Rīgas 64. vidusskola

**Mākslīgā intelekta problēmu risināšana,**

**izmantojot klasisko meklēšanu**

Zinātniski pētnieciskais darbs datorzinātnēs

**Darba autori**:

Rīgas 64. vidusskolas 12. klases skolēni

Kristiāns Magons, Daniels Muļukins

**Darba vadītājs**:

Vadošais arhitekts, SAP SE

Krists Magons

**Darba konsultants**:

Programmēšanas skolotājs, Rīgas 64. vidusskola

Eduards Bukovskis

Rīga, 2025

**Ievads**

Pēdējo gadu laikā ir strauji attīstījušās datortehnoloģijas [1], kas līdzās sekmējušas arī mākslīgā intelekta attīstību, jo radās iespējas efektīvāk glabāt daudz datus mākoņos [2][3] un jaudīgās videokartes [4] ļāva ātrāk veikt skaitļošanu un apstrādāt vairāk datus. Šīs iespējas ļāva realizēt tādas tehnoloģijas kā TensorFlow, kas vienlaikus paralēli izmanto vairākus procesorus, kas būtiski paātrina matemātiskus aprēķinus [5]. Šādas skaitļošanas platformas brīvi pieejamas, pateicoties daudzkodolu videokaršu attīstībai. Pagaidām mākslīgais intelekts nav stingri definēts, jo atšķiras izpratnes par to. Vēsturiski ir nonākts pie četrām pamata definīcijām, kā izprast mākslīgā intelekta jēdzienu – domāšana vai rīkošanās cilvēciski, domāšana vai rīkošanās racionāli [6]. Mūsu pieeja būs racionāli rīkojošs, inteliģents aģents. Mēs izstrādāsim un izpētīsim putekļsūcēja modeli, kura mērķis, kā jau rīkojoties racionāli, ir visefektīvāk nokļūt no sākuma stāvokļa līdz rezultāta stāvoklim, proti, nonākt stāvoklī, kurā visas istabas ir tīras.

Mākslīgā intelekta rīki cilvēkiem jau ir kļuvuši par būtisku palīgu ikdienā, taču prognozes par tā tālāko nākotni ir dažādas. Parasti nepilnības rodas, kad spriedumus mēģina veikt, nevienojoties par mākslīgā intelekta definīciju, jo, kā jau tika minēts, mākslīgā intelekta sfēra ir ļoti plaša ar dažādām definīcijām. Sākot no šaha spēles risinājumiem līdz robotam, kas domā kā cilvēks. Mākslīgā intelekta problemātika ir pētīta jau vairākus desmitus gadus, un šī sfēra ir piedzīvojusi gan lielus uzplaukumus, gan arī kritumus [7]. Tāpēc mēs vēlamies apskatīt problēmjautājumu pamatus, lai gūtu skaidrību par mākslīgā intelekta tehnoloģiju iespējām un gūtu priekšstatu par tā limitācijām.

**Pētījuma mērķi**:

1. Iepazīties ar zema līmeņa programmēšanu un datu struktūrām.
2. Konstruēt racionālu aģentu, putekļsūcēja prototipu, kas darbojas mākslīgā, stingri definētā vidē un izmanto tā sensorus, lai veiktu stāvokļa pārejas, maksimizējot rezultātu (īsākais ceļš no sākuma stāvokļa līdz rezultāta stāvoklim, kad visas istabas ir iztīrītas).
3. Realizēt aģenta stāvokļu pārejas programmu, kā klasisku grafa meklēšanas algoritmu un novērtēt aģenta efektivitāti atkarībā no tā darba vides sarežģītības (istabu skaita).

**Darba uzdevumi**:

1. Valodā C++ realizēt zema līmeņa datu struktūras un pamatalgoritmus (saistītais saraksts, prioritāšu rinda, steks, grafs), kā arī iepazīties ar teksta failu apstrādi;
2. Iepazīties ar programmatūras versiju pārvaldības sistēmu GIT;
3. Izmantojot realizētās datu struktūras un OOP (Objektorientētā programmēšana) paradigmas, realizēt aģenta un vides implementāciju un darbības simulāciju;
4. Realizēt aģenta stāvokļa pārejas funkciju, izmantojot A\* vai citus meklēšanas algoritmus;
5. Novērtēt aģenta veiktspēju atkarībā no vides iestatījumiem;
6. Noskaidrot aģenta potenciālās vajadzības pēc atmiņas daudzuma un izšķirtspējām;
7. Izvērtēt mākslīgā intelekta potenciālās spējas nākotnē.

**Hipotēze**:

Atrast teorētisku risinājumu mākslīgā intelekta problēmai nenozīmē, ka to ir viegli realizēt praktiski.

**Darbā izmantotās metodes**:

Programma tiks vairākkārt palaista, mainot vides iestatījumus, aģenta īpašības un tā skaitu. Iegūtie rezultāti tiks analizēti (laiks, atmiņas patēriņš, utt.)

1. **Ieskats mākslīgajā intelektā**

1.1 **Kas ir mākslīgais intelekts?**

Tūkstošiem gadu cilvēki ir mēģinājuši saprast, – kā cilvēks spēj uztvert, saprast, paredzēt, un ietekmēt pasauli, kas ir daudz plašāka par viņu? Mākslīgā intelekta sfēra tiecas vēl augstāk – tā nav tikai mēģinājums izprast, bet arī veidot inteliģentas būtnes [8]. Vadoties pēc Stjuarta Rasela un Pītera Norviga mācību grāmatas “Artificial Intelligence: A Modern Approach”, mākslīgajam intelektam nav vienotas definīcijas. Izmantojot astoņas dažādas mākslīgā intelekta definīcijas [9], autori tās ir apvienojuši četrās galvenās kategorijās:

**Cilvēciska rīcība** [10]

1950. gadā Alans Tjūrings izgudroja testu, kas vēl līdz šai dienai ir aktuāls, definējot mākslīgo intelektu, un ir plaši zināms, kā “Tjūringa tests”. Šis tests iekļauj pratinātāju, kurš aizklāti intervē datoru un cilvēku. Ja pratinātājs nevar noteikt, vai atbildes sniedz dators vai cilvēks, tad dators ir nokārtojis testu. Tjūringa tests speciāli neiekļauj fizisko saskarsmi starp pratinātāju un datoru, jo fiziskā cilvēka attēlojums nav nepieciešams inteliģencei.

**Cilvēciska domāšana** [11]

Lai mākslīgais intelekts spētu domāt cilvēciski, vispirms ir jāsaprot, ko tas nozīmē – domāt cilvēciski? Ir trīs veidi, kā to noteikt. Pirmais veids ir pašpārbaude – mēģināt noķert savas domas. Otrais ir psiholoģiski eksperimenti – cilvēka rīcības novērošana. Trešais ir smadzeņu attēlveidošana – smadzeņu rīcības novērošana. Šī pieeja mākslīgajam intelektam cieši saistās ar kognitīvo zinātni, jo, lai konstruētu cilvēciski domājošu mākslīgā intelekta moduli, jāzina precīzas un pārbaudāmas teorijas par cilvēka prātu. Tiekot pie pietiekami precīzas teorijas par prātu, ir iespējams to realizēt, kā datora programmu. Ja programmas ievade-izvade saskan ar cilvēka rīcību, tad tiek iegūts reāls pierādījums, ka šis programmas mehānisms varētu darboties arī cilvēkā.

**Racionāla domāšana** [12]

Grieķu filozofs Aristotelis bija viens no pirmajiem, kurš kodificēja “pareizu domāšanu”, proti, “neapstrīdamu spriešanas procesu”. Viņa siloģismi sniedza modeļus argumentu struktūrām, kas vienmēr sniedza korektus secinājumus pie pareiziem pieņēmumiem. Piemērs: “Sokrāts ir cilvēks. Visi cilvēki ir mirstīgi. Tātad Sokrāts ir mirstīgs.” Šie likumi domāšanā regulēja prāta darbību, un viņa pētījumi aizsāka loģikas jomu.

Līdz 1965. gadam pastāvēja programmas, kuras spēja atrisināt jebkādu atrisināmu problēmu, paskaidrotu ar loģiskiem apzīmējumiem (ja atrisinājuma nebija, tad programma ieciklējās mūžīgā ciklā). Tā sauktā loģikas tradīcija mākslīgā intelekta jomā cer balstīties uz šādām programmām, lai radītu inteliģentas sistēmas.

**Racionāla rīcība** [13]

Aģents ir kaut kas, kas rīkojas. Protams, visas datora programmas kaut ko dara, bet datora aģentam ir jāspēj vairāk – rīkoties autonomi, uztvert apkārtējo vidi sev apkārt, pastāvēt ilgstošu laiku, adaptēties izmaiņām, veidot un sasniegt mērķus. Racionāls aģents ir tāds, kas rīkojas, lai sasniegtu labāko rezultātu, bet, ja ir neskaidrības, labāko sagaidāmo rezultātu.

1.2 **Izcelsme**

Mākslīgais intelekts ir viena no jaunākajām zinātnes un inženierijas nozarēm [14]. Runājot par mākslīgā intelekta tēmu, vārdam “nozare” ir jāliek īpašs uzsvars, jo nereti mākslīgais intelekts tiek pārprasts par atsevišķu, patstāvīgu zinātni. Mākslīgo intelektu saista vairākas zinātnes, piemēram, matemātika, psiholoģija, datortehnika un pat filozofija [15]. Ievērojamie rezultāti mākslīgā intelekta jomā gūti vien pēdējo gadu laikā (par mākslīgā intelekta uzplaukumu lasīt nodaļā 1.3 Uzplaukums), taču tas ir bijis garš process vairāku desmitu gadu garumā. Mēģinājumi veiksmīgi izveidot mākslīgā intelekta modeli ir bijuši gan ar lielām cerībām, gan vilšanām.

Pirmie atzītie darbi mākslīgā intelekta jomā datējami 1943. gadā, kad, izmantojot primitīvu psiholoģiju, prepozīciju loģiku un Tjūringa skaitļošanas teoriju, tika izgudrots pirmais modelis, kas reprezentēja mākslīgus neironus. 1950. gadā, tika izveidots pirmais neironu tīkla dators. Lai gan tas simulēja vien 40 neironus, tas sastāvēja no 3000 vakuuma caurulēm un tehnoloģijām, kas tika aizgūtas no bumbvedēja B-24 automātiskā pilota mehānisma pārpalikumiem. [16]

Ietekmīgs veikums 1956. gada vasarā bija Allena Nevela un Herberta Saimona izveidotā spriešanas programma “Loģikas teorētiķis”, kas bija spējīga domāt neskaitliski. [17] Drīz pēc tam viņi izveidoja arī programmu “Vispārējais problēmu risinātājs”, kas bija programma, spējīga atdarināt cilvēcisku domāšanu. Savukārt 1958. gads ir nozīmīgs ar Džona Makartija izveidoto teorētisko aprakstu programmai “Padomu ņēmējs”, kas hipotētiski spēja izmantot vispārējas zināšanas par pasauli, lai meklētu problēmu risinājumus, un ko varēja uzskatīt par pirmo pilnīgo mākslīgā intelekta sistēmu. [18]

Kad tik salīdzinoši īsā laika posmā tika izveidotas mašīnas, kas spēja domāt, mācīties un veidot – cerības par tuvākajiem nākotnes plāniem bija spožas. Ārpus jaunu metožu mēģinājumiem [19], mākslīgā intelekta joma uz ilgāku laiku stagnēja, jo viss atdūrās pret datoru nespēju veikt grūtākus vai apjomīgākus uzdevumus. [20]

1.3 **Uzplaukums**

Mākslīgā intelekta 60 gadu attīstības laikā akcents lielākoties ir bijis algoritmu izvēlei un atbilstībai, taču pēdējā gadsimta laikā lielāka uzmanība tikta vērsta datiem, un to milzīgajai pieejamībai dažādos avotos, interneta tīmekļos. Ar plašo datu pieejamību, mākslīgā intelekta modulēšana atvieglojas, jo ar mazāk informāciju nepieciešami cilvēciski piemēri un sarežģīti algoritmi, turpretī ar vairāk informāciju var iztikt ar primitīvāku algoritmu. Šī iemesla dēļ pieeja mākslīgā intelekta veidošanai mainījusies no mehāniskas zināšanu iekodēšanas uz datora pašmācību pie dotiem datiem. [21]

Lai apstrādātu datus un trenētu mākslīgā intelekta modeļus, ir nepieciešams veids, kā šos lielos daudzums datu efektīvi glabāt. Lai to realizētu, dati tiek glabāti mākoņos, un to apstrādei tiek izmantoti jaudīgi skaitļotāji vai jaunas pieejas, kā datus apstrādāt.

Jau vairākus gadus, izpildoties Mūra likumam [22], mums ir pieeja jaudīgiem daudzkodolu procesoriem, kuri vari ātri un efektīvi veikt daudzus uzdevumus. No otras puses, milzīgu daudzumu datu apstrāde, pat tādiem procesoriem, aizņemtu pārāk daudz laika, lai apstrāde tiktu veikta efektīvi. Tādēļ cilvēki pievērsās grafiskajiem procesoriem jeb videokartēm [4], jo, atšķirībā no procesoriem, tām ir daudz vairāk atsevišķu kodolu, lai veiktu 3D kalkulācijas. Piemēram, grafiskais procesors “GeForce RTX 3070” satur 5888 kodolu, kuri paralēli veic savas funkcijas [23]. Šī grafisko procesoru īpatnība, veikt vienlaikus paralēlus darbus, padara tās par ļoti efektīvu rīku paralēlajā datu apstrādē [24]. Spilgtākais piemērs videokaršu pielietošanā, lai veidotu un apmācītu mākslīgo intelektu, ir CNN (konvolucionālie neironu tīkli). Tas ir neironu tīkls, kas apstrādā bildes. Lai paātrinātu bilžu apstrādi, bildes un neironu tīklu pārveido, kā divus masīvus, bet apstrādes rezultāts ir šo masīvu reizinājums. Šo masīvu reizinājumu var veikt paralēli ar videokaršu palīdzību, kas būtiski samazina laiku masīvu sareizināšanai.

Šo tehnoloģiju attīstības rezultātā, it īpaši pēdējos gados, ir visai strauji attīstījies mākslīgais intelekts, kuru ikdienā izmantojam, piemēram, rakstot *ChatGPT*, veidojot bildes ar *MidJourney* vai tulkojot tekstu ar *DeepL*.

2. **Algoritmu sarežģītība**

2.1 **Kas ir lielā O notācija?**

Lielā O notācija datorzinātnēs ir veids, kā pierakstīt algoritma sarežģītību, proti, cik daudz algoritma soļu skaits vai izmantotā atmiņa pieaug, palielinoties ievaddatu skaitam [25]. Lielo O notāciju pieraksta kā *O(n)*, kas cēlās no vācu matemātiķa “Paul Bachmann” pieraksta. *O(n)* apraksta funkcijas uzvedību, kad arguments tiecas uz noteiktu vērtību vai bezgalību, kur O – nozīmēja augšanas kārtību, bet n – argumenta lielumu [26].

Tostarp datorzinātnēs n definē, nevis, kā argumenta lielumu, bet kā argumentu skaitu vai ievaddatu skaitu [27]. Piemēram, runājot par algoritmu ātrumu, *O(n)* nozīmēs, ka ar n ievaddatu skaitu algoritms izpildīs n soļus – algoritms izpildās lineāri, bet algoritms ar sarežģītību *O(1)* izpildīsies konstanti, veicot 1 soli. Šī algoritmu aprakstīšanas metode ļauj viegli un saprotami salīdzināt algoritmus, jo, pat nezinot neko par algoritmiem, varēsim secināt, ka algoritms ar sarežģītību *O(n)* būs ātrāks par algoritmu *O()*, jo n < .

Svarīgi ir piebilst, ka turpmāk aprakstot, salīdzinot vai pētot algoritmu sarežģītību, īsināsim to šādi:

*O(n + 1)* pārveidosim par *O(n)* tā, ka viens papildus solis algoritma izpildes laiku ietekmēs ļoti minimāli. Tātad īsināsim nelielus konstanta soļus vai atmiņas skaitu.

2.2 **Lielās O notācijas nozīme mūsu projektā**

Mākslīgā intelekta veidošana, apmācīšana un pētīšana ir saistīta ar lielām un sarežģītām datu kopām, un pareiza algoritma un datu struktūru izvēle var būtiski mainīt risinājumu ātrumu un precizitāti. Kā jau tika minēts 2.1 nodaļā, lielā O notācija ir ērts rīks, kas ļauj viegli salīdzināt algoritmus un novērtēt tā grūtību. Tas būs ļoti svarīgi 6. un 7. uzdevuma izpildījumā, jo ar lielo O notāciju spēsim konkrēti aprakstīt mūsu pamatalgoritmu un datu struktūru ātrumus un prasības pēc atmiņas. Tas arī ļaus noskaidrot aģenta potenciālās vajadzības pēc atmiņas daudzuma un izšķirtspējām. Zinot šo informāciju, varēsim veikt pamatotus spriedumus par mākslīgā intelekta iespējām nākotnē.

**Atsauces**

1. Moore, G. E. Cramming more components onto integrated circuits (1965). Pieejams tiešsaistē [šeit](http://cva.stanford.edu/classes/cs99s/papers/moore-crammingmorecomponents.pdf) [23/09/2024]
2. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.10 nodaļa
3. “What is Cloud Computing?” “AI and Cloud Computing Explained” Pieejams tiešsaistē [šeit](https://www.oracle.com/artificial-intelligence/ai-cloud-computing/#:~:text=Cloud%20computing%20provides%20the%20necessary,data%20analytics%2C%20and%20improving%20cybersecurity) [23/09/2024]
4. Chellapilla, K.; Puri, S.; Simard, P. High Performance Convolutional Neural Networks for Document Processing (2006). Pieejams tiešsaistē [šeit](https://inria.hal.science/inria-00112631/document) [23/09/2024]
5. TensorFlow dokumentācija pieejama tiešsaistē [šeit](https://www.tensorflow.org/guide/migrate/mirrored_strategy) [23/09/2024]
6. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1 attēls
7. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.4 nodaļa
8. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1 nodaļa
9. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1 attēls
10. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1.1 nodaļa
11. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1.2 nodaļa
12. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1.3 nodaļa
13. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.1.4 nodaļa
14. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1 nodaļa
15. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.2 nodaļa
16. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.1 nodaļa
17. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.2 nodaļa
18. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.3 nodaļa
19. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.5 – 1.3.9 nodaļa
20. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.4 nodaļa
21. Russel, S. J.; Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.), 1.3.10 nodaļa
22. Our World in Data. Moore’s law: The number of transistors per microprocessor (2022). Pieejams tiešsaistē [šeit](https://ourworldindata.org/grapher/transistors-per-microprocessor) [30/09/2024]
23. GeForce RTX 30 Series. Pieejams tiešsaistē [šeit](https://www.nvidia.com/en-eu/geforce/graphics-cards/30-series/) [30/09/2024]
24. Meritt, R. Why GPUs Are Great for AI (2023). Pieejams tiešsaistē [šeit](https://blogs.nvidia.com/blog/why-gpus-are-great-for-ai/) [30/09/2024]
25. Bachmann, P. G. H. (1894). Analytische Zahlentheorie, 401. lpp
26. Paul E. Black, “big-O notation”, in [Dictionary of Algorithms and Data Structures](https://xlinux.nist.gov/dads/) [online], Paul E. Black, ed. 6 September 2019 Pieejams tiešsaistē [šeit](https://xlinux.nist.gov/dads/HTML/bigOnotation.html) [25/09/2024]
27. Mohr, A. Quantum Computing in Complexity Theory and Theory of Computation (2014). Pieejams tiešsaistē [šeit](https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=194d4f0ae54b2dbca9133c582cd2451eb13f3889) [25/09/2024]