se) a paměť*

  Tyto vlastnosti zabezpečují rychlejší a silnější odezvu systému na sekundární objevení se antigenu, který ohrožoval organizmus.
- *Odolnost*

  Receptory mají jistou toleranci, která umožňuje reagovat na několik podobných antigenů. Imunita se tím zabezpečuje proti molekulárnímu šumu.
- *Samoorganizace*

  Při vstupu cizorodé látky do organizmu je narušena jeho rovnováha a začíná příprava na imunitní reakci na tuto látku. Aby se imunitní systém udržoval v činnosti, potřebuje neustále množství takovýchto neznámých podnětů. V období klidu si tyto podněty poskytují jednotlivé buňky navzájem. Tím se udržuje dynamická rovnováha systému a připravenost na další podněty.

Umělé imunitní systémy se jako samostatná oblast umělé inteligence se vyčlenila poměrně nedávno. Existuje několik alternativ definice umělých imunitních systémů:

- V (Codd 1968) je definován umělý imunitní systém následovně: *„Umělé imunitní systémy jsou inteligentní metodologie inspirované imunitním systémem směřující k řešení problémů skutečného světa“.*
- V (de Castro 2000) je definice rozšířena s cílem zvýraznit rozdíl mezi umělými imunitními systémy a matematickou teoretickou imunologii: *„Umělý imunitní systémy jsou adaptivní systémy inspirované teoretickou imunologií, pozorovanými funkcemi*

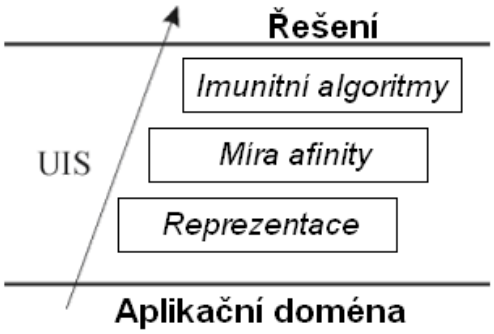
imunity, jejími principy a modely. Tyto systémy jsou aplikované pro řešení problémů".

- V(de Castro 2002) je uvedena velmi obecná definice umělého imunitního systému: *„Umělý imunitní systém je výpočtový systém založený na metaforách přirozených imunitních systémů".*

Všeobecný model umělého imunitního systému

Umělý imunitní systém by měl představovat určité řešení konkrétního problému, které vychází z principů imunity a které je patřičně přizpůsobené své aplikační oblasti. V (de Castro 2002) je uveden následující návrh všeobecného modelu umělého imunitního systému (viz obr. 8.2). Tento model zahrnuje tři základní části:

- *Reprezentace prvků*
  tj. jednotliví agenti systému.
- *Míra afinity*
  tj. mechanizmy ocenění vzájemného působení agenta s okolím, které je zde většinou simulované impulsy.
- *Imunitní algoritmy*
  tj. procesy adaptace, které řídí dynamiku celého systému



Obrázek 8.2: Model umělého imunitního systému.

Jedním z hlavních přístupů k modelování umělých imunitních systémů je modelování pomocí celulárních automatů. Reprezentace umělých imunitních systémů popisuje *způsoby* mapování elementů budovaného umělého imunitního systému a *typy* imunitních komponent. Její konkrétní realizace závisí na aplikaci. Většina modelů umělých

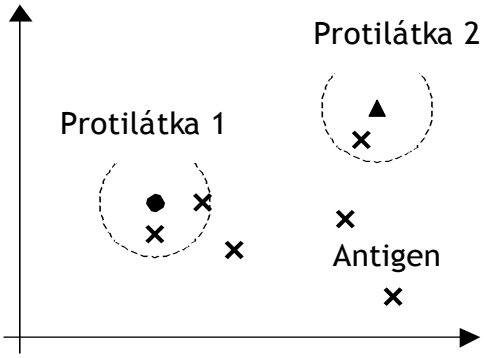
imunitních systémů používá korespondenci dvojice „antigen - protilátka“. Koncept antigenu zodpovědný za konkrétní instanci problému (tj. vzor, pozorovaná situace atd.) a vhodná protilátka jsou detektorem a řešením tohoto problému. Umělý imunitní systém se snaží najít resp. vytvořit (nadaptovat) co nejlepší řešení konkrétního problému. Vhodnost řešení pak odpovídá afinitě mezi antigenem a protilátkou.

Biologický imunitní systém	**Umělý imunitní algoritmus**
Antigen	Problém
Protilátka	Řešení problému
Afinita	Vhodnost řešení
Produkce protilátek	Použití genetických operátorů k tvorbě nových řešení
Produkce protilátek z paměťových buněk	Vyvolávání/obnovování minulých úspěšných řešení

Tabulka 8.1. Reprezentace mezi typickým modelem UIS a jeho biologickou motivací.

V rámci umělých imunitních systémů se stupeň slučitelnosti dvou elementů nazývá *míra afinity*. V imunitě se většinou jedná o dvojice „protilátka – antigen“, jež jsou na sebe vázány chemickými vazbami. Pokud předpokládáme, že vlastnosti prvku ***m*** se dá vyjádřit jako *n* - tice atributů $\boldsymbol{m} = (m_1, m_2, \ldots m_n)$, potom konkrétní element reprezentuje bod v *n* - rozměrném příznakovém prostoru $\boldsymbol{S^n}$. Afinita dvou prvků pak odpovídá vzdálenosti *D* mezi dvěma body v definovaném prostoru.

Biologický imunitní systém má určitou toleranci afinity, aby byl odolný na drobné varianty struktury bílkovin, tzv. *molekulární šum*. V příznakovém prostoru je tato tolerance afinity reprezentována oblastí vlivu okolí konkrétního bodu (viz obr. 8.3). Tato oblast je nazývána *práh vzájemné reakce* nebo-li *oblast rozpoznávání*. Protilátka je schopná rozeznat více příbuzných typů antigenů.



Obrázek 8.3: Příznakový prostor a oblasti afinity jednotlivých protilátek

Imunitní algoritmy

Imunitní algoritmus určuje, jakým způsobem je vytvořen model umělého imunitního systému (tj. způsob produkce imunitních prvků, selekce, podmínky aktivace – interakce aj.).

Mějme množinu vlastních prvků, které chceme chránit a mějme definovanou velikost množiny detektorů, potom **algoritmus negativní selekce (**Timmis 2000) pracuje následovně:

1. *Inicializace*
   náhodně vygenerujeme množinu kandidátů na detektory.
2. *Cenzura*
   pokud nebyla vyprodukovaná množina detektorů dané velikosti, potom vyhodnotíme afinitu mezi každým vlastním prvkem a kandidátem.
3. *Selekce*
   pokud kandidát rozpozná některý element z vlastních množiny, je tento kandidát eliminovaný, jinak je umístěný do množiny detektorů.
4. *Monitorování*
   fáze „života“ systému, tj. množina detektorů je porovnávána s kontrolovanou množinou a pokud je nějaký prvek rozpoznaný, je nevlastní.

Tento algoritmus je poměrně všeobecný a existuje velké množství modifikací, které optimalizují jeho činnost. Algoritmus negativní selekce se často používá při problémech týkajících se detekce anomálií, jako např. počítačová bezpečnost anebo odhalování chyb v obrazech.

Mějme množinu antigenů, které chceme rozpoznávat a velikost množiny protilátek, kterou chceme vyprodukovat, potom **algoritmus klonální selekce (**Timmis 2000) pracuje následovně:

1. *Inicializace*
   náhodně vyprodukujeme populaci imunitních buněk.
2. *Generování populace*
   pro každý antigen vybereme ty buňky, které mají nejvyšší afinitu k antigenu.
3. *Generování klonů*
   čím lépe daná buňka antigen rozpoznává, tím víc kopií buňky vyprodukujeme.
4. *Mutace*
   každou novou buňku zmutujeme podle pravidla – čím je afinita větší, tím budou mutace dané buňky menší.
5. *Vyhodnocení afinity*
   pro každou zmutovanou buňku vyhodnotíme afinitu k antigenu.

I když se tento algoritmus velmi podobá evolučním algoritmům, existuje několik rozdílů – u algoritmu klonální selekce jsou buňky selektované a mutované v poměru k hodnotě afinity, přičemž mutace probíhá u každé nové buňky. V evolučních algoritmech jsou mutace náhodné a jsou realizované v podstatně menší míře. Algoritmus klonální selekce také nepoužívá operátor křížení.

8.3 Aplikace umělých imunitních systémů

Aplikace umělých imunitních systémů je možné rozdělit do dvou hlavních oblastí. První oblast se zabývá *technicky zaměřenými* umělými imunitními systémy a druhá oblast prezentuje aplikace umělých imunitních systémů *v oblastech biologie*. Aplikace v biologii se zaměřuje na simulování reálných imunitních mechanizmů s cílem lepšího pochopení činnosti pro lékařský výzkum, farmaceutický nebo chemický průmysl. Dále se především zaměříme na aplikace v technicky zaměřených oblastech, jejichž cílem je vývoj nových výpočetních

systémů pro technické oblasti, např. pro informatiku nebo umělou inteligenci. Pomocí algoritmu klonální selekce lze popsat princip kolektivního chování robotů, kdy agenti musí umět komunikovat navzájem mezi sebou i s prostředím, ve kterém se nacházejí. Pomocí adaptivního optimalizačního algoritmu inspirovaného teorií imunitní sítě lze řešit problému obchodního cestujícího (Toma 1999). Příkladem systému pro rozpoznávání vzorů je systém Immuno-81 (Kephart 1994) založený na kontrolovaném učení se klasifikace vzorů. I když byl tento systém použitý pouze k analýze lékařských údajů, je to v podstatě analyzátorem libovolných dat.

Umělý imunitní systém je nejčastěji používaný na ochranu a zabezpečení počítačů, protože je velmi dobrý v rozpoznávání a eliminací infekčních mikroorganismů, má také schopnost dynamicky se adaptovat na neznámé nemoci. Tyto vlastnosti jsou žádány v počítačové bezpečnosti. Počítačovou bezpečnost můžeme rozdělit do dvou hlavních oblastí:

- *Detekce počítačových virů a jejich eliminace:* Nejznámější z této oblasti je imunitní antivirový program od firmy IBM, ale dostupný je jen jeho popis v (Kephart 1994). Autoři programu vycházeli z předpokladu, že program *X* infikovaný virem, může být upravený zpět na svůj originál. Když virus infikuje program - vnese do něj své funkce, přičemž musí být schopný rekonstruovat původní program tak, aby žádný s bytů původního programu nebyl zničený. Pro tuto skupinu „nepřepisujících“ virů platí, že infikovaný program *X* je reverzibilní (schopný zpětného procesu) transformací *X'*. Při aktualizaci souboru se však tato transformace nezachovává. Po instalaci imunitního antivirového programu se vypočítává tzv. *fingerprint* (otisk) každého spustitelného programu a ten je uložen do imunitní databáze. Fingerprint je nejčastěji 100 bytová informace o programu, obsahující různé kontrolní součty, data, velikosti souborů v programu apod. Autoři tohoto antivirového programu kladou důraz na mechanizmus rozpoznání viru a na adaptabilitu na nové viry. Hlavním předpokladem úspěchu obrany

před viry je dosažení stavu, kdy je rychlost rozpoznání a následná likvidace viru větší než rychlost množení viru. Možnost udržování a obnovování znalostí v imunitních databázích a rychlé šíření těchto informací pomocí imunitních agentů může vést k dosažení takového stavu.

- *Detekce nedovolených operací v počítačových sítích (útoky na sítě):* Nelegální vniknutí do počítačových sítí může být analogií výskytu infekce v organizmu, kterou je potřeba detekovat a určitým způsobem na ni reagovat. V (Kim 1999) byly navrženy požadavky na vlastnosti umělého imunitního systému, který byl použitý pro rozpoznávání útoků na sítě: (1) *distribuovanost* detekční sítě, (2) *samoorganizovanost* zabezpečená negativní selekcí a evolucí knihoven genů pro tvorbu detektorů a (3) „*lehkost*“ (lightweight) systému ve formě možnosti částečného rozpoznávání.

Kontrolní otázky:

1. Co se rozumí pod pojmem „umělý imunitní systém“?
2. Jak lze umělé imunitní systémy modelovat?
3. Jaké jsou aplikace umělých imunitních systémů?

Úkoly a otázky k textu:

Nalezněte na Internetu další simulátory umělých imunitních systémů a vyzkoušejte je na vlastní aplikaci.

Korespondenční úkol:

Nalezněte např. na Internetu další možné využití umělých imunitních systémů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Tato kapitola představuje především úvod do problematiky umělých imunitních systémů. Imunitní systém vykazuje velmi zajímavé vlastnosti, nejen z lékařského hlediska, např. přizpůsobivost, robustnost a schopnost učení se. Seznámíme se zde především s využitím vlastností imunitních systémů v technicky zaměřených oblastech.

Pojmy k zapamatování

- biologický imunitní systém,
- umělý imunitní systém,
- antigen,
- afinita,
- protilátka.

9 Výpočty na bázi DNA

V této kapitole se dozvíte:

- Co je to deoxyribonukleová kyselina a jaký je její význam pro život organismů.
- Jak využíváme principů DNA při výpočtech.

Po jejím prostudování byste měli být schopni:

- Charakterizovat co je to deoxyribonukleová kyselina a jaký je její význam pro život organismů.
- Vysvětlit, jak využíváme principů DNA při výpočtech.

Klíčová slova této kapitoly:

Deoxyribonukleová kyselina (DNK), ribonukleová kyselina (RNK), adenin, guanin, thymin, cytosin Adlemanův experiment.

Průvodce studiem

Tato kapitola představuje především úvod do problematiky DNA computingu. Seznámíte se zde především se strukturou DNA, její funkcí, replikací a s využitím principů DNA při výpočtech (tj. Adlemanův experiment).

9.1 Deoxyribonukleová kyselina

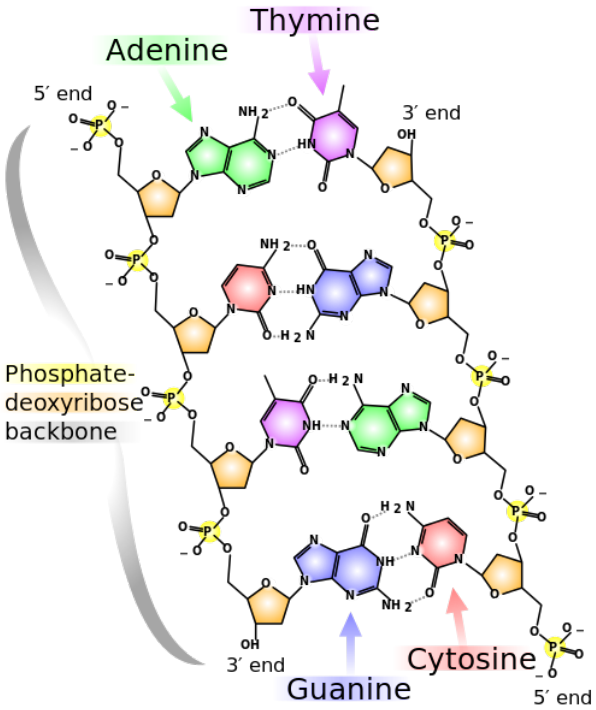
DNA (*Deoxyribonukleová kyselina - DNK*) je nositelkou genetické informace všech organismů s výjimkou těch nebuněčných organismů, u nichž hraje tuto úlohu **RNA** (*RNA-viry, virusoidy a viroidy*). DNA je pro život nezbytnou látkou, která ve své struktuře kóduje a buňkám zadává jejich program a tím předurčuje vývoj a vlastnosti celého organismu. Genová výbava člověka obsahuje přibližně 3,2 x 109 vazebných párů.

Kdyby se měla jejich struktura popsat, jejich začátečními písmeny, vznikla by kniha s více než 500 000 stranami.

Struktura DNA

DNA je biologická makromolekula - polymer, dvoušroubovice tvořená dvěma řetězci nukleotidů v obou vláknech. Jednotlivé nukleotidy se skládají ze tří složek:

- *fosfátu* - vazebný zbytek kyseliny fosforečné;
- *deoxyribózy* - pětiuhlíkový cukr – pentóza;
- *nukleové báze*- konkrétní dusíkaté heterocyklické sloučeniny.



Obrázek 9.1: Struktura DNA

V DNA se v různých kombinacích vyskytují čtyři *nukleové báze*: purinové báze jsou **Adenin** (**A**) a **Guanin** (**G**) a pyrimidinové báze jsou **Thymin** (**T**) a **Cytosin** (**C**). Na obrázku 9.1 je uvedena struktura DNA. Do řady za sebou jsou uspořádány *nukleotidy*. Jejich spojení v řadě zajišťují *fosfátové zbytky*, které spojují uhlík 3' jedné deoxyribózy s uhlíkem 5' druhé deoxyribózy. Směr vláken se označuje právě podle orientace deoxyribózy v něm, tedy: směr 3´→5´ a opačný směr 5´→3´. Na uhlík 1' deoxyribózy se váží dusíkaté báze (A,G,C,T). Ty se spojují

navzájem s odpovídající bází z protějšího řetězce, podle jednoduchého klíče:

A ↔ **T** + **T** ↔ **A** *(vzájemně jsou spojeny dvěma vodíkovými vazbami)*

C ↔ **G** + **G** ↔ **C** *(vzájemně jsou spojeny třemi vodíkovými vazbami)*

Jedná se o tzv. *komplementaritu bází*, z níž vychází vzájemná komplementarita obou vláken DNA. Vždy je na určité pozici v molekule jeden nukleotid z dvojice a v protějším vlákně druhý z nich. Takto se uchovává v každém z vláken tatáž informace, pouze s tím rozdílem, že se jedná o vzájemný „negativ". Vzájemné spojení nukleotidů není uskutečněno regulérní chemickou vazbou, ale „jen" vodíkovými můstky. Primární struktura DNA je dána pořadím nukleotidů a jejich sekvencí a nese v sobě genetickou informaci.

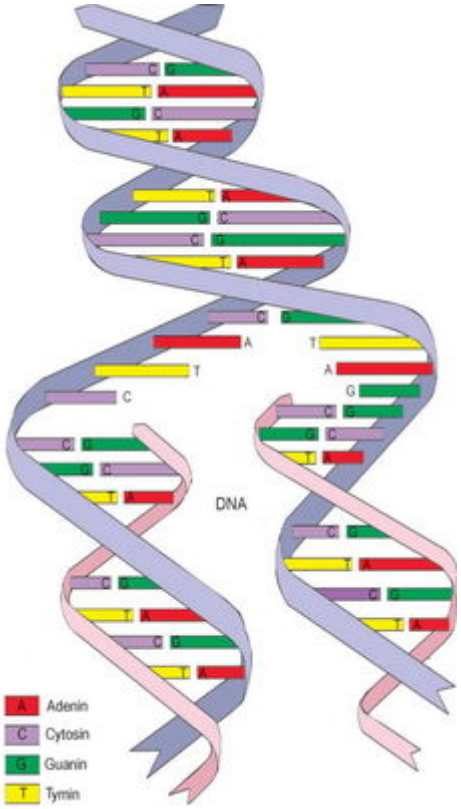
Funkce DNA

DNA uchovává ve své struktuře genetickou výbavu celého jedince, nicméně jen určitá část této informace, je v konkrétní buňce realizována. Pro každou konkrétní buňku je DNA určitou „kuchařkou", podle níž specificky realizuje svůj program. Většina genů potřebných pro život se nachází v jádře na chromozómech buňky. Přepis sekvence DNA do sekvence RNA se nazývá transkripce. Část molekul RNA pak po přesunou z jádra do cytoplazmy (tekuté prostředí buňky) a slouží jako šablona pro translaci. Tímto způsobem, na základě genetického kódu, vznikají nové bílkoviny

Replikace DNA

Replikace molekuly DNA znamená vlastně její symetrické zdvojování podle předlohy, viz obrázek 9.2. Má-li DNA plnit svou funkci a předávat svou informaci do dceřiných buněk (a dalších generací organizmu), musí být schopná zdvojení sebe sama. Jedná se o enzymaticky řízený proces přesného kopírovaní sekvence DNA na základě komplementarity nukleových bází. Tímto způsobem se z jedné původní molekuly DNA vytvoří dvě molekuly dceřiné, každá s jedním vláknem původním a s jedním vláknem nově dosyntetizovaným. V následujícím

(vlastně zároveň probíhajícím) procesu dělení buněk se každá z molekul začne skládat (do vyšších úrovní struktury) do chromozómů a je spolu s ním pečlivě řízeným procesem oddělena do jiné dceřiné buňky.

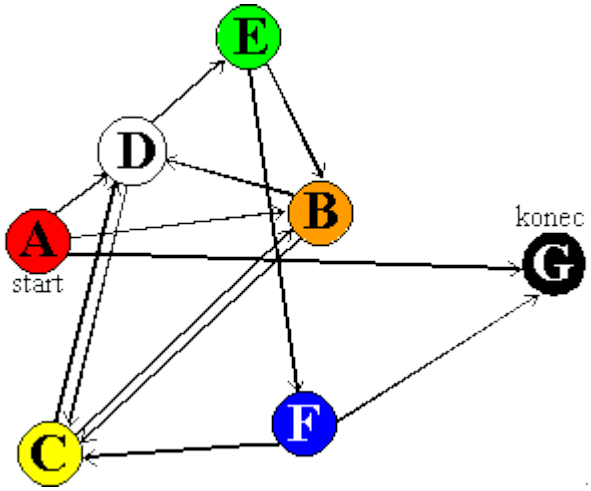


Obrázek 9.2: Replikace molekuly DNA. Světle červená vlákna představují nově formovaná vlákna DNA, která vznikají na základě komplementárního párování bází s vlákny původní molekuly, která jsou naznačena modře.

9.2 Využití principů DNA při výpočtech

Příkladem, na kterém se často demonstruje využití DNA výpočtů, je problém obchodního cestujícího (tj. nalezení hamiltonovské cest, viz obr. 9.3). Úlohu je možné řešit pouze hrubou silou, tedy stanovením všech možností a jejich seřazením dle vzdálenosti. V tomto případě by nebylo cílem najít nejkratší cestu, ale cestu libovolnou. Ne všechny body grafu jsou totiž spolu spojeny, cesty jsou navíc pouze jednosměrné a předem není jasné ani to, kolik řešení existuje. Podle charakteru zadání nemusí být úloha řešitelná vůbec. Problém existence

hamiltonovské cesty v orientovaných grafech je NP-úplný. Znamená to, že dosud nebyl nalezen algoritmus, který by pro libovolný orientovaný graf a pro jeho libovolné dva vrcholy *A* a *G* (viz obr. 9.3), po *n* krocích rozhodl, zda existuje cesta z *A* do *G*, přičemž každý vrchol grafu by byl navštívený právě jednou. Ověřování existence hamiltonovské cesty v orientovaném grafu je NP-úplné problém, protože počet potřebných kroků na vyřešení problému roste exponenciálně s velikostí grafu.



Obrázek 9.3. Problém nalezení hamiltonovské cesty v grafu.

Uveďme si nejprve hrubý náčrt algoritmu ověřujícího existenci hamiltonovské cesty z uzlu *A* do uzlu *G* v orientovaném grafu (viz obr. 9.2) se šesti vrcholy:

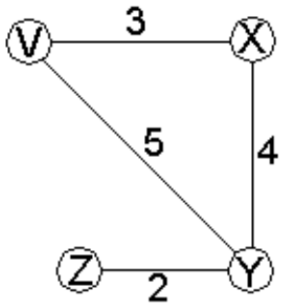
1. Generuj náhodné cesty v grafu.
2. Ponechej si jen cesty vycházející z uzlu *A* a končící v uzlu *G.*
3. Vyluč všechny cesty, jejichž délka je různá od *šesti.*
4. Ponechej si jen cesty, které navštíví každý vrchol nejvíce jednou.
5. Pokud je množina zbylých cest neprázdná, je nalezeno řešení úlohy, v opačném případě řešení úlohy nalezeno není.

Nyní si popíšeme jak se problém hamiltonovské cesty v orientovaném grafu může řešit pomocí molekul DNK a operací s nimi.

Problém obchodního cestujícího v *DNA computingu*

V DNA computingu (Kvasnička 2000) budeme při řešení úlohy postupovat následujícím způsobem: Připravíme si sadu nukleových kyselin (molekul, sekvenci nukleotidů), které budou odpovídat

jednotlivým městům. Jiné sekvence budou reprezentovat cesty mezi uzly. Například cesta mezi bodem ***V*** a ***X*** bude mít na počátku výpočtu báze komplementární s koncem sekvence města ***V*** a na konci výpočtu bude mít báze komplementární se začátkem řetězce reprezentujícího bod ***X***. Délka úseku mezi komplementárními částmi cesty pak bude odpovídat vzdálenosti mezi body (viz obr. 9.4).



Řešením je cesta *V-X-Y-Z*. Zakódování do DNA může v tomto případě vypadat například takto:
města:***V**: A-A*, ***X**: C-C*, ***Y**: T-T*, ***Z**: G-G*
cesty: ***VX:*** T-T-G-A-G, ***VY:*** T-C-C-G-A-A-A, ***XY:*** G-A-G-G-C-A, ***YZ:*** A-G-A-C.
Řetězec odpovídající řešení, bude vypadat následovně:

T-T-G-A-G G-A-G-G-C-A A-G-A-C
A-A C-C T-T G-G

Obrázek 9.4: Problém obchodního cestujícího.

Ve zkumavce smícháme všechny připravené sekvence nukleové kyseliny, v nichž by mělo být i řešení. V okamžiku proběhne obrovské množství reakcí, jednotlivé řetězce se k sobě budou přibližovat, dotýkat a zase vzdalovat. Tam, kde se k sobě přiblíží komplementární řetězce, se vytvoří mezi párovými bázemi vodíkové vazby a obě části již zůstanou pohromadě. Nyní tedy máme směs všech možností, protože reagující látky jsme přidali ve velkém množství, tak jsou zde jednotlivé kombinace zastoupeny ve větším počtu. Z našeho hlediska se prostě spoléháme na to, že je statisticky víceméně vyloučeno, aby nějaký způsob poskládání řetězců zcela scházel.
Nyní nám zbývá najít správné řešení, tj. analytickými metodami od sebe látky oddělit dle jejich vlastností. Hypoteticky můžeme postupovat třeba následujícím způsobem: ke směsi přidáme "*činidlo 1*", sekvenci

nukleotidů se začátkem molekuly ***V***. Činidlo 1 má na sobě navázán atom železa. Všechny cesty, vyhovující našemu řešení, začínají v bodě ***V***. Vyhovující řetězce musí mít tedy začátek bodu ***V*** prázdný, bez "nalepeného" komplementárního řetězce "cesty". Na toto místo se "přilepí" činidlo 1. Všechny vyhovující molekuly jsou tedy nyní označené železem. Směs vystavíme účinkům magnetického pole a látky rozdělíme na magnetické a nemagnetické. Nyní jsme tedy dostali množinu všech cest, které vyhovují podmínce "začátek v bodě V", a nějakou enzymaticky specifickou reakcí vazbu bod *V - činidlo 1* rozštěpíme. Směs opět podrobíme účinkům magnetického pole a oddělíme činidlo 1 (frakce, která nás nyní zajímá, je samozřejmě ta nemagnetická).

Můžeme pokračovat úplně stejně, až na to, že nyní aplikujeme "*činidlo 2*", tedy opět nějak označenou sekvenci nukleotidů komplementárních s koncem bodu ***Z***. Tím dostaneme množinu molekul, které již odpovídají pouze cestám začínajícím v bodě ***V*** a současně ukončeným v bodě ***Z***.

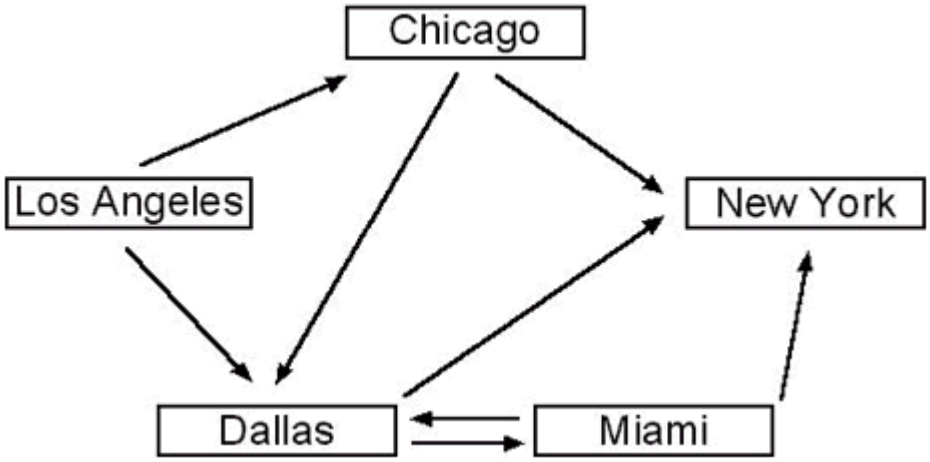
Analogickými metodami získáme nakonec směs molekul, jež přesně vyhovuje zadaným podmínkám. Každá z těchto molekul obsahuje sekvence kódující všechny body, a to právě jednou. Molekuly se však liší cestami mezi body. Námi hledaná cesta je právě ta nejkratší, a protože délka "nekomplementárních" částí cest odpovídá délce skutečné, musíme získat nejkratší molekulu. Taková molekula by např. měla být také nejlehčí, jednotlivé řetězce nukleové kyseliny se dle své velikosti liší i v dalších fyzikálních vlastnostech (body tání, chování v gravitačním poli nebo v elektromagnetickém poli), takže můžeme použít klasické postupy chemické analýzy typu chromatografických kolon. Směs tedy opět nějak rozdělíme a "přečteme" správné řešení. V praxi samozřejmě přečteme molekul více a zjistíme, zda jsou shodné. Pokud ne, ověříme, jaká z nich kóduje výhodnější cestu.

Adlemanův postup

Leonard Adleman využil biologické procesy na simulaci matematických výpočtů a zrealizoval příslušný pokus se sedmi uzly (Aldeman 1994). Podstata jeho přístupu je v kódovaní informace na řetězcích DNK a ve

využití enzymů na simulaci jednoduchých výpočtů. Adleman postupoval analogicky předešlému případu. Připravil si sekvence vždy 20 nukleotidů odpovídající jednotlivým místům a cestám mezi nimi - zahrnul zde pouze řetězce reprezentující existující cesty, včetně příslušné směrové orientace. Sekvence pro cestu pak obohatil pouze části komplementární s řetězci příslušných měst, tj. scházela volná, nepárová část vyjadřující vzdálenost. Cesty byly tvořeny dvaceti nukleotidy, počátečních deset komplemetárních se začátkem cesty, dalších 10 s jejím koncem.

Nechť je problém zadaný následujícím grafem a naším výchozím vrcholem je *Los Angeles* a cílovým vrcholem *New York*. Naší úlohou bude nalézt cestu, která začíná v *Los Angeles* a končí v *New Yorku* (viz obr. 9.5).

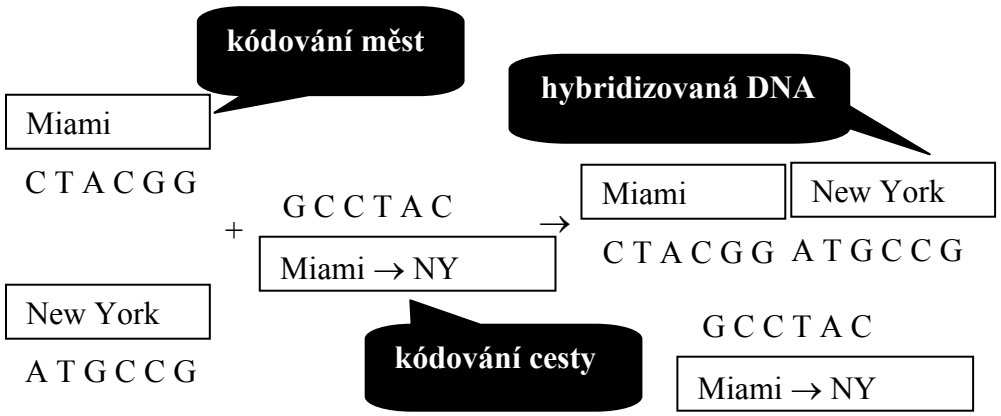


Obrázek 9.5: Problém obchodního cestujícího řešený Adelmanem.

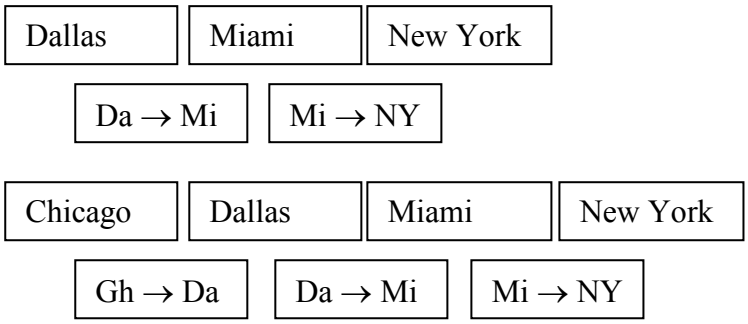
Algoritmus řešení s použitím DNK bude realizovaný následovně:
Vytvoříme jedinečnou DNK sekvenci pro každé město, kódování vrcholů je pak dáno takto:

Los Algeles	*GCTACG*
Chicago	*CTAGTA*
Dallas	*TCGTAC*
Miami	*CTACGG*
New York	*ATGCCG*

Pro každou cestu z města *X* do města *Y* se vytvoří sekvence DNK, která reprezentuje cestu z města *X* do města *Y*. Tato cesta je komplementární pro druhou polovinu DNK kódu města *X* a pro první polovinu DNK kódu města *Y*.



Následovně je uveden příklad některých možných kombinací cest



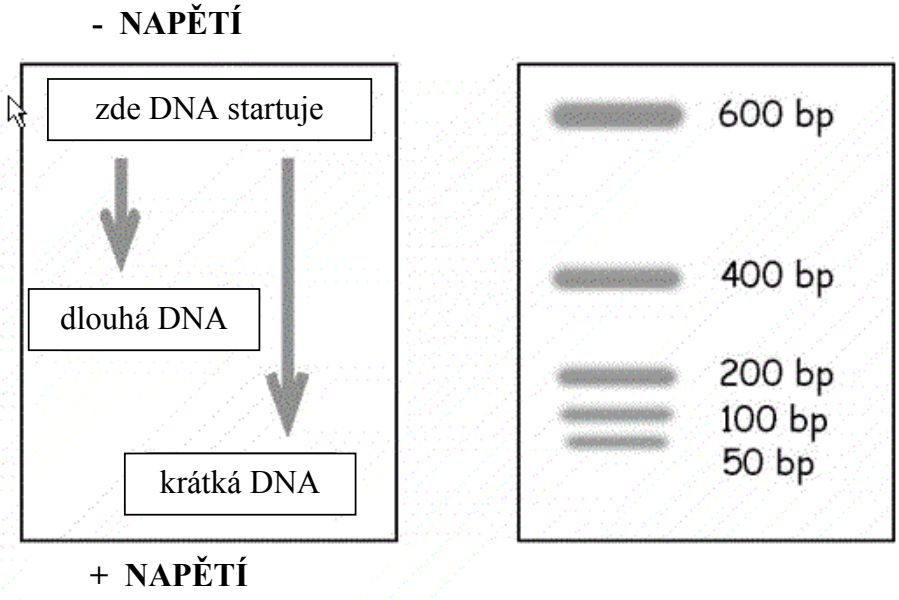
apod.

Dále ilustrujeme způsob, jak lze kódovat cestu mezi oběma městy. Je náročné separovat konkrétní DNK z výsledku dosáhnutého v kroku 1. Proto bude cílová DNK, která začíná v městě *Los Angeles* a končí v městě *New York* vícenásobně kopírovaná. Dosáhneme tak stavu, kdy zkumavka bude obsahovat mnoho správných a relativně málo nesprávných řetězců. Je to podobný případ, jako když do zásuvky s jednou, nebo dvěma barevnými ponožkami přidáme sto černých. Je téměř jisté, že pokud potom náhodně vybereme jedny, budou černé. DNK kopírujeme pomocí enzymu nazývaného *polymeráza*, který nám

umožňuje replikovat přednostně ty řetězce DNK, jež začínají a končí na námi požadovanou sekvenci.

Potom, podle váhy molekul DNK, separujeme od zbytku ty řetězce DNA, které kódují cestu právě přes pět vrcholů. Je použitá prostorová mřížková struktura a to tak, aby menší řetězce DNK přešly rychleji, než větší, jejichž průchod je pomalejší.

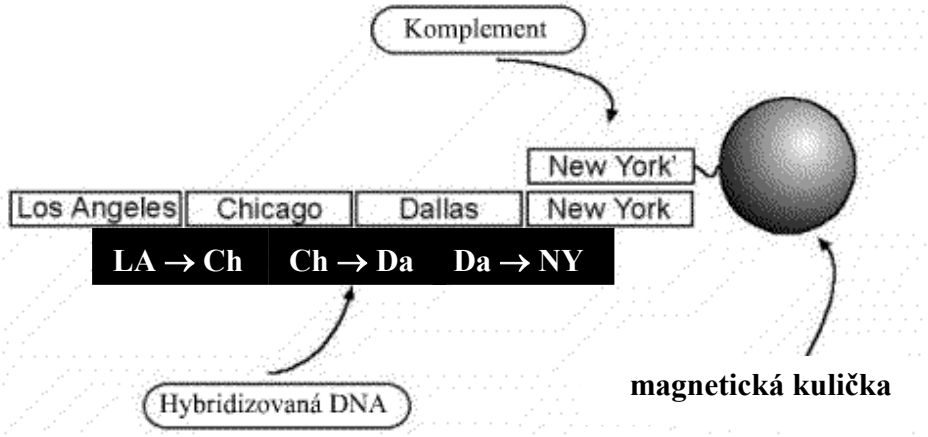
Použitá metoda se nazývá *gelová elektroforéza* a je založená na tom, že DNK protlačíme přes gelovou matici pomocí elektrického pole. DNK molekula má negativní elektrický potenciál. Bude tedy přitahovaná kladným pólem elektrického pole. Separování DNK podle délky řetězce dosáhneme na základě toho, že struktura gelové matice zpomaluje pohyb DNK molekul v závislosti na jejich velikosti. Po určitém čase budou molekuly DNK separované do různých částí matice na základě velikosti řetězce DNK. Znázorňuje to následující obrázek 9.6 (*bp* znamená bázový pár).



Obrázek 9.6: Gelová elektroforéza.

Na to, aby nám zůstaly jen řetězce, které obsahují všechna města, použijeme postup nazývaný *afinní purifikace*. Využijeme magnetickou kuličku, která postupně „podrží“ řetězce DNK, které obsahují sekvence DNK kódující jednotlivé vrcholy grafu. Děje se to na základě komplementu navázaného na tuto magnetickou kuličku. Nejprve tedy

"podržíme" řetězce obsahující sekvenci DNK pro *New York*. Ostatní řetězce DNK jsou odstraňované. Zbavíme se tak řetězců, které neobsahují sekvenci DNK kódující *New York*. Postup opakujeme pro každé jednotlivé město (tj. sekvenci DNK, která jej kóduje). Nakonec nám zůstanou jen takové sekvence DNK, které v sobě mají zakódované všechny vrcholy grafu. Tento princip je uveden obrázku 9.7. Tyto řetězce DNK kódují hledanou cestu a zbývá jen určit nejkratší z nich.



Obrázek 9.7: Afinní purifikace.

Výše popsaný problém obchodního cestujícího se sedmi body zabral Adlemanovi přibližně sedm dní práce v laboratoři. Stávající PC vyřeší úlohu za necelou sekundu. Člověk by měl úlohu vyřešit do půl hodiny. Výhodu, kterou člověk může použít na rozdíl od klasického i DNA počítače, však v tomto konkrétním případě představuje "poloanalytickou" metodu, kdy úlohu řešíte raději protisměrně, z koncového bodu. Pokud vás tato možnost napadne, trvá řešení dokonce jen několik sekund.

Kontrolní otázky:

1. Co je to deoxyribonukleová kyselina a jaký je její význam pro život organismů?
2. Jak využíváme principů DNA při výpočtech?
3. Vysvětlete Adlemanův experiment.

Korespondenční úkol:

Nalezněte např. na Internetu informace týkající se DNA počítačů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Tato kapitola představuje především úvod do problematiky DNA computingu. Seznámili jste se zde především se strukturou DNA, její funkcí, replikací a s využitím principů DNA při výpočtech (tj. Adlemanův experiment).

Pojmy k zapamatování

- deoxyribonukleová kyselina (DNK),
- ribonukleová kyselina (RNK),
- adenin,
- guanin,
- thymin,
- cytosin,
- Adlemanův experiment.

10 Robotika

V této kapitole se dozvíte:

- Jaké jsou definice robotů?
- Jak probíhá zpracování přirozeného jazyka?
- Jaké jsou principy počítačového vidění?
- Jaké jsou techniky zpracování 2D obrazů?

Po jejím prostudování byste měli být schopni:

- Vysvětlit jak je chápán robot v USA a v Japonsku.
- Objasnit princip rozpoznávání hlasu a řeči.
- Vysvětlit principy počítačového vidění.
- Objasnit jaké jsou techniky zpracování 2D obrazů.

Klíčová slova této kapitoly:

Robot, textový korpus, morfologická analýza, počítačová lematizace, segmentace obrazu, digitalizace, prahování, klasifikace.

Průvodce studiem

Tato kapitola se snaží vnést jasno do chápání pojmu robot tak, aby nešlo jen o formální vnější rysy (pohyb, ruka, oko), ale zejména o styl řešení úloh (záměr, působnost).

K rozpoznávání lidské řeči pomocí počítače lze přistupovat ze dvou rozdílných hledisek. Buď se rozpoznávají hlasy různých řečníků od sebe nebo je nezávisle na řečníkovi zjišťován obsah jeho promluvy.

Jako technická disciplína bývá počítačové vidění označováno za kombinaci počítače, snímacího zařízení a softwaru, který je schopný snímanou předlohu zpracovat nebo klasifikovat.

Robot je stroj vystavěný na principech umělé inteligence. Podle současného stavu a trendů je jasné, že robotika jako taková se stane nedílnou součástí naší civilizace. Její aplikace lze očekávat téměř na všech úrovních. V domácnosti jako domácí sluhy, v lékařství jako experty na různé speciální zákroky, ale také jako diagnostické systémy v podmínkách, kde nebude lékař. Ani mořské hlubiny nezůstanou výjimkou. I pro toto prostředí jsou dnes vyvíjeny roboty, které mají tvar ryby. Tyto roboty mají samozřejmě výzkumný charakter a slouží k odhalování tajemství mořských hlubin. To je vcelku významný úkol, vezme-li se v úvahu fakt, že právě moře by mohlo pomoci při surovinové a potravinové krizi, která velmi pravděpodobně lidstvo čeká

10.1 Co je to robot

Existuje řada různých definic robotů. Ve snaze poskytnout všeobecně přijatelnou definici stanovila Mezinárodní organizace pro standardizaci definici robota v normě ISO 8373, kde je robot definován jako „*automaticky řízený, opětovně programovatelný, víceúčelový manipulátor pro činnost ve třech nebo více osách, který může být buď upevněn na místě nebo mobilní k užití v průmyslových automatických aplikacích*“. Tato definice se používá při porovnávání počtu robotů v různých zemích.

Oproti definici ISO mají země, jako např. USA a Japonsko, odlišné definice robotů. Japonsko má velmi mnoho částečných robotů, protože zde je považováno za roboty více strojů. Vzhledem k tomu, že Japonsko i USA jsou důležitými subjekty ve vývoji robotů, uvádíme definice používané v těchto zemích.

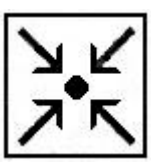
Americký institut pro robotiku (RIA) definuje robota jako „*přeprogramovatelný vícefunkční manipulátor určený k přemisťování materiálu, součástek, nástrojů nebo specializovaných přístrojů pomocí různě naprogramovaných pohybů za účelem provádění různých úkolů*“. RIA rozlišuje čtyři třídy robotů:

- manipulační zařízení s ručním řízením;
- automatické manipulační zařízení s předem určenými cykly;
- programovatelné, řízené servoroboty pohybující se od bodu k bodu po spojité trajektorii;
- roboty, které k inteligentnímu pohybu vyžadují informace z prostředí.

Japonské robotické sdružení (JARA) rozděluje roboty do šesti tříd:

- ruční manipulační zařízení uváděné do chodu operátorem;
- robot s pevnou sekvencí;
- robot s proměnlivou sekvencí snadno měnitelnou řídicí sekvencí;
- přehrávací robot (playback), který může zaznamenat pohyb pro pozdější opakování;
- číslicově řízený robot s pohybovým pro gramem a ručním ovládáním;
- inteligentní robot, který rozumí prostředí a je schopen dokončit úkol navzdory změnám v provozních podmínkách.

Použití umělé inteligence se v budoucnosti bude vyvíjet dvěma hlavními směry, a to v samostatných systémech a v systémech buď s centrálně umístěnou inteligencí nebo s tzv. distribuovanou umělou inteligencí. Pod pojmem samostatný systém lze rozumět např. plně samostatného robota, který se bude umět samostatně chovat v jakémkoliv prostředí bez pomoci vzdáleného supervizora, ať už jím bude člověk či nadřízený systém. Takováto zařízení jsou vyvíjena již dnes pro různé lidské činnosti, například roboty pro výzkum vulkánů či práci v jiných životu nebezpečných prostředích. Rovněž jsou používaní v armádě pro průzkum nepřátelského prostředí ze vzduchu či země.



Roboti s centrálně umístěnou inteligencí

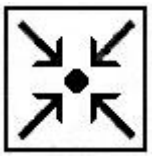
Ukázkou robota s centrálně umístěnou inteligencí, který je dnes jako prototyp zkoušen v námořnictvu USA, může být např. malý robotek – kamikadze, jež má za úkol vyhledávat miny a svým výbuchem je

likvidovat. Má tvar šestinohého brouka a převrátí -li se, jednoduše si „překloubí“ nohy a pokračuje v cestě.

Další dnes již funkční ukázkou je robot pro hlídání vzdušného prostoru vyráběný ve Velké Británii, který je plně samostatný. Má zásobník protiletadlových střel a samostatně pomocí radaru identifikuje objekt a rozhoduje o jeho zařazení (tj. zda je viděné letadlo nepřítele). Robot musí nejprve získat obraz letadla (radarový odraz), zpracovat jej („vytažení“ relevantních informací ze signálu), oklasifikovat (rozhodnout, co je daný objekt zač) a rozhodnout o zásahu (tj. zda daný letoun sestřelil nebo ignoroval).

Ani jiné roboty nejsou výjimkou. Například prvky umělé inteligence v sobě obsahovalo i vozítko Sojuner ze sondy PathFinder. Pokud by sonda operující na Marsu či jiné vzdálené planetě neměla samostatnou, byť velmi primitivní inteligenci, která by umožnila samostatné jednání a orientaci v prostoru musela by při každém problému čekat dlouhé minuty na povel ze Země, což je nejen nehospodárné, ale i nebezpečné (např. zpráva o průrvě + povel „stop“ ze Země).

Roboti s distribuovanou umělou inteligencí



Systémy s distribuovanou umělou inteligencí jsou hierarchicky stupňovány tak, že nejvyšší inteligence je na nejvyšším stupni řízení a rozhodování a s klesajícími stupni „slábne“. Klasickým příkladem může být úsek továrny řízený počítači, které rozhodují o chodu jako celku a „nestarají“ se o činnost jednotlivých částí. Avšak s poklesem stupně řízení do nižších úrovní mizí význam celkového chodu systému a do popředí vystupuje význam činnosti jednotlivých jeho prvků.

Tyto systémy naleznou uplatnění nejen v továrnách, ale i jako samostatné jednotky. Např. robot – mozek, který se bude pohybovat se svým týmem robotů vykonavatelů v daném prostředí (cizí planety, nepřátelská území, doly ...) a vykonávat daný úkol. Celý systém robotů je samozřejmě výhodnější než jeden samostatný robot, např. při havárii v případě samostatného robota – mozku bychom přišli o jednoho vykonavatele, jehož funkci by převzal jiný robot a mise by nebyla

ohrožena. To je jistě jednodušší a levnější než v případě robota jako samostatného systému.

Lze rovněž očekávat, že nastane věk robotí miniaturizace. Již dnes existují roboty, které si svou velikostí nezadají s ptáky či obry hmyzí říše. Jejich využití je především v oblasti špionáže, ale lze je používat kdekoliv, kam se člověk nedostane. Od čištění nepřístupných míst až po superjemné opravy různých mikrostruktur. Mohou se podílet na likvidaci hmyzích škůdců, nebo na likvidaci ekologických katastrof. Další z oblastí využití počítačového vidění je dálkový průzkum země (lesnictví, geodézie, meteorologie, archeologie, ...), lékařské aplikace (rozpoznávání rakovinných buněk, texturní analýza myokardu v echokardiografie, ...) a aplikace ve výrobě (počitadlo lahví, obsluha textilních strojů).

10.2 Zpracování přirozeného jazyka

V podstatě existují dva způsoby generování lidské řeči. *První* způsob spočívá v tom, že se vytvoří zvuková databáze předem namluvených slov, která se aktivují a přehrávají v závislosti na tom, jak jsou obsaženy v textu, jenž má být přečten. Tento způsob je dnes používaný zejména u různých slovníků a překladačů. *Druhý* způsob spočívá opět ve vytvoření databáze, která neobsahuje zvukové vzory slov, ale jejich matematické charakteristiky a ty pak při čtení využívá ke generování slov ve vhodném zvukovém generátoru. Na rozdíl od prvního způsobu má výhodu v tom, že není tak náročný na objem dat. Malý objem dat vychází z faktu, že každé slovo je reprezentováno jistou matematickou charakteristikou a ne jeho celým zvukovým záznamem.

Automatická analýza přirozeného jazyka počítačem vyžaduje rozdělit práci na několik menších, dobře definovaných podproblémů, které pak řešíme nezávisle. V oblasti zpracování přirozeného jazyka se mluví o tzv. *rovinách popisu jazyka*. Každá rovina má své vlastní jednotky popisu pro definice vztahů uvnitř roviny a bezprostředně navazuje na

rovinu vyšší (tzn. výstup z nižší roviny je vstup do následující vyšší roviny). Obvykle se hovoří o pěti až šesti rovinách (Hajíčova 1985)

1. *Fonetická rovina:*
   Vstupem do fonetické roviny je akustický signál, výstupem pak posloupnost fónů (zvuků — vektorů různých charakteristik).
2. *Fonologická rovina:*
   Vstupem je posloupnost znaků čili písmen, výstupem pak posloupnost slov a interpunkce. Předzpracování můžeme rozdělit na roviny: pravopis, fonologie, morfonologie.
3. *Morfologická rovina:*
   Vstupem je jak se dá předpokládat posloupnost slov a výstupem dvojice [lemma, značka]. Morfologické analýze se budu podrobněji věnovat v další části.
4. *Syntaktická (povrchová) rovina:*
   Vstupem je posloupnost dvojic [lemma, značka].Výstupem větná struktura (strom) s označením větných vztahů.
5. *Sémantická (tektogramatická, hloubková) rovina:*
   Vstupem je větná struktura (strom) s pojmenováním vztahů.Výstupem je rovněž stromová struktura, ale jsou odstraněna pomocná slova (hloubkové struktura věty).
6. *Pragmatická (logická) rovina:*
   Vstupem je hloubková struktura věty (propozice). Výstupem je logická forma, která může být vyhodnocena (pravda/nepravda).

Někdy je vhodné některé roviny dále rozdělit, nebo naopak sloučit či přeskočit. To závisí na faktorech řešeného problému, např. jaký jazyk zpracováváme; nebo zda se jedná zpracování mluvené řeči; nebo jen překladu z jednoho jazyka do druhého atd.

Textový korpus

Textový korpus se většinou definuje jako rozsáhlý vnitřně strukturovaný a ucelený soubor textů daného jazyka elektronicky uložený a zpracovávaný. Korpusy mohou vznikat pouhým převodem dostupných textů v elektronické podobě, což je velice levné řešení. Každý

rozsáhlejší systém, který má být snadno použitelný, musí vytvářet nějakou abstraktní úroveň s dostatečně jednoduchými prvky. V korpusu by takový základní prvek mohlo být slovo. Texty ovšem obsahují i jiné řetězce znaků: čísla a interpunkce (tečky, čárky, uvozovky apod.). Základním stavebním prvkem by tedy mělo být něco, co může být slovem, číslem nebo nějakým znaménkem. Tento prvek se označuje jako *pozice*. Korpus tedy tvoří posloupnost pozic.

Morfologická analýza

Úkolem morfologické analýzy je určit morfologické kategorie příslušející danému slovu v textu. Při počítačovém zpracování je však situaci třeba definovat a popsat mnohem přesněji. Především je třeba jasně rozlišovat mezi morfologickou kategorií a její hodnotou. V češtině rozlišujeme např. následující: slovní druh, rod, číslo, pád, osobu, čas, slovesný rod apod. Hodnotami kategorie jsou např. čísla 1 až 7 pro české pády, "aktivní" a "pasivní" pro slovesný rod apod. Morfologická analýza pracuje bez ohledu na kontext, tj. zpracovává izolovaně vždy jen jedno slovo (slovní tvar).

Pro počítačové zpracování se zavádí tzv. množina morfologických značek, z nichž každá reprezentuje hodnoty morfologických kategorií pro jeden slovní tvar. Pro vlastní zpracování se používá několik typů notací, z nichž nejrozšířenější je tzv. *poziční notace*, v níž se každé kategorii přiřadí pozice ve značce a každé hodnotě jeden znak, který se zapisuje na příslušnou pozici.

Pro každý slovní tvar se určí všechny možnosti kombinací hodnot morfologických kategorií, které mu přísluší. Počítačová morfologická analýza však musí řešit ještě jeden problém, tzv. problém *lematizace*, jež určuje pro každý slovní tvar jeho základní podobu. Je nutno rozlišovat mezi slovy, která jsou v základním tvaru homonymní - např. stát (jako státní útvar) a stát (jako sloveso). Počítačová lematizace proto ještě navíc tato slova rozlišuje a jednoznačně identifikuje (např. připojením číselného indexu k základnímu tvaru slova, např. stát-1, stát-2 atd.). Morfologická analýza je v počítači realizována jako výpočetní procedura. Základní datovou strukturou pro daný přirozený jazyk je jeho

morfologický slovník, který je používán vlastním algoritmem morfologické analýzy (v zásadě pak již na jazyce nezávislým).

Pro zjednoznačnění výstupu morfologické analýzy se může využít značkování. Dnes se téměř výhradně používají pro značkování metody statistické, založené na strojovém učení (Hajič 2001). Počítač se tedy naučí, že po určitých předložkách následují jen některé pády, že na začátku věty nalezneme nejspíše pád první než jakýkoliv jiný apod. Nyní se naskýtá otázka: Jak se může počítač takovou věc naučit? Potřebuje k tomu předem ručně označkovaný korpus. Takový korpus je samozřejmě velmi pracnou záležitostí; pro spolehlivé naučení, kdy procento chyb klesá (pro češtinu) pod 5%, bylo třeba označkovat přes 1.5 miliónu výskytů slov v textu. Označkované korpusy jsou proto velmi cenným zdrojem lingvistických informací. Učení z ručně označkovaného korpusu (takovému korpusu se říká trénovací data) může probíhat několika způsoby. Velmi jednoduchý a účinný je postup, při kterém se spočítají relativní četnosti značek následujících po dvojici bezprostředně předcházejících značek v textu. Pro každou dvojici značek se tak vytvoří menší či větší tabulka, ve které jsou uvedeny relativní četnosti značek po ní následujících v trénovacích datech. Jakkoli je tento systém lingvisticky jasně neadekvátní, značkování založené na efektivním algoritmu aplikace těchto tabulek dává na kontinuální text velmi dobré výsledky: pro angličtinu se dosahuje i méně než 3% chyb na prakticky libovolném textu, pro češtinu pak okolo 5%.

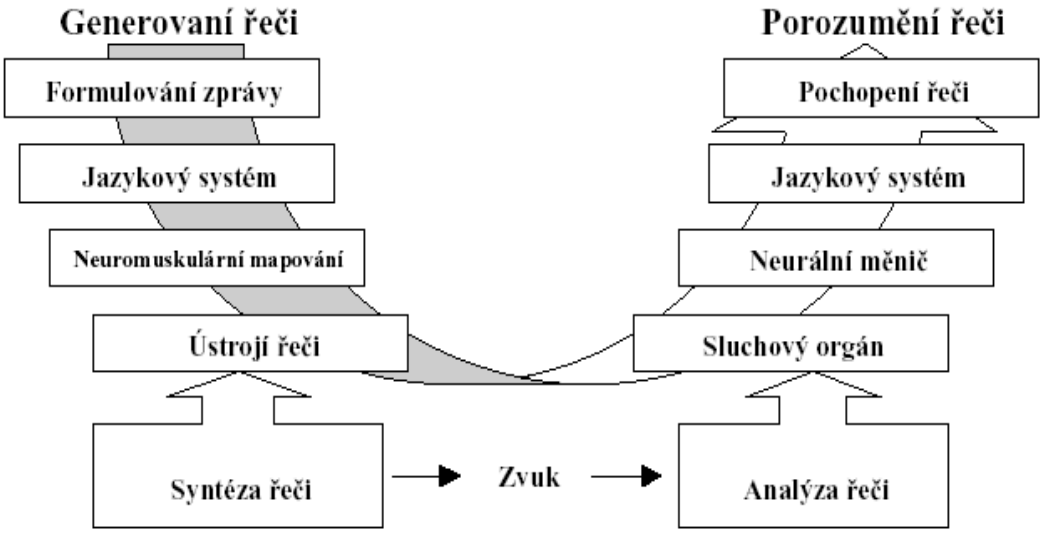
Příklad (Habiballa, 2004):

Tzv. "přímá" analýza slovních tvarů je založena na vyčerpávající analýze slova z hlediska možné segmentace na kmen a koncovku. Příkladem může být slovo (slovní tvar) housky. Toto slovo je možno rozdělit na kmen housky + nulovou koncovku, nebo na *housk* + *y*, nebo na *hous* + *ky*, atd. až k *h* + *ousky* (kmen nulové délky se nepřipouští). Z těchto možností nakonec bude správná jen možnost *hous* + *ky*, neboť ve slovníku je neměnná část základu (zde jen *hous*, neboť 2. p. mn. čísla je *hous+ek*). Koncovky *y*, *sky*, a *nulová koncovka* jsou sice ve

slovníku koncovek uvedeny také, ale kmen *housk* (*hou*) je nepřipouští (resp. nejsou uvedeny v seznamu koncovek pro vzor příslušný danému kmeni).

Rozpoznávání hlasu a řeči

Základní jednotkou popisu řeči jako zvukové formy je elementární zvukový segment. Volba jeho délky ovlivňuje další zpracování řeči. Segment musí být dostatečně krátký, musí být schopen tvořit další jednotky tzv. řetězením. Nutná je znalost vztahu segmentu k jazyku jako lingvistické formě. Při zpracování řeči lze kombinovat segmenty o různých velikostech. Základní řečová jednotka je *foném*, jež označuje minimální fonetickou jednotku identifikující jednotlivé primitivní zvuky. Počet fonémů v existujících v českém jazyce je 36. Fonémy se skládají do větších celků – *slabik*. Libovolná promluva je potom vlastně opakování různých slabik.



Obrázek 10.1: Přenos informací pomocí řeči (Habiballa 2004).

Přenos informací řečí se pro komunikaci člověk-člověk rozděluje na dvě části. První část je *generování* řeči (vytváření), druhá část je *rozpoznání* řeči (porozumění) viz obrázek 10.1. Mezi těmito dvěma celky vystupuje řeč (zvuk) jako médium pro přenos informace (informační kanál).

Hlasový vstup je ovlivněn mnoha faktory. První je řečová složitost vstupu. Na aplikace pro zadávání příkazů a ovládání jsou kladeny nejmenší nároky. Jedná se o přístup, kdy člověk ovládá zařízení pouze jednoduchými (jedno nebo více slovnými) příkazy. Tyto příkazy se rozpoznají ze slovníkové databáze čítající několik desítek slov.Vyšší nároky jsou kladeny na aplikace pro diktát. Tyto aplikace se musejí vyrovnat s plynulou řečí a nezávislosti na mluvčím. Nejvyšší nároky jsou na aplikace pro diktát s integrovaným porozuměním mluvenému jazyku. Další hledisko je prostředí použití. Nejjednodušší je použití v tichém prostředí a pro jednoho mluvčího. Obtížnost stoupá s použitím plynulé řeči, nezávislosti na prostředí a na mluvčím.

Informace obsažená v mluvené řeči je obvykle získávána mikrofonem. Pro další zpracování je zapotřebí tyto analogové kmity převést do číslicových údajů. Tento proces se nazývá *digitalizace* a zahrnuje provedení dvou kroků, a to vzorkování a kvantizaci s kódováním.. Výsledkem je potom číslo, či vektor čísel, který jistým způsobem charakterizuje daný zvuk, který se dalšími metodami vylepšuje, zpracovává a nakonec se realizuje jeho rozpoznání.

10.3 Počítačové vidění

Úlohou počítačového vidění je napodobit pomocí technických prostředků schopnosti lidského oka. Jde nejen o samotnou schopnost zpracování obrazu, ale také o rozpoznání toho, co obraz představuje. V tom hraje podstatnou roli inteligence člověka a jeho předchozí zkušenosti. Počítačové vidění lze rozdělit do dvou oblastí: rozpoznávání 2D obrazu (např. textu a znaků na papíře) a rozpoznávání 3D obrazu (např. rozpoznávání součástky pro její následné uchopení).

Důležitou částí počítačového vidění jsou techniky zpracování 2D obrazů. (Šonka 1992):

1. *Získání digitálního obrazu*:
   Snímání digitálního obrazu je proces, kdy se používají zařízení typu

scanner či videokamera k získání vstupní spojité fyzikální veličiny a jejich převod na spojitý elektrický signál. Digitalizací obrazu rozumíme převod vstupního spojitého signálu na diskrétní tvar. Vstupní analogový signál je popsán funkcí $f(i,j)$ představující souřadnice v obrazu. Funkční hodnota odpovídá jasu (intenzitě). Digitalizace se provádí tak, že se příslušný obraz zaostří na plochu čidel, a ta generují elektrický signál úměrný intenzitě obrazu. Dochází tak k diskrétnímu převodu jak plošnému, tak amplitudovému, tj. signál se vzorkuje a kvantuje. Kvantováním se spojité hodnotě jednoho vzorku přidělí diskrétní hodnota, obvykle jedna z 256 možných jasových úrovní. Po digitalizaci obrazu dostaneme matici přirozených čísel popisujících obraz.

2. *Předzpracování digitálního obrazu:*

   Úpravou digitálního obrazu v podstatě rozumíme použití různých filtrů a opravných mechanizmů k odstranění poruch obrazu (tj. různých šumů přidaných k obrazu před vstupem do kamery nebo během jeho digitalizace). Během tohoto procesu se také zlepšuje kvalita některých částí obrazu, které jsou podstatné pro další zpracování (např. vyostřování hran, potlačení či zvýraznění některých vlastností obrazu pomocí filtrace apod.).

3. *Segmentace obrazu:*

   Segmentace je proces, který umožňuje rozpoznávat jednotlivé objekty. Snažíme se v obrazu nalézt nepřekrývající se oblasti odpovídající objektům (jde například o hledání černých písmen na bílém podkladě). Segmentaci rozdělujeme na úplnou, kdy se zcela povede přiřadit oblasti objektům a částečnou, kdy některé nalezené části neodpovídají objektům. V průběhu tohoto procesu sice dochází k ohraničení objektů, ale také k redukci dat. Nejjednodušší segmentací je tzv. *prahování*, při kterém se vychází ze znalosti odrazivosti povrchů jednotlivých těles a lze ho používat, pokud se hledané objekty odlišují výrazně svým jasem od pozadí. Určuje se tzv. *jasová konstanta* a přes daný filtr se propustí jen ty části obrazu, které mají odpovídající nebo vyšší jas. Existuje mnoho

dalších algoritmů segmentace, výsledkem každého z nich je zvýraznění objektu v daném obraze.

4. *Popis objektů digitálního obrazu:*
   Objekty musí být popsány nějakým vhodným a jednoznačným způsobem, který by umožnil také jejich jednoznačnou klasifikaci. To lze udělat několik způsoby. Jedním je například tzv. *řetězový kód*. Jde o vyjádření pozic jednotlivých pixelů pomocí čísel, které odpovídají pozici daného pixelu, jasu, atd. Existuje mnoho dalších algoritmů výsledkem každého z nich je provést pokud možno takový popis, aby stroj mohl rozpoznat, co je to za objekt.
5. *Klasifikace objektů digitálního obrazu:*
   Vlastní klasifikace objektů digitálního obrazu se provádí tzv. *klasifikátory*, které určují, do jakých tříd patří právě vyšetřovaný objekt. Obecně lze říci, že klasifikace objektů digitálního obrazu znamená interpretaci jeho dat, o kterých se předem nic nepředpokládalo.

Mezi základní úlohy pro využití počítačového vidění patří:

- detekce událostí – jedná se například o vizuální kontrolu výrobků ve výrobním procesu,
- modelování objektů nebo prostředí – průmyslové inspekce, topografické modelování, lékařské obrazové analýzy,
- řízení procesů, systémů – průmyslové roboty, hydraulická ramena, samočinná vozidla a podobně,
- organizace událostí – indexace databází obrázků, obrázkových sekvencí.

Kontrolní otázky a úkoly:

1. Vysvětlete jak je chápán robot v USA a v Japonsku.
2. Jaké je uplatnění robotů?
3. Objasněte princip rozpoznávání hlasu a řeči.
4. Co je to textový korpus?
5. Jak probíhá morfologická analýza?

6. Jaké jsou principy počítačového vidění?
7. Jaké jsou techniky zpracování 2D obrazů?

Korespondenční úkoly

1. Vyberte si jednu oblast použití robotů. Téma zpracujte jako seminární práci v rozsahu nejméně tří stran.
2. Zpracujte historii robotiky v rozsahu v rozsahu nejméně tří stran.
3. Nalezněte konkrétní aplikace zpracování přirozeného jazyka a napište na toto téma pojednání v rozsahu nejméně tří stran.
4. Vyberte si jednu oblast použití počítačového vidění a napište na toto téma pojednání v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Tato kapitola se snaží vnést jasno do chápání pojmu robot tak, aby nešlo jen o formální vnější rysy (pohyb, ruka, oko), ale zejména o styl řešení úloh (záměr, působnost). Dále jsme se věnovali problematice zpracování přirozeného jazyka. K rozpoznávání lidské řeči pomocí počítače lze přistupovat ze dvou rozdílných hledisek. Buď se rozpoznávají hlasy různých řečníků od sebe nebo je nezávisle na řečníkovi zjišťován obsah jeho promluvy. V závěru kapitoly je stručně představeno počítačové vidění. Je to technická disciplína, která využívá kombinaci počítače, snímacího zařízení a softwaru, který je schopný snímanou předlohu zpracovat nebo klasifikovat

Pojmy k zapamatování

- robot, textový korpus,
- morfologická analýza,
- počítačová lematizace,
- segmentace obrazu,
- digitalizace,
- prahování,
- klasifikace.

11 OCR

V této kapitole se dozvíte:

- Jak pracují OCR programy.
- Jaké jsou metody vektorizace a skeletonizace.

Po jejím prostudování byste měli být schopni:

- Vysvětlit princip činnosti OCR.
- Charakterizovat preprocesing a postprocesing v OCR programech.

Klíčová slova této kapitoly:

OCR (Optical Character Recognition), vektorizace, skeletonizace.

Průvodce studiem

OCR (z anglického Optical Character Recognition) je metoda, která pomocí scanneru umožňuje digitalizaci tištěných textů, s nimiž pak lze pracovat jako s normálním počítačovým textem. Počítačový program převádí obraz buď automaticky nebo se musí naučit rozpoznávat znaky. Převedený text je téměř vždy v závislosti na kvalitě předlohy třeba podrobit důkladné korektuře, protože OCR program nerozezná všechna písmena správně.

11.1 Optické rozpoznávání znaků

OCR neboli optické rozpoznávání znaků je technologie převodu textu uloženého v bitmapovém formátu do formátu textového. OCR je speciálním případem vektorizace (Optical Character Recognition), tedy rozpoznávání písma. Text uložený v bitmapě není chápán jako text, je

to jen sada tmavých a světlých bodů v obrázku. OCR program tedy musí identifikovat v bitmapě různé tvary a porovnat je s předlohou a rozhodnout jaké písmenko, ten který shluk představuje. Situace je navíc zkomplikována tím, že texty bývají napsány v různých fontech a dokumenty bývají často nekvalitní. Zvláště xeroxované dokumenty bývají „zašpiněné“, tzn. obsahují rozmazaná písmenka a šmouhy. Program se tedy musí snažit i určit zda tečka poblíž identifikovaného písmenka „c“ je háček a nebo jen nějaké smítko. Novější trasovací programy pracují tak, že dokument procházejí několikrát za sebou a při posledních průchodech už spolupracují se *spell-checkerem*. Mnohé programy se umí „učit“. Takže když chcete převést do textového formátu sadu dokumentů psaných na jednom psacím stroji, můžete OCR program naučit, že dotyčnému stroji ustřelovalo písmenko „z“ a „k“ bylo trochu rozmazané.

Vektorizace

Vektorizací rozumíme převod dat z rastrového formátu do formátu vektorového. Jedná se o úlohu obtížnou, neboť informací uložených v rastrovém formátu je méně, než informací uložených ve formátu vektorovém, a tak je potřeba nové informace automaticky generovat, nebo je ručně do dat doplnit.

Při vektorizaci jsou používány tři základní metody: (Habiballa 2004):
Ruční - obsluhovaná uživatelem. Na zobrazovacím zařízení (monitoru) se zobrazí rastrový obrázek a podle něj uživatel zadává pomocí vstupního digitalizačního zařízení (tablet nebo myš) jednotlivé vektorové entity. Ruční vektorizace je nejpřesnější, ale velmi zdlouhavá a náročná. Zvláštním přikladem ruční vektorizace může být přímá digitalizace papírové předlohy digitizérem bez použití mezistupně rastrového obrázku.

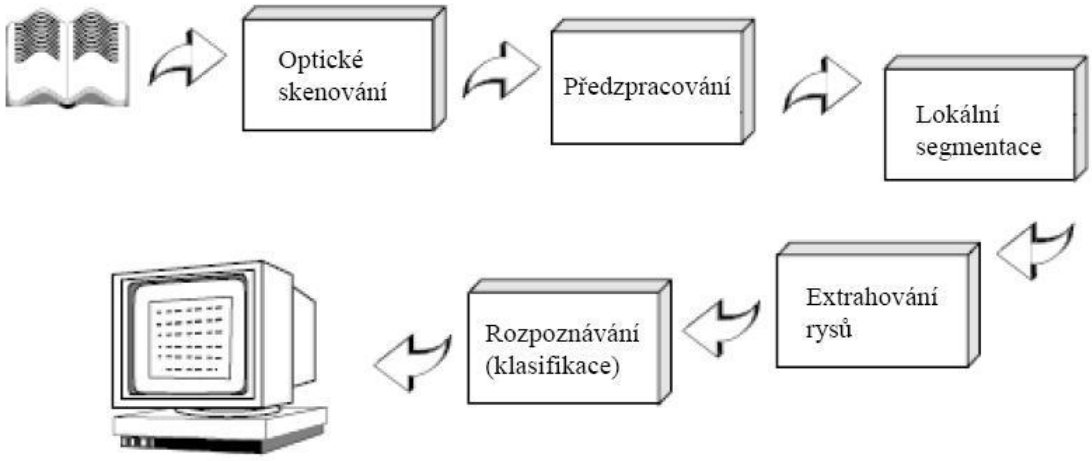
Automatická - obsluhovaná programem. Program musí být schopen na základě informací o barvě jednotlivých pixelů určit základní entity, ze kterých se obraz skládá. V mnoha případech se pro takový postup

používají metody příbuzné umělé inteligenci. Automatickou vektorizaci lze v současné době použít jen u jednoduchých a zřetelných předloh.

Poloautomatická vektorizace je dnes nejčastější. Samotnou vektorizaci provádí program, který je v případě sporných situací korigován a opravován uživatelem. Kvalita vektorizace a její rychlost závisí na stupni automatizace.

11.2 OCR algoritmy

V současné době se rozpoznávání ručně psaných znaků potýká s řadou problémů. Zejména je to odstraňování pozadí, korekce sklonu a velikosti písma, digitální způsob přemýšlení, který se liší od lidského. Některé z těchto problémů se podařilo do jisté míry odstranit tzv. preprocessing and postprocessingem, které se vykonávají před nebo po samotném OCR. Je rozumné oddělovat fáze „předprocesní“ a „poprocesní“ od algoritmů OCR a to z důvodu, že většina algoritmů OCR umí pracovat jen s černými znaky na bílém pozadí. Na obrázku 11.1 je uvedena typická činnost OCR programu.



Obrázek 11.1: Části OCR programu (převzato z http://geo3.fsv.cvut.cz/vyuka/kapr/SP/2008_2009/vymetalek_viktora/index.html)

Preprocessing

Typickým příkladem algoritmů pro předzpracování v OCR je odstraňování vzorů na pozadí, vypreparování textu a srovnání šikmo psaného textu, korekce velikosti a sklonu znaků. Text je vlastně často psán na vzorovaném pozadí (papíře), například při posílání dopisů lidé často používají pěkně graficky zpracované obálky, proto by dobrý systém měl umět rozeznat text od okrasné grafiky. Lidé toto samozřejmě zvládnou díky přirozené intuici, ale podíváme-li se na tento problém z hlediska počítače, je to složité. Není výjimkou, že se celý blok textu rozpozná jako grafika a je tak vyloučen z rozpoznávací procedury.

Jednou z možných metod je tzv. *thresholding*, jedná se o algoritmus na získání prahové úrovně šedi, kdy každému pixelu na pozadí se přiřadí "0" a pixelu na popředí hodnota "1". Takto připravená data mohou přistoupit k dalšímu kroku v preprocesní fázi, kterým je segmentace a následně skeletonizace.

Algoritmus *segmentace* hledá na obrázku sloupcovité skrumáže bílých pixelů, které znamenají oddělení jednotlivých znaků.

Vypreparování textu (znaků), tzv. *skeletonizace*, je velice důležitou úlohou, jelikož takto získaná informace o tvaru znaků je základní vstupní informací pro algoritmus OCR. Skeletonizace může pomoci odstranit nepravidelnosti ve znacích, a tím zjednodušit celý rozpoznávací algoritmus, který pak uvažuje jen znakové úhozy, jež jsou pouhý pixel široké. Rapidně se taktéž sníží paměťové nároky na uchovávané informace o vstupních znacích. Toto je jedna z hojně užívaných metod protože mimo jiné, zkracuje čas samotného rozpoznávání. Kostry (skeletony) se získávají tzv. ztenčovacími metodami nebo transformacemi vzdáleností.

Zjednodušení algoritmu OCR můžeme dosáhnout také tím, že algoritmus bude považovat všechna vstupní data za normalizovaná na standardní rozložení (míněno, že text není "do kopce" ani "z kopce"),

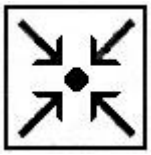
velikost a sklon. Existuje řada úspěšných algoritmů, které umožňují vytvořit horizontální rozložení, otáčet znaky a vytvářet konzistentní velikost. Algoritmy OCR pak mohou dostávat normalizované tvary znaků. Jiným příkladem normalizace je případ, kdy může být znak psán více způsoby.

Postprocessing

Postprocesní zpravování dat je velice důležité, protože slouží k napravování chyb, kterých se případně dopustí algoritmus OCR. Nejtypičtějším případem postprocesního zpracování je kontrola pravopisu (spell checking), která automaticky opraví drobné chybičky a u větších chyb se zeptá uživatele na správný tvar. V mnoha případech se raději přímo přistupovalo k přepisování dokumentů než k použití OCR a následné manuální opravě, protože množství chyb, jichž se OCR systémy dopouštěly, bylo příliš velké. Musíme si uvědomit, že postprocesní systémy by neměly znehodnotit původně správně rozpoznaná data. Poprocesní systém opravující chyby OCR algoritmu, je možné napojit na jeho výstup.

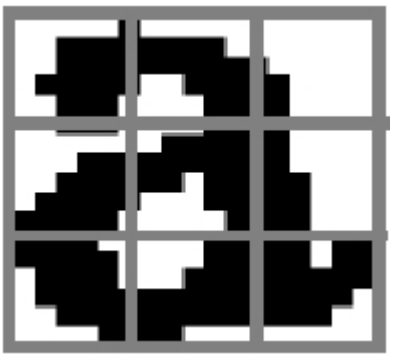
Nejběžněji používané techniky OCR jsou založeny na Markovových modelech, neuronových sítích, srovnávacích metodách tvarů a vzorů (stromové struktury vzorů jsou úspěšné při rozpoznávání netradičních fontů), polygonálních odhadech (vstupní data jsou převedena na polygonální reprezentaci), filtrech na extrakci tahů a bodů (on-line rozpoznávání). Markovovy modely zaznamenaly velký úspěch v rozpoznávání řeči, jsou to vlastně stavové stroje, které využívají kontextové informace. Počítače obecně nemají problémy s rozpoznáváním dobře napsaných znaků, které se moc neliší od daných vzorů, ale psané znaky jsou mnohdy víceznačné a nečitelné i pro člověka. Lidé jsou schopni číst slova s nečitelnými znaky, a tak by tomu mělo být i u počítačů. Algoritmy založené na principu rozpoznávání znaku po znaku by na takovém slově neuspěly, ukazuje se, že kontextová informace je nejen užitečná, ale často i nutná.

Extrakce příznaků



Klasifikace vzorů je nejproblematičtější částí z celého OCR (Suchý 2007). Jejím úkolem je získání základních charakteristik každého znaku. Většina metod se snaží popsat znak přímo ze skenovaného obrázku, jiné zase získávají specifické rysy, které jednotlivé znaky charakterizují. Prvně jmenovaná metoda popsání znaku přímo ze skenovaného obrázku je založená na rozložení bodů v mřížce. Tato metoda má dva zástupce a jsou jimi: rozdělení do pásem a průsečíky.

Rozdělení do pásem. Políčko s lokalizovaným znakem je rozděleno na několik oblastí a zkoumá se histogram tmavých míst v jednotlivých oblastech znaku, jak je vidět na obrázku 11.2. Histogramy se pak porovnávají s rysy jednotlivých znaků, které vzejdou z tzv. trénovacích dat.

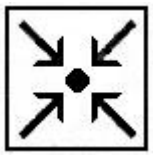


Obrázek 11.2: Rozdělení do pásem.

Průsečíky. Tato metoda je založena na počtu průsečíků předem zvolených vektorů v políčku se znakem. Názorně je to vidět na obrázku 11.3. Metoda rozpoznávající na základě specifických rysů je nazývána strukturální analýzou, kdy jsou jednotlivé znaky popisovány geometrickou a topologickou strukturou znaků. Tato metoda je však ještě předmětem aktivního výzkumu.

Obrázek 11.3: Průsečíky.

Praktické využití OCR



V zásadě existují dva velcí uživatelé systémů na rozpoznávání ručně psaných znaků. Jedná se o sektory *bankovních* a *poštovních* služeb. Finanční instituce využívají těchto systémů zejména v souvislosti se šeky, jelikož šek obsahuje značné množství položek, které je třeba řádně ověřit, aby byla zajištěna bezpečnost bankovních operací. Systém by například mohl kontrolovat, zdali souhlasí částka uvedená číslicemi a částka uvedená slovy, dále by mohl kontrolovat, jestli jméno plátce souhlasí s účtem, ze kterého má být částka odečtena, avšak první věcí, která by nás v souvislosti s šeky napadla, by bylo ověřování podpisu. Když vezmeme v úvahu množství šeků, které projde každý den bankovním systémem, a i kdyby takový systém rozpoznával pouze polovinu z nich, znamenal by značnou úsporu mravenčí práce. Francouzská pošta zavedla počátkem devadesátých let podobný systém, který měl mít 0,01% chybovost a umožňoval, aby až 50% šeků prošlo bez nutnosti předání k ručnímu zpracování.

Již mnoho poštovních úřadů zavedlo systémy OCR, které rozeznávají samostatné číslice, tím napomáhají k automatickému třídění zásilek na základě poštovních směrovacích čísel. Mnoho výzkumů potvrdilo obrovskou výhodu používání výlučně číslic v poštovních směrovacích číslech (USA, Česká republika atd.), jelikož číslice lze relativně nejsnadněji automatizovaně rozpoznávat. Systémy na poštách však

mohou jít mnohem dál. Bylo by je možné používat k rozpoznávání celých adres a třídění obálek jen na jejich základě, tedy bez PSČ.

Jinou oblastí, kde se využívá ručně psaného vstupu dat, je vyplňování formulářů, psaní si poznámek, korektura dokumentů a udržování záznamů. Celkem pěknou aplikací by se mohl stát systém, jež by přijímal data z digitálního pera, např. při přejímaní dodávek zboží, kde by se pak data přenášela do centrálního počítače. To by ale vyžadovalo, aby se ve větší míře rozšířily elektronické podpisy. Když bychom se přenesly ze skladu do kanceláří, pak by nás určitě potěšil systém, jenž by automaticky četl faxové zprávy. S tím souvisí i podpora pro nevidomé, která jednoznačně ukazuje, že rozpoznávání ručně psaných znaků se nevyužívá výhradně pro komerční nebo humanitární aplikace, ale obě úrovně se navzájem překrývají.

Kontrolní otázky a úkoly:

1. Jak pracují OCR programy?
2. Jaké znáte metody vektorizace a skeletonizace?
3. jaké jsou typické oblasti použití OCR programů?

Korespondenční úkoly

1. Vyberte si jednu oblast použití OCR a napište na toto téma pojednání v rozsahu alespoň 5 stran.
2. Napište pojednání v rozsahu alespoň 5 stran na téma „OCR algoritmy“.

Shrnutí obsahu kapitoly

Tato kapitola představuje stručný úvod OCR neboli optického rozpoznávání znaků (z anglického Optical Character Recognition. OCR je metoda, která pomocí scanneru umožňuje digitalizaci tištěných textů,

s nimiž pak lze pracovat jako s normálním počítačovým textem. Nejproblematičtější částí OCR programů je klasifikace vzorů, proto je jejich nezbytnou součástí preprocessing a postprocessing zpracovávaných dat. V závěru kapitoly jsou zmíněny typické oblasti využití OCR.

Pojmy k zapamatování

- OCR (Optical Character Recognition),
- vektorizace,
- skeletonizace.

12 Vybrané aplikace umělé inteligence

V této kapitole se dozvíte:

- Jaké jsou možné oblasti použití umělé inteligence.

Po jejím prostudování byste měli být schopni:

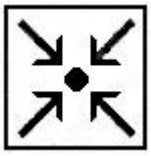
- Vysvětlit kde lze používat umělou inteligenci,
- Objasnit čím je významná počítačová hra Black & White.

Klíčová slova této kapitoly:

Boti, navigační síť strojový překlad.

Průvodce studiem

V této kapitole jsou zmíněny některé možnosti použití umělé inteligence v počítačových hrách a dalších oblastech informatiky. Samozřejmě, že existují i mnohé další oblasti použití umělé inteligence.



V této kapitole jsou zmíněny některé možnosti použití umělé inteligence.

Vyhledávání na internetu pomocí hlasu

Určitě to znáte: když potřebujete na internetu něco najít, spustíte internetový vyhledávač Google, do příslušného políčka napíšete odpovídající výraz, chvíli počkáte a nakonec projdete výsledky vyhledávání. O co lepší by však bylo, kdyby za vás počítač udělal i již zmíněné prohledávání ve výsledcích hledání, a tak vám připravil skutečně pouze to, co vás zajímá. Přesně na výše zmíněném úkolu pracují vědci a studenti ve výzkumné oblasti vytváření inteligentního rozhraní pro uživatele v německém výzkumném centru pro umělou

inteligenci (DFKI). Jejich projekt se jmenuje Smart Web (http://www.smartweb-projekt.de). Jeho cílem je vytvořit mobilního průvodce, který je bezdrátově připojen k internetu a který uživateli prostřednictvím hlasového nebo textového výstupu automaticky podává informace. Na otázku typu „Kdo byl William Shakespeare?“, kterou uživatel položí, přístroj odpoví (a to rovněž hlasem) „Významný anglický básník a dramatik“. Uživatel tak vůbec nemusí prohledávat žádné výsledky vyhledávání.

Vyhledávání obrázků podle motivu

Program Imagesorter 2.0.1 třídí obrázky na pevném disku podle jejich podobnosti, v závislosti na obsahu. Nástroj je zdarma a najdete jej, popřípadě na internetové adrese http://mmk.f4.fhtw-berlin.de jako soubor IMAGESORTERV2_XP.ZIP o velikosti 7,80 MB. Další funkcí programu je možnost označit obrázek a zobrazit všechny, co mu jsou podobné. Tímto způsobem se tedy dají velmi snadno najít kupříkladu všechny obrázky Eiffelovy věže, které máte na svém počítači. Utilita pracuje tak, že provádí analýzu obsahu obrázku. Prostřednictvím matematických algoritmů popisuje fotografie jako vektory, popřípadě body ve vícerozměrném vektorovém prostoru.

Strojový překlad

Mezi počítačové programy, které se pokouší napodobovat lidskou řeč, patří aplikace pro strojový překlad, rozpoznávání řeči a hlasový výstup. Čím více se v daném programu skrývá inteligence, tím lepší jsou logicky výsledky, které jsou jako výstup poskytovány uživateli. Jedním z těchto algoritmů je k patentování připravený neuronální přenos, jenž se uplatňuje v programu Linguatec Personal Translator 2008. Tento program pro překlad textů si můžete za 49 eur zakoupit na internetové adrese http://www.linguatec.net. Tato metoda dokáže podle kontextu správně přeložit slova s více významy. Správný význam slova rozpoznává program i na základě dalších vět. Pokud si sami chcete udělat obrázek o tom, jak výkonná je aplikace Personal Translator 2008 od firmy Linguatec a kde asi leží hranice překladu vytvořeného pomocí

programu využívajícího umělou inteligenci, navštivte internetovou stránku www.liguatec.net/onlineservices/pt. Zde můžete zadat pro ukázkový překlad až 500 znaků dlouhý text, který lze přeložit do několika jazyků.

Rozpoznávání obličeje

Rozpoznávání obličeje se dá využít dvěma způsoby: buď pro identifikaci určité osoby na obrázku (obličej této osoby však musí být uložen v databázi), nebo pro interpretaci mimiky osob. Identifikace osoby podle obličeje řeší například program X-Login 1.0 vám umožní pracovat na počítači pouze tehdy, pokud před ním bude sedět určitá dříve zadaná osoba. Obraz se do programu snímá obyčejnou webovou kamerou. Programy tohoto druhu mají sklon pracovat spíše jednodušeji, neboť si pamatují pouze umístění osoby a vzdálenost partií obličeje. Ochrana počítače tímto softwarem se tedy dá svým způsobem obejít tak, že použijete fotografii oprávněné osoby, kterou umístíte před webovou kameru.

Rozpoznávání pocitů

Jedná se o program Real Time Face Detector 4.05, který najdete jako 60denní demoverzi , popřípadě ji můžete stáhnout na internetové adrese http://www.iis.fraunhofer.de/bf/bv/kognitiv/biom/dd.jsp jako soubor RTFACEDETECT_SETUP_V405_TCM97-78258.EXE o velikosti 7 MB. I některé digitální fotoaparáty dokáží rozpoznat lidské obličeje a zaměřit na ně objektiv. Forma Sony jde se svými modely DSC-T70 a DSC-T200 dokonce ještě dále. V jejich případě můžete nastavit, aby přístroj pořídil snímek až tehdy, když na tváři fotografované osoby objeví úsměv.

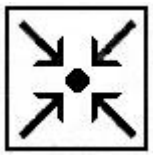
Herní umělá inteligence

Herní umělá inteligence je v současné době jedním z nejbouřlivěji se rozvíjejících odvětví interaktivního zábavního průmyslu. Až do druhé poloviny 90. let byla umělá inteligence v počítačových hrách považována za okrajovou záležitost; herní vývojáři měli v tu dobu jinou

prioritu – dokonalejší a realističtější grafiku. To se projevilo nejen ve velmi malém podílu na výkonu procesoru, který měla herní umělá inteligence k dispozici, ale také ve spěchu a nekoncepčnosti, se kterými byla do her na poslední chvíli implementována.

Situace se začala měnit až s rozmachem grafických akcelerátorů na sklonku 90. let, jejichž nasazení výrazně odlehčilo hlavnímu procesoru, a uvolnilo tak výkon pro algoritmy umělé inteligence. Herní vývojáři si pomalu začali uvědomovat, že atraktivitu hry už nelze stavět pouze na realistické grafice, a zaměřili svou pozornost i na umělou inteligenci. Dnes v sobě herní umělá inteligence zahrnuje herní programování, herní design, ale také psychologii, drama a samozřejmě akademickou umělou inteligenci. Od té se ale výrazně odlišuje už samotným cílem – herní umělou inteligenci zajímá skutečná inteligence totiž jen do té míry, do které přispívá k tomu, aby hra hráče více (a případně déle) bavila. Proto se herní umělá inteligence někdy označuje také termínem zábavní inteligence (entertainment intelligence) - musí být realistická a nesmí obsahovat prvky, které by logice herního světa odporovaly.

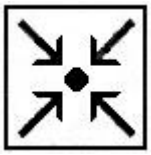
Navigační síť



Každý hráč 3D akčních her typu Quake se střetnul s botem, tedy inteligentním agentem nahrazujícím ve hře lidského protihráče. Ať už v roli protivníka nebo spoluhráče si dnešní boti vedou v boji velmi dobře, přestože zatím většinou nebývá problém odlišit je od živého hráče. Základním problémem, který musí každý bot řešit, je navigace a pohyb v herním prostředí. Bot si za tímto účelem musí nejen pamatovat pozici významných prvků v herní mapě (protivník, zbraň/střelivo, vlajka nepřítele apod.), ale musí být i kdykoliv schopen najít do vybraného místa pokud možno nejkratší cestu. O to se stará tzv. *pathfinding*, dnes v naprosté většině her založený na algoritmu heuristického prohledávání grafu. Pro jeho efektivní nasazení je potřeba k dané úrovni nejdříve zkonstruovat tzv. navigační síť, tedy graf, který popisuje odkud kam je možno se v dané úrovní dostat. Kromě informací týkající se pohybu může navigační síť obsahovat i speciální značky označující místa výhodná pro obranu či naopak pro číhanou na protivníka. V

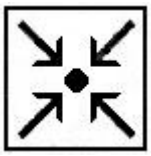
poslední době se proto objevuje snaha tvořit navigační síť automaticky pomocí nástrojů umělé inteligence, což však představuje nelehký úkol.

Řídicí systém



Zatímco navigaci lze považovat za podpůrný systém, skutečným mozkem bota je jeho řídicí systém. Na základě aktuálního stavu bota, jeho záměrů a situace v nejbližším okolí volí akce, které bot v příštích okamžicích provede. Tyto akce lze většinou rozdělit do několika úrovní – hovoříme proto o hierarchickém řídicím systému. Na nejvyšší úrovni jsou akce, které se váží k dlouhodobějším a strategičtějším rozhodnutím – bot si (ve hře určitého typu) může vybrat, zda bude prozkoumávat aktuální úroveň, doplňovat střelivo, útočit či naopak ustupovat z boje. O úroveň níž bot rozhoduje např. o tom, kterým směrem se v mapě při svém průzkumu vydá, na jaký konkrétní cíl zaútočí a jakou přitom použije zbraň. Nejnižší úroveň se pak týká konkrétních atomických úkonů, tj. např. kdy vystřelit, kam zamířit, kudy uhnout střele protivníka apod. Zdaleka nejčastějším způsobem implementace řídicího systému je určitá obdoba tzv. hierarchického stavového automatu, pravidlové systémy či cílově orientované dynamické plánování.

Black & White



Jestliže s herními boty se hráč setká velmi často, umělá inteligence ve hře Black & White (Lionhead Studios, 2001) zůstává dodnes výjimečným počinem. Black&White je od základů postavena na schopnosti herních postav učit se, a je proto často považována z pohledu umělé inteligence za nejinovativnější hru vůbec. Každému hráči je zde přidělen tzv. titán, prostřednictvím kterého hru hraje. Největším lákadlem Black&White je možnost titánovu osobnost zásadním způsobem formovat. Kromě přímých rozkazů hráče a jeho zpětné vazby (trest nebo odměna), se titán dokáže zdokonalovat i z pozorování ostatních postav herního světa a z vlastního průzkumu prostředí. Základem titánovy tvárnosti je jeho schopnost *učit se*. Vzhledem k různým způsobům učení v Black & White zvolili autoři i

několik různých modelů *reprezentace znalostí*. Fakta o jednotlivých objektech jsou reprezentovány jednoduchým seznamem atributů. Znalost o tom, jak jsou které předměty vhodné k uspokojování titánových potřeb, je reprezentována pomocí rozhodovacích a regresních stromů. Závislost mezi stavem agenta a jeho přáními je pak reprezentována pomocí *umělé neuronové sítě*. Lze tak mít titána, který začne hledat kořist, až když má opravdu velký hlad, a naopak titána, který žere pořád, obzvlášť je-li v depresi. Zásadní je, že tyto reprezentace jsou vytvářeny a dále modifikovány dynamicky až během hry pomocí *algoritmů strojového učení*. Black & White lze považovat za ukázku, jak může vhodné použití technik umělé inteligence vést nejen ke kvalitativnímu skoku v herní umělé inteligenci.

Kontrolní otázky:

1. Jaké jsou možnosti použití umělé inteligence?

Korespondenční úkol:

Nalezněte další možné oblasti použití umělé inteligence a napište na toto téma pojednání v rozsahu nejméně tří stran.

Shrnutí obsahu kapitoly

Obsahem této kapitoly byly příklady možného umělé inteligence.

Pojmy k zapamatování

- boti,
- navigační síť,
- strojový překlad.

Literatura

Aldeman, L. Molecular computationof solutionsto combinatorial problems. Science 266 (1994) 1021-102.

Bonabeau E.W., Theraulaz G. „Why we do Need Artificel Life“. In Langton Ch. (ed.) Artificel Life (An Overview). MIT Press, Cambridge , Massachusetts 1995. pp. 303-325.

Buc M. a kol. Imunológia. Skriptá LF Univerzita Komenského, Bratislava 1998.

Codd E.F. Cellular Automata. Academic Press, New York 1968.

de Castro, L.N. - Von Zuben, F.J. Artificial Immune Systems: Part II. - A Survey of Applications, Technical Report - RT DCA 02/00, 2000.

de Castro, L. N., Timmis, J. Artificial Immune Systems: A New Computational Intelligence Approach, Springer-Verlag 2002.

Dvořák, J. Expertní systémy. Ústav automatizace a informatiky, 2004. 92 s. Skripta. Vysoké učení technické v Brně.

Gardner, M. „The fantastic combination of John Conway’s new solitaire game life“. Scientific American 223 (1970), 120 -123.

Habiballa, H. Umělá inteligence. Texty pro distanční studium. Ostravská univerzita. Ostrava 2004.

Hajič, J., Statistické modelování a automatická analýza přirozeného jazyka (morfologie, syntax, překlad). Jarošová, A. (ed.) Slovenčina a čeština v počítačovom spracovaní.VEDA, vydavateľstvo SAV, Bratislava. 2001.

Hajíčová, E., Sgall, P., Towards an automatic identification of topic and focus, ACL Proceedings, Second European Conference,s.263-7, 1985.

Houser P. Paradoxy umělé inteligence: Turingův test 50 let poté, Computerworld 11/2003.

Kephart, J.O. A Biologically Inspired Immune System for Computers. In R.A. Brooks, P. Maes (eds.) Artificial Life IV Proc. of the Fourth

International Workshop on the Synthesis and Simulation of Living Systems, MIT Press, 1994, pp. 130-139.

Kim, J., Bentley, P. The Human Immune System and Network Intrusion System. Proc. of the EUFIT'99, 1999.

Kubík, A., Agenty a multiagentové systémy, Slezská univerzita v Opavě, 2000.

Kukal, J.: Dimenze a míry. Automatizace, ročník 52, číslo 5 (2009) pp. 292-293. ISSN 0005-125X.

Kvasnička, V., Pospíchal, J., Tiňo, P. Evolučné algoritmy. STU, Bratislava 2000.

Mandelbrot, B. B. Fraktály: Tvar, náhoda a dimenze. Mladá Fronta, 2003. ISBN 80-204-1009-0.

Mařík, V., Štěpánková, O., Lažanský, J. a kol. Umělá inteligence (3). Academia, Praha 2000.

Mataric, M., Interaction and Intelligent Behavior, MIT PhD thesis, MIT Press, Cambridge, Mass., 1994

Netrvalová, A. Úvod do problematiky multiagentních systémů. 2004. [Cit. 2013-07-16].
Dostupné online http://www.kiv.zcu.cz/~netrvalo/phd/MAS.pdf.

Neumann, J. Theory of self-reproducing automata. University of Illinois Press, London 1966.

Pokorný, M. Expertní systémy. Ostravská univerzita v Ostravě, 2004. 103 s. Skripta. Ostravská univerzita v Ostravě.

Šonka, M. a Hlaváč, V. Počítačové vidění. Praha: Grada a.s., 1992. ISBN 08-85424-67-3.

Timmis, J. Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory. Ph.D. Dissertation, Department of Computer Science, University of Whales, 2000.

Toma, N., Endo, S., Yamada, K. Immune Algorithm with Immune Network and MHC for Adaptive Problem Solving. In Proc. of the IEEE System, Man, and Cybernetics, IV, 1999, pp. 271-276

Wolfram S. Universality and complexity in cellular automata. Physica, 10D (1984) 1-35.

Zelinka I., Čandík M. Fraktální geometrie – principy a aplikace. BEN – technická literatura, Praha 2006.

Zelinka I. Řízení deterministického chaosu, Automatizace, 5, 2003, 310-315., ISSN 0005-125X.

Zelinka I. Umělá inteligence. Hrozba nebo naděje? BEN – technická literatura, Praha 2003.