Slovenská Technická Univerzita v Bratislave

Fakulta informatiky a informačných technológií

Dopredný produkčný systém

Zadanie 4

**Predmet:** Umelá inteligencia

**Obdobie:** Letný semester 2019/2020

**Cvičiaci:** Ing. Ivan Kapustík

**Študent:** Bc. František Gič

# Obsah

[Obsah 2](#_Toc40635420)

[Zadanie 3](#_Toc40635421)

[Implementácia 4](#_Toc40635422)

[Inštalácia 4](#_Toc40635423)

[Reprezentácia údajov 5](#_Toc40635424)

[Algoritmus 7](#_Toc40635425)

[Testovanie 8](#_Toc40635426)

[Zhodnotenie 9](#_Toc40635427)

# Zadanie

Úlohou je vytvoriť jednoduchý dopredný produkčný systém, s prípadnými rozšíreniami, napríklad o kladenie otázok používateľovi alebo vyhodnocovanie matematických výrazov.

Produkčný systém patrí medzi znalostné systémy, teda medzi systémy, ktoré so svojimi údajmi narábajú ako so znalosťami. Znalosti vyjadrujú nielen informácie o nejakom objekte, ale aj súvislosti medzi objektami, vlastnosti zvolených problémov a spôsoby hľadania ich riešenia. Znalostný systém je teda v najjednoduchšom prípade dvojica – program, ktorý dokáže všeobecne manipulovať so znalosťami a báza znalostí, ktorá opisuje problém a vzťahy, ktoré tam platia. Znalosti majú definovanú nejakú štruktúru a spôsob narábania s touto štruktúrou – to sa nazýva formalizmus reprezentácie znalostí. Program vie pracovať s týmto formalizmom, ale nesmie byť závislý od toho, aké konkrétne znalosti spracováva, inak by to už nebol systém, kde riešenie úlohy je dané použitými údajmi.

# Implementácia

Riešenie zadania som vypracoval v jazyku *JavaScript* , lokálnom environmente - *Node.js* s použitím supersetu *Typescript* pre striktné otypovanie.

# Inštalácia

Prerekvizity:

* Node.js (<https://www.nodejs.org/>)
* Node package manager (<https://www.npmjs.com/>)

V root adresári spustite nasledovné príkazy:

npm install

A následne, pre každú transpiláciu typescriptového kódu na javascript a spustenie kódu v node.js:

npm run dev

# Reprezentácia údajov

Vstupné dáta načítavam vo formáte JSON. Sú to primitívne dátové typy, reťazce (string). Pravidlá produkčného systému sú pre zjednodušenie uložené v objekte, aby som nemusel riešiť základné problémy ako koniec riadku alebo nejaké oddelovače.

A screenshot of a video game

Description automatically generated

Fakty sa následne sparsujú do objektu typu Relationship. Tento objekt obsahuje dáta (aktérov daného vzťahu) a ich vzťah v reťazci – rodič, muž atď.

A screenshot of a cell phone

Description automatically generated

Pravidlá, tie sa sparsujú na objekt typu Rule. Ten obsahuje názov (name), if – akcie ktoré su podmienkami – je to pole typu Relationship a then, čo je pole typu Action – názov akcie (pridaj, vymaž, správa) a vzťah z ktorým sa pracuje.

A picture containing monitor, black, screen, television

Description automatically generated

# Algoritmus

Samotný hlavné telo programu je pomerne jednoduché. V každej iterácií cyklu prechádzame všetky pravidlá. Ku každému pravidlu získame Bindingy – možné aplikovateľné pravidlá ktoré spĺňajú jednotlivé podmienky – pole výsledkov z prvej podmienky, druhé pole z druhej a pod... Ak je teda podmienka, že X je rodič Y, vráti to pole vzťahov – napr Peter -> Jano, lebo ich vzťah je typu rodič.

Následne sa tieto bindings posielajú do funkcie getRuleMatches ktorá tieto polia vyhodnotí, nájde zhodu medzi nimi a vráti pole výsledkov.

Následne cez tieto výsledky iterujeme a vykonávame akcie ktoré sú definované v then daného pravidla. Napríklad, ak je tam pridaj, do stacku sa pridá nové pravidlo (pokiaľ už neexistuje v stacku alebo faktoch).

Na koniec sa do faktov (pracovnej pamäte) priradí prvé pravidlo zo stacku (pomocného výstupu).

Toto sa opakuje pokiaľ v pomocnom výstupe (stacku) existujú nejaké elementy.

A screenshot of a cell phone

Description automatically generated

# Testovanie

Riešenie som testoval na príkladovom zadaní, ktorého pravidlá a fakty som vložil s načítal zo súboru.

Výstup obsahoval dve hlášky, podľa pravidiel, a taktiež výpis pracovnej pamäte tak ako v príkladovom projekte na webstránke.

A picture containing black, man, holding

Description automatically generated

# Zhodnotenie

Riešenie som vypracoval a otestoval na vzorovom a považujem ho za správne.

Najväčším problémom nad ktorým som strávil veľa času bola funkcia getRuleMatches, ktorá dostane ako parameter pole polí ktoré spĺňajú jednotlivé podmienky daného pravidla a výstupom by malo byť pole kombinácií (X,Y,Z) ktoré jednotlivé podmienky spĺňa. Pôvodne som daný problém chcel riešiť rekurzívne, ale použil som možnosť spôsob, kde dvojdimenzionálne pole iterujem v kombináciách po dvoch. Vezmem spoločný stĺpec daných dvoch polí a spojím ich podľa kľúča v danom stĺpci. Potom s každým nasledujúcim pravidlom už len robím prienik množín pravidiel.

Tieto výsledky sa potom vrátia do hlavnej časti programu ako trojice X Y Z s príslušnými hodnotami a pokračuje sa vyhodnotením funkcie then.

Taktiež som mal problémy s neustálym výpisom správ z funkcie sprava. Problém som vyriešil evidovaním už vypísaných správ a pred výpisom správy kontrolujem či sa daná správa už neeviduje (čiže nebola už vypísaná predtým).