

საქართველოს ტექნიკური უნივერსიტეტი  
გიორგი აბელაშვილი

მრავალაგენტიან ვიდეოთამაშებში კოლექტიური გადაწყვეტილებების  
მოდელები

წარმოდგენილია დოქტორის აკადემიური ხარისხის მოსაპოვებლად

სადოქტორო პროგრამა "ინფორმატიკა" შიფრი 0401

საქართველოს ტექნიკური უნივერსიტეტი  
თბილისი, საქართველო  
ივლისი, 2016 წელი

საავტორო უფლება © გიორგი აბელაშვილი  
თბილისი  
2016 წელი

სამუშაო შესრულებულია საქართველოს ტექნიკურ უნივერსიტეტში  
ინფორმატიკისა და მართვის სისტემების ფაკულტეტი  
ეკონომიკური ინფორმატიკის იდეპარტამენტი

ხელმძღვანელი : პროფ. ზურაბ ბოსიკაშვილი

რეცენზენტები-----  
-----

დაცვა შედგება-----წლის "-----"-----,----- საათზე  
საქართველოს ტექნიკური უნივერსიტეტის-----  
-----ფაკულტეტის სადისერტაციო საბჭოს კოლეგიის  
სხდომაზე, კორპუსი ----- , აუდიტორია----- მისამართი:  
0175, თბილისი, კოსტავას 77.

დისერტაციის გაცნობა შეიძლება სტუ-ს  
ბიბლიოთეკაში, ხოლო ავტორეფერატის - ფაკულტეტის გვერდზე

სადისერტაციო საბჭოს მდივანი პროფ. თინათინ კაიშაური.

საქართველოს ტექნიკური უნივერსიტეტი

ინფორმატიკის და მართვის სისტემების ფაკულტეტი

ჩვენ, ქვემოთ ხელისმომწერი ვადასტურებთ, რომ გავეცანით გიორგი აბელაშვილის მიერ შესრულებულ სადისერტაციო ნაშრომს დასახელებით: „ვიდეოთამაშებში კოლექტიური გადაწყვეტილებების მოდელები“ და ვაძლევთ რეკომენდაციას საქართველოს ტექნიკური უნივერსიტეტის ინფორმატიკისა და მართვის სისტემების ფაკულტეტის სადისერტაციო საბჭოში მის განხილვას დოქტორის აკადემიური ხარისხის მოსაპოვებლად.

თარიღი:

ხელმძღვანელი: პროფ. ზურაბ ბოსიკაშვილი

რეცენზენტი:

რეცენზენტი:

საქართველოს ტექნიკური უნივერსიტეტი  
2016 წელი

ავტორი: გიორი აბელაშვილი

დასახელება: „მრავალაგენტიან ვიდეოთამაშებში კოლექტიური  
გადაწყვეტილებების მოდელები”

ფაკულტეტი: ინფორმატიკისა და მართვის სისტემები

ხარისხი: დოქტორი

სხდომა ჩატარდა:

ინდივიდუალური პიროვნებების ან ინსტიტუტების მიერ ზემომოყვანილი დასახელების ნაშრომის გაცნობის მიზნით მოთხოვნის შემთხვევაში მისი არაკომერციული მიზნებით კოპირებისა და გავრცელების უფლება მინიჭებული აქვს საქართველოს ტექნიკურ უნივერსიტეტს.

---

ავტორის ხელმოწერა

ავტორი ინარჩუნებს დანარჩენ საგამომცემლო უფლებებს და არც მთლიანი ნაშრომის და არც მისი ცალკეული კომპონენტების გადაბეჭდვა ან სხვა რაიმე მეთოდით რეპროდუქცია დაუშვებელია ავტორის წერილობითი ნებართვის გარეშე.

ავტორი ირწმუნება, რომ ნაშრომში გამოყენებული საავტორო უფლებებით დაცული მასალებზე მიღებულია შესაბამისი ნებართვა (გარდა ის მცირე ზომის ციტატებისა, რომლებიც მოითხოვენ მხოლოდ სპეციფიურ მიმართებას ლიტერატურის ციტირებაში, როგორც ეს მიღებულია სამეცნიერო ნაშრომების შესრულებისას) და ყველა მათგანზე იღებს პასუხისმგებლობას.

**"ექვნიება ჩემს მეუღლე სალომეს და ჩემს შვილს ელენეს.  
ისინი ზუსტად ისეთები არიან, როგორებიც  
ბავშვობაში წარმომეგინა რომ იქნებოდნენ"**

## რეზიუმე

კომპიუტერული და ინფორმაციული ტექნოლოგიების განვითარების თანამედროვე ეტაპზე შეიქმნა ვიდეოთამაშების ინდუსტრია. ადამიანები ეთამაშებიან კომპიუტერებს (აგენტებს) ვირტუალურ გარემოში. რაც უფრო ინტელექტუალურია მოთამაშე კომპიუტერი მით უფრო საინტერესოა თამაში. განსაკუთრებით აქტუალურია კოლექტიური თამაშები, როდესაც კომპიუტერები ერთიანდებიან და კოლექტიურად ეთამაშებიან ადამიანებს. კოლექტიური გადაწყვეტილების მიღების ამოცანები ერთერთი მნიშვნელოვანია თანამედროვე ხელოვნური ინტელექტის სფეროში.

უამრავი უანრის ვიდეოთამაში არსებობს. მრავალფეროვანია ხელოვნური ინტელექტის ამოცანებიც ვიდეოთამაშებში. ამ ამოცანებში, ყველაზე ხშირად მოცემულია ინდივიდუალურ აგენტთა თეორია. ინდივიდუალური აგენტები მოქმედებენ დამოუკიდებლად, მათთვის არსებობს მხოლოდ კონკრეტული გარემო, არ არსებობენ სხვადასხვა მსგავსი აგენტები, არ ეკონტაქტებიან მათ, არ უზიარებენ, ან იღებენ ცოდნას, ან რაიმე ინფორმაციას ერთმანეთისგან.

არსებობს კონკრეტული შემთხვევები, როდესაც შესაბამისი ამოცანა შეიძლება უფრო სწრაფად და ოპტიმალურად ამოიხსნას მრავალაგენტიანი სისტემით, ანუ ინდივიდუალურ აგენტთა გაერთიანებით.

აგენტთა გაერთიანების შემთხვევაში, მათი გადაწყვეტილებები უკვე აღარ არის ინდივიდუალური, არამედ კოლექტიური. კოლექტიურია გადაწყვეტილება, როდესაც ჯგუფში რამოდენიმე აგენტის ცოდნაზე დაფუძნებით მიიღება. კოლექტიურ ხელოვნურ ინტელექტში აგენტები ერთმანეთს ცოდნას უზიარებენ, ეკონტაქტებიან ერთმანეთს, ისინი ყოველთვის არ არიან დამოუკიდებელი. ნაშრომის მიზანია შევქმნათ ოპტიმალური კოლექტიური გადაწყვეტილებების სისტემა. ამ სისტემის მიზანი იქნება ვაჩვენოთ, რომ ზოგიერთი ამოცანის ამოხსნა მრავალაგენტიანი მიდგომით უფრო ოპტიმალურად და სწრაფად იხსნება, ვიდრე ინდივიდუალურ აგენტთა სისტემაში. ჩვენს ნაშრომში გამოყენებული კვლევა სხვა თამაშებშიც შეიძლება გამოიყენონ, ან მსგავს ხელოვნური ინტელექტის ამოცანებში, სადაც მულტიაგენტური სისტემით მიდგომა გაცილებით კარგ შედეგს იძლევა.

კოლექტიური გადაწყვეტილებების განხილვისას შევეხებით ხელოვნური ინტელექტის ერთერთი მნიშვნელოვან თემას, არგუმენტაციის მექანიზმს[4,6]. აგენტთა კოლექტივებში არგუმენტაციის მექანიზმს დიდი ადგილი უკავია. აგენტები საკუთარი ცოდნიდან გამომდინარე ამბობენ არგუმენტებს კონკრეტულ გადაწყვეტილებაზე. ხოლო არგუმენტაციის მაკონტროლებელი მექანიზმი უზრუნველყოფს კოლექტიურ გადაწყვეტილებას.

არგუმენტაციის მექანიზმის ამოსავალი წერტილი არის ცოდნა. ხელოვნური ინტელექტის აგენტები თამაშებში აგროვებენ გარკვეულ ცოდნას, ფაქტებზე და პირობებზე დაყრდნობით. ამ ცოდნას ინახავენ ცოდნის ბაზაში[14]. ინდივიდუალური აგენტის შემთხვევაში, ცოდნა ინდივიდუალურად გროვდება - აგენტი მხოლოდ თავისი ქცევითა და გამოთვლებით ახდენს ცოდნის მოპოვებას, არ ართმევს, ან გადასცემს სხვა აგენტს ინფორმაციას. ჯგუფური ხელოვნური აგენტების სისტემაში კი აგენტები აგროვებენ საერთო ცოდნას. ისინი გარკვეული

საშუალებით ურთიერთობენ ერთმანეთთან. კოლექტიური ცოდნის მოდელისა და ინფორმაციის გაცვლისთვის საჭიროა მოქნილი ცოდნის გადაცემის მექანიზმის.

ვიდეოთამაშებში დღესდღეობით არსებობს ბევრნაირი ცოდნის გადაცემის სისტემა. ჩვენ განვიხილავთ ერთერთ ვერსიას - ცოდნის გადაცემა ცოდნის მარკირების საშუალებით.

ჩვენს ნაშრომში ცოდნის მარკირების სისტემა არის ერთერთი საფუძველი არგუმენტაციის სისტემის ჩამოყალიბებისა. ეს უკანასკნელი კი უზრუნველყოფს აგენტთა ჯგუფურ გადაწყვეტილებებს. ცოდნის მარკირების სისტემის საფუძველზე აგენტები ერთმანეთს უტოვებენ საჭირო ცოდნას გარემოს გარკვეულ პოზიციაზე. ხოლო სხვა აგენტები კი ამ ცოდნას პირდაპირი გზით იღებენ ცოდნის ბაზაში. მათ არ უწევთ სპეციალური ალგორითმების გამოყენება რათა ფაქტებისა და პირობების დაფუძნებით გამოითვალონ გარკვეული გადაწყვეტილება, ან ცოდნა დააგროვონ ამის შემდეგ. ჩვენ განვიხილავთ ამ მეთოდის დადებით და უარყოფით მხარეებს.

ნაშრომში მოცემული გვაქვს მაგალითი, ჭიანჭველების ამოცანა. ამოცანა განხილულია, როგორც კონკრეტული ვიდეოთამაშში, სადაც მნიშვნელოვანია ოპტიმალური შედეგი. პირობითად, მოცემულია ორგანიზმილებისანი სიბრტყე, სიბრტყეზე ჩნდებიან გარკვეული ტიპის ობიექტები, მათ შორის ფოთლები. ფოთლები შეიცავენ მარტივ ანბანურ ასო-სიმბოლოებს. სიბრტყეზე გამოდიან ჭიანჭველა-აგენტები, რომელთა მიზანია ფოთლების შეგროვება და საცავში გადატანა. თვითოეულ ჭიანჭველას შეუძლია ერთი ასო-ს ამოცნობა, ფოთოლზე შესაბამისი სიმბოლოს პოვნის შემთხვევაში იგი მოაჭრის ამ სიმბოლოს და გადაიტანს საცავში. თამაშის მიზანია, რაც შეიძლება მეტი ფოთოლი გადაიტანონ ჭიანჭველებმა გარკვეულ დროში.

ამოცანა განხილული გვაქვს როგორც ინდივიდუალურ აგენტთა სისტემაში, ასევე - მრავალაგენტიანში. ინდივიდუალური აგენტთა თეორიის მიხედვით ჭიანჭველების ამოცანა მიიღებს კლასიკურ, ერთაგენტიანი სისტემის ხელფენური ინტელექტის სახეს. ამ შემთხვევაში, ყოველ ჭიანჭველას მოუწევს, ყოველი ნაპოვნი ობიექტის შემოწმება, არის თუ არა ფოთოლი, შემდეგ სიმბოლოს ცნობა, ფოთლის დაჭრა და საცავში გადატანა. ხოლო მრავალაგენტიან სისტემაში, ჩვენ დაეყოფთ ჭიანჭველებს როლებად და ამოცანის ამოსხნას შევეცდებით ჭიანჭველათა ჯგუფებით[3].

## Abstract

Computer and information technology for the development of the modern stage set up videogame industry. A People play with computers (agents) in a virtual environment. Playing is very interesting when agents acts more intellectually. But more interesting, when computer Ai - agents join together and they collectively play with people. Collective decision-making is one of the important tasks of modern artificial intelligence field and also in videogames

There are lots of video game genres. Videogames includes many interesting and mesmerizing AI-problems. These problems, most often are based on Individual ai-agents theory. Individual agents operate independently, they are only for the specific environment, there are a variety of such agents, they do not communicate, do not share, or receive knowledge from, or any information about each other.

There are specific cases when the problem can be solved more quickly and optimally with multiagent ai-system, i.e. the union of the individual agents.

When agents are in union, many of their decisions aren't individual, but also collective. Collective decision is, when a main knowledge based on each individual knowledge of each agents of some group. Agents in group are not always individual, because they communicate each other, they exchange information and knowledge each other. The goal of this article is to create optimal decision-making solution for collective system of agents. The purpose of this system will show that some of the approaches to the solution of multiagent-Ai-system is more optimal and dissolves rather than individual agents in the individual ai-system. The paper used in our study can be used by other games, or similar artificial intelligence problems, where multiagenturi system approach is more effective.

So, Artificial Intelligence is one of the important issues discussed concerning collective decisions, reasoning mechanism. Argumentation mechanism takes huge place in modern Ai. Agents say some arguments based their own knowledge. There is a special argument-controlling-mechanism which controls collective decisions.

The starting point of argumentation mechanism is knowledge. Artificial intelligence agents gather some knowledge of the Games, based on the facts and circumstances. This knowledge is stored in the knowledge base. Individual agent's case, knowledge is collected individually - agent learns only with its individual acting. It doesn't communicate with other agents, it doesn't exchange any information or knowledge with other agents. But agents union collects common knowledge. They communicate each other with special communication mechanism. Unlike individual agents, they transfer each other's knowledge and some information. The collective knowledge of the agents group model and they exchange of information needed for a flexible mechanism for the transfer of knowledge.

Nowadays, in videogames there are many ways to transfer knowledge. We consider one version - Problem of Marking of Knowledge.

In this article we develop argumentation mechanism based on Problem of Marking of Knowledge, and this one also provides some collective decisions for our



article. In Problem of Marking of Knowledge, agents leave some form of knowledge to each other on certain positions of game world. And the other agents get this type of knowledge directly in their database of knowledge. They do not have to use special algorithms to calculate some of the facts and conditions to gain this kind of knowledge, because this knowledge is already final, final to "remember". We also discuss the advantages and disadvantages of this method.

In this paper we have the example of the ant task. The task is considered as a specific video game, which is essential for optimal results. There is a the two-dimensional plane. On the plane appear certain types of objects, including the leaves. The leaves contain a simple alphabetical letters. On the plane there are also the ant-agents, whose goal is to collect the leaves and take to repository. Each ant is able to recognize one of the letters on the leaf, if it found it, it cut the letter-character and takes in repository. The game goal is to take leaves as much as possible in special given time.

The task we have as individual agents in the system, as well as - in multiagent. Individual agents will take on the classic individual theory of ant activit - they should check every object to find leaves. After findining the leaf object, they must check symbol contain test. After this they cut the symbol and they take it in repository. And with multiagent system we will try to solve problem with group of ants.

## შინაარსი

შესავალი.....	16
თავი 1. ხელოვნური ინტელექტი ვიდეოთამაშებში.....	19
1.1 ხელოვნური ინტელექტის ძირითადი მიზანი.....	22
ვიდეოთამაშებში.....	22
1.2 ინტელექტუალურ აგენტთა თეორია.....	25
1.3 ინტელექტუალურ აგენტთა ტიპები ვიდეოთამაშებში.....	31
1.4. აგენტთა ტიპები.....	35
1.5. მარტივი რეფლექსური აგენტი. ....	36
1.6. მოდელზე დაფუძნებული რეფლექსური აგენტები.....	37
1.7. მიზანზე და მოდელზე დაფუძნებული აგენტები. ....	38
1.8. სარგებლიანობაზე დაფუძნებული აგენტები. ....	40
1.9. სწავლებადი აგენტები.....	41
თავი 2. ცოდნის წარმოდგენა და დამუშავება ვიდეოთამაშების ხელოვნური ინტელექტის სისტემაში.....	43
2.1. ცოდნის წარმოდგენის აუცილებლობა. ....	43
2.2. სემანტიკური ქსელები. ....	44
2.3. ფრეიმები.....	49
2.4. კონცეპტუალური გრაფები. ....	51
თავი 3. თამაში "ჭიანჭველების ამოცანა".....	56
3.1 ამოცანის განსაზღვრა.....	56
3.2 ერთაგენტიანი და მრავალაგენტიანი სისტემა.....	57
3.3 ჭიანჭველების ამოცანის ერთაგენტიანი და მრავალაგენტიანი მოდელი.....	59
3.4 ცოდნის გადაცემათა ზოგიერთი ტიპი ვიდეოთამაშებში.....	61
3.5 ჭიანჭველების ამოცანის მრავალაგენტიანი მოდელი.....	63
3.6 ჭიანჭველების ამოცანის უნარჩვევებზე დაფუძნებული მრავალაგენტიანი მოდელი Skill Based Model.....	65
3.7 ცოდნის მარკირების ამოცანა, როგორც ცოდნის გადაცემის საშუალება ჭიანჭველების ამოცანაში.....	68
3.8 ცოდნის მარკირება ჭიანჭველების ამოცანაში მზვერავი ჭიანჭველის შემთხვევაში.....	70
3.9 ცოდნის მარკირება ჭიანჭველების ამოცანაში ფოთლის მჭრელი ჭიანჭველის შემთხვევაში.....	72
3.10 ცოდნის მარკირება ჭიანჭველების ამოცანაში ფოთლის გადამზიდი ჭიანჭველის შემთხვევაში.....	74
თავი 4. ობიექტთა გამოცნობა ხელოვნურ ინტელექტში.....	76
4.1 სტრუქტურული აღმწერები.....	80
4.2 ობიექტების გრაფებად წარმოდგენა და.....	85
ამოცნობის სხვადასხვა მიდგომები.....	85
თავი 5. არგუმენტაციის მექანიზმი ვიდეოთამაშებში.....	89
5.1 არგუმენტაციის მექანიზმი ხელოვნურ ინტელექტში.....	89
5.2 კლასიკური არგუმენტაცია.....	93
5.3 მნიშვნელობაზე დამოკიდებული არგუმენტაცია.....	94
5.4 სემანტიკური შეთავსების მოდელი არგუმენტაციის მექანიზმისათვის.....	95
5.4 თვისობრივი არგუმენტების გენერირება.....	96
5.5 მიმართებითი არგუმენტების გენერირება.....	97
5.6 აგენტების მიერ სტრუქტურული არგუმენტების გენერაცია.....	98

თავი 6. კოლექტიური გადაწყვეტილებების მიღება ვიდეოთამაშებში.....	100
6.1 მულტიაგენტური სისტემის ზოგადი დახასიათება .....	100
6.2 მულტიაგენტური სისტემის საჭიროება .....	103
ჭიანჭველების ამოცანაში.....	103
6.3 კოლექტიური გადაწყვეტილების მაკონტროლებელი მექანიზმი-შავი დაფის არქიტექტურა. ....	104
6.4 შავი დაფის არქიტექტურა სტრატეგიულ ვიდეოთამაშებში .....	105
6.4 მზვერავთა გუნდის დინამიური ფორმირება დაკოლექტიური გადაწყვეტილებები.....	106
6.5 ფოთლის მჭრელთა კოოპერაცია და ჯგუფური გადაწყვეტილებები .....	109
თავი 7. ჭიანჭველების ამოცანის პროგრამული არქიტექტურა .....	112
7.1 ჭიანჭველის, როგორც პროგრამული აგენტის სტრუქტურა .....	112
7.2 მზვერავი ჭიანჭველის ქცვის ძირითადი ფსევდოკოდი.....	114
7.2 ფოთლის მჭრელის მოქმედების ძირითადი ფსევდოკოდი .....	117
7.3 ჭიანჭველა-გადამზიდის მოქმედების ძირითადი ფსევდოკოდი.....	120
დასკვნა.....	123
გამოყენებული ლიტერატურა.....	125

## ნახაზების ნუსხა

სურ. 1 pac man .....	23
სურ. 2 Pong .....	23
სურ. 3 Assassin's Creed .....	24
სურ. 4 FarCry 3 .....	24
სურ. 5 გარემო, როელშიც მოქმედებს აგენტი.....	27
სურ. 6 რეფლექსური აგენტის სტრუქტურა .....	36
სურ. 7 მოდელზე დაფუძნებული რეფლექსური აგენტი.....	39
სურ. 8 მიზანზე და მოდელზე დაფუძნებული აგენტის სტრუქტურა .....	39
სურ. 9 სარგებლიანობაზე დაფუძნებული აგენტის სტრუქტურა.....	41
სურ. 10 კოლინის და კეილის მიერ შედგენილი სემანტიკური ქსელი.....	46
სურ. 11 სამი სიბრტყე, რომელშიც წარმოადგენენ სიტყვა "პლანტ"-ის სამ განმარტებას.....	47
სურ. 12 ცებების cry და comfort თანაკვეთა.....	48
სურ. 13 კუბის გამოსახულება და მისი ფრეიმი.....	50
სურ. 14 სხვადასხვა ნ-არობის კონცეპტუალური მიმართებები.....	52
სურ. 15 კონცეპტუალური გრაფები, რომლებიც გვიჩვენებენ მარკერის და სახელის გამოყენების სხვადასხვა ვაიანტებს .....	54
სურ. 16 კონცეპტუალური გრაფი წინადადებისათვის "ძალი იფხანს ყურს თათათით" .....	55
სურ. 17 ჭიანჭველების თამაშის გრაფიკული პროტოტიპი.....	56
სურ. 18 ბრძოლის სცენა თამაშიდან Call Of Duty Ghosts 2.....	58
სურ. 19 ვიდეოთამაში Stronghold Crusaders 2.....	59
სურ. 20 საერთო გონების მქონე აგენტთა მოქმედება .....	62
სურ. 21 აგენტების მიერ სიგნალების მეშვეობით გაზიარებული ცოდნის გრაფიკული მოდელი.....	63
სურ. 22 თამაში Dark Souls პერსონაჟის უნარჩვევების განსაზღვრა თამაშის დაწყების წინ .....	64
სურ. 23 თამაში World Of Warcraft პერსონაჟის უნარჩვევების განსაზღვრა თამაშის აწყების წინ.....	65
სურ. 24 თამაში Mount And Blade პერსონაჟის.....	65
სურ. 25 ცოდნის მიწოდება არაპირდაპირი გზით.....	69
სურ. 26 ცოდნის მიწოდება პირდაპირი გზით.....	70
სურ. 27 სანამ დაიწყება შაბლონის დამთხვევა, ნიმუშს შესაძლოა შეუცვალონ ორიენტაცია, ან ზომა. ....	76
სურ. 28 სურათზე ნაჩვენებია თუ როგორ შეიძლება ასო -ს შაბლონი დაემთხვეს ასო ღ -ს შაბლონს და პირიქით .....	77
სურ. 29 ასო Q და ასო O-ს სიმბოლოთა ვარიანტები რომელთა გარჩევაც ხდება.....	77
სურ. 30 ადამიანის ასოების აღქმის უნარის შინაარსი .....	79
სურ. 31 ასო T -ს სტრუქტურული აღწერა.....	81
სურ. 32 ასო T-ს სამგანზომილებიანი ფიგურა .....	82
სურ. 33 სათამაშოების ნიმუში, რომელთა ცნობაც შეეძლო ვინსტონის პროგრამას.....	83
სურ. 34 ტრენინგის მიმდევრობა.....	83
სურ. 35 ზევიდან-ქვევით ტიპის გამოცნობის არქიტექტურა.....	87
სურ. 36 კომბინირებული ტიპს გამოცნობის არქიტექტურა.....	88

სურ. 37	დენგის არგუმენტაციის ფრეიმვორქის მაგალითი.....	93
სურ. 38	დენგის თეორიის გავრცობის რამდენიმე ვარიანტი.....	95
სურ. 39	მხვერავთა გუნდური ფორმირება და საერთო რადიუსი.....	107

### ცხრილების ნუსხა

ცხრილი 1 თვისობრივი არგუმენტების შეთავსებადობა .....	97
ცხრილი 2 მიმართებითი არგუმენტების შეთავსებადობა .....	97
ცხრილი 3 დაგენერირებული სტრუქტურული არგუმენტების ცხრილი..	99
ცხრილი 4 ფოთლის მჭრელთა საწყისი პარამეტრების ცხრილი.....	110
ცხრილი 5 ფოთლის მჭრელის ჰემ-ცხრილის გარკვეული ცვლილებები გარკვეულ ბიჯზე .....	111

მადლიერება

დიდ მადლობას მოვასხენებ ჩემს ხელმძღვანელს, ზურაბ ბოსიკაშვილს,  
გაწეული დახმარებისათვის

## შესავალი

ადამიანები იმით გამოირჩევიან სხვა ცოცხალი არსებებისაგან, რომ მათ შეუძლიათ აზროვნება. უკვე მრავალი საუკუნეა ადამიანი ცდილობს ამოხსნას აზროვნების ფენომენი, გაიგოს თუ როგორ ახერხებს გარე სამყაროს შეცნობას გრძნობათა ორგანოების საშუალებით, უკვე ერთხელ შეხვედრილი სიტუაციების დამახსოვრებას, სწავლას მათ საფუძველზე, თავისი ცოდნის განზოგადებას, ლოგიკური მსჯელობის საფუძველზე სწორი გადაწყვეტილებების მიღებას და თავისი მოთხოვნილებების დასაკმაყოფილებლად გარე სამყაროს მართვასა და შეცვლას. სწორედ ასეთ ქმედებათა ერთობლიობა ქმნის ადამიანის ისეთ ქცევას, რომელსაც ჩვენ ვუწოდებთ ინტელექტუალურ ქცევას.

ხელოვნური ინტელექტი არის მეცნიერების დარგი, რომელიც ცდილობს არა მარტო შეიცნოს ინტელექტის ბუნება, არამედ შექმნას კიდევ (ზოგჯერ სწორედ ესაა უფრო მთავარი) ინტელექტუალური ქცევის მქონე ხელოვნური სისტემები, ანუ როგორც მათ უწოდებენ ხელოვნური ინტელექტუალური სისტემები.

დღესდღეობით ადამიანთა ცხოვრებაში ერთერთი განუყოფელი ნაწილია ვიდეოთამაშები. არსებობს უამრავი უანრისა და სახის ვიდეოთამაში, მათი მუშაობის ყველაზე გავრცელებული ვარიანტი სწორედ, ხელოვნურ ინტელექტზე არის დაფუძნებული - არსებობს მთავარი მოთამაშე ადამიანი, ხოლო ადამიანი ეთამაშება ხელოვნურად შექმნილ "ადამიანის ტვინის სიმულაციას" - ხელოვნურ ინტელექტის აგენტებს.

ვიდეოთამაშებში ხელოვნური ინტელექტის მრავალფეროვანი ამოცანები არსებობს. ძირითადი პრინციპი საერთოა - ხელოვნურმა აგენტმა რაც შეიძლება მეტი წინააღმდეგობა გაუწიოს ადამიან მოთამაშეს, რათა მიზანი ნაკლებად იქნეს მიღწეული. რადგან თამაშები გათვლილია მომხმარებელზე, ასევე გათვალისწინებულია ხელოვნური ინტელექტისა და აგენტების დონეები. რაც უფრო მეტად "ჭკვიანია" აგენტი მიტ უფრო რთულია მისთვის თამაშის მოგება.



მულტიაგენტური ტექნოლოგია ფართოდ გამოიყენება ვიდეოთამაშებში. იგი აერთიანებს თამაშების თეორიის, რთული სისტემების, გამოთვლითი სოციოლოგიისა და ხელოვნური ინტელექტის საკითხებს. ვიდეოთამაშებში ხელოვნური ინტელექტის ამოცანები, ძირითადად, დამყარებულია ინდივიდუალურ აგენტზე - ერთი ინდივიდი რომელსაც არ ჰყავს არც დამხმარე და არც ალტერნატივა, ცდილობს მოთამაშეს გაუწიოს მეტოქეობა, გამოიცნოს მისი სვლები, დაიმახსოვროს გარკვეული გადაწყვეტილებები, გააკეთოს დასკვნები. ინდივიდუალური აგენტის შემთხვევაში ხელოვნური აგენტის მოქმედება უნიფიცირებულია და მორგებულია როგორც ერთ ინდივიდზე.

დისერტაციის მთავარი ამოცანის განხილვამდე საჭიროა განვიხილოთ ზოგიერთი მნიშვნელოვანი თემატიკა, რაც ვიდეოთამაშებში კოლექტიური გადაწყვეტილებებისთვის გამოიყენება, მაგალითად, არგუმენტაციის მექანიზმი, მრავალაგენტური სისტემაში და ცოდნის გადაცემის მექანიკა.

დისერტაციის კვლევის საგანია ვიდეოთამაშების ხელოვნური ინტელექტი. ვიდეოთამაშებში აგენტთა ერთმანეთთან ცოდნის გაცვლა, რითაც საბოლოო ჯამში იქმნება თამაშების ხელოვნური ინტელექტის კოლექტიური გადაწყვეტილებები.

იმისათვის, რომ კოლექტიური გადაწყვეტილებების მექანიზმი ავაგოთ, საჭიროა განვიხილოთ ერთაგენტური და მრავალაგენტური სისტემები, მათი მოქმედების პრინციპები. მნიშვნელოვანია ცოდნის გადაცემის მექანიზმი, ნაშრომში ასევე გვაქვს მეცნიერულად განხილული არგუმენტაციის მექანიზმი.

კვლევის მიზანია შევქმნათ მრავალაგენტური ვიდეოთამაშებში კოლექტიური გადაწყვეტილებების მიღების მექანიზმი. უამრავი თამაში არსებობს, სადაც ხელოვნური ინტელექტის მრავალფეროვანი ამოცანებია მოცემული. სამწუხაროდ, ბევრი ამოცანის წყარო დახურულია და მხოლოდ კონკრეტული მფლობელებისათვის ან კომერციული კლიენტებისათვისაა წვდომადი. ჩვენ ამოცანას ჩამოვაყალიბებთ, როგორც ღია კოდის მქონე რესურსს, რომელსაც გამოიყენებენ სხვადასხვა ვიდეოთამაშების შემქმნელები.

სადისერტაციო ნაშრომში მეცნიერულ სიახლეს წარმოადგენს:

- მრავალაგენტიან სისტემაში ცოდნის გადაცემის ტიპი - მარკირება, როდესაც ჩვენ შეგვიძლია მრავალაგენტიან სისტემაში რაღაც ნიშნით, გარკვეულ ადგილას დავტოვოთ ცოდნა სხვა ინტელექტუალური აგენტისთვის. მისთვის ეს ცოდნა იქნება პირდაპირი გზით აღებული, რაც სისტემის მუსაობის სისწრაფეს განსაზღვრავს
- მრავალაგენტიან ვიდეოთამაშებში, ცოდნის მარკირების ამოცანით და არგუმენტაციის კონკრეტული მექანიზმით კოლექტიური გადაწყვეტილებების მექანიზმის შექმნა. ამ მექანიზმის მიხედვით ხდება აგენტთა ჯგუფების დინამიური ფორმირება და კოლექტიურად მოქმედება.

ჩვენს ნაშრომში ჩატარებული კვლევა, შეიძლება გახდეს უნიფიცირებული ფრეიმვორქი, რომლის გამოყენებაც შეიძლება თამაშების სფეროს გასცდეს. შეიძლება ჩვენი სისტემა მოერგოს ზოგად სფეროებს, სადაც არგუმენტაციის მექანიზმი, ცოდნის გადაცემა და მრავალაგენტიანი მექანიზმების კოლექტიური გადაწყვეტილებები აქტუალურია.

დისერტაცია შედგება 126 (ასოცდაექვსი) გვერდისაგან. და მოიცავს, ხელშეწყობის ინტელექტის ზოგად მიმოხილვას ვიდეოთამაშებში, ცოდნის წარმოდგენას და დამუშავებას ვიდეოთამაშებში, აღწერილია კონკრეტული ამოცანა-ვიდეოთამაშში - "ჭიანჭველების ამოცანა", სადაც ჩვენი საკვლევი თემის ექსპერიმენტი აღწერილია. აღწერილია სიმბოლოთა გამოცნობის მექანიზმები, ასევე არგუმენტაციის მექანიზმი ვიდეოთამაშებში და არგუმენტაციის საფუძველზე კოლექტიური გადაწყვეტილებების მექანიზმი ვიდეოთამაშებში.

## თავი 1. ხელოვნური ინტელექტი ვიდეოთამაშებში

ვიდეოთამაშებში მულტიაგენტურ ტექნოლოგიის ძირითადი სიბლი მდგომარეობს აგენტთა გაერთიანებებში საერთო ცოდნისა და მიზნის ქვეშ, წარმოვიდგინოთ სიტუაცია, როდესაც მოთამაშეს ეწინააღმდეგება არა ერთი აგენტი, როგორც ერთი ხელოვნური ინდივიდი, არამედ მსგავსი, ან სხვადასხვა ტიპის აგენტთა გაერთიანება. აგენტთა გაერთიანება შეიძლება იყოს მოქნილი მექანიზმი ხელოვნური ინტელექტის ამოცანისთვის. ამ შემთხვევაში, არსებობს ჯგულის საერთო ცოდნა.

სამუშაოს მიზანია ვაჩვენოთ ვიდეოთამაშებში აგენტთა ჯგუფებად გაერთიანების გარკვეული მეთოდები, რომელიც საბოლოოდ იქნება როგორც საერთო, ზოგადი მოდელი. ამ მოდელის მიზანია, გახდეს ინტელექტუალური ბირთვი, რომელიც მოერგება ნებისმიერ თამაშს და თავისი მუშაობის პრინციპიდან გამომდინარე, ეფექტური იქნება კონკრეტული ამოცანისთვის.

აგენტთა კოლექტივად გაერთიანების ბევრი მეთოდი არსებობს. ჩვენ განვიხილავთ არგუმენტაციის მექანიზმს, როდესაც აგენტთა ჯგუფის დინამიურად ფორმირება ხდება. ფორმირება არის დამყარებული თვითოეული აგენტის ინდივიდუალურ ცოდნასა და მოქმედებაზე. ცონდა მოიპოვება აგენტების დამოუკიდებელი მუშაობისას. მათი არგუმენტები არის დამყარებული ე.წ "პირად გამოცდილებაზე".

კვლევის ასევე ერთერთი ძირითადი მიზანია შევიმუშაოთ ცოდნის გადაცემის კარგი მექანიზმი. ცოდნის გადაცემა ხელოვნურ ინტელექტში ხდება უამრავი გზით. ყველა მათგანს აქვს დადებითი და უარყოფითი მხარეები. ძირითადი უარყოფითი თვისება არის კომპიუტერული რესურსი - ამოცანიდან გამომდინარე, შეიძლება, დიდი რაოდენობის მონაცემის დამუშავება მოუწიოს აგენტს. ინფორმაციის დიდი რაოდენობით დამუშავება პირდაპირ კავშირშია რესურსთან. ჩვენ ამ მხრივ განვიხილავთ ცოდნის მარკირების ამოცანას, როდესაც ცოდნა აგენტს მიეწოდება პირდაპირი გზით რაღაც ნიშნულზე. ამ შემთხვევაში, აგენტს არ უწევს სპეციალური გამოთვლების კეთება დასკვნის მისაღებად, ცოდნა პირდაპირი გზით გადადის ცოდნის ბაზაში. მულტიაგენტური სისტემა ადვილად ამუშავებს ცოდნის გადაცემას, რადგან ამ

შემთხვევაში შეიძლება აგენტთა როლების გადანაწილება კოლექტივში. აგენტთა როლები უზრუნველყოფს დამოუკიდებელ პროცესებში მიმდინარეობდეს გამოთვლები.

კვლევის ძირითადი თემა შეეხება თამაშებს, შესაბამისად, საკვლევი ობიექტიც არის თამაში. თამაშების ჟანრობრივი სხვაობა მათ სპეციფიკაციასაც განასხვავებს ერთმანეთისგან. ხელოვნური ინტელექტი ნებისმიერი ჟანრის თამაშისთვის იდეალურად შეგვიძლია მივასადაგოდ, რადგან, ასე ვთქვათ, თამაშებია შექმნილი ხელოვნური ინტელექტისთვის და პირიქით.

ყველაზე კარგი ჟანრი ვიდეო თამაშებში ხელოვნური ინტელექტის ექსპერიმენტებისთვის არის ე.წ. "რეალური დროის სტრატეგია". ამ ტიპის ვიდეოთამაშში გვექნება წარმოდგენილი კვლევაშიც.

"რეალური დროის სტრატეგია" ჟანრის თამაშში დაფუძნებულია მოთამაშის მიერ რესურსების მოპოვებაზე, ასევე მის მიერ მართვად აგენტთა ჯგუფის შექმნაზე. ასეთი ჯგუფები თამაშში ასევე იქმნება ხელოვნური ინტელექტის მეშვეობით, მოთამაშის მიზანია, რეალურ დროში აჯობოს ხელოვნურ ინტელექტს.

ასეთი ტიპის თამაშებში შეიძლება მოთამაშე და ხელოვნური ინტელექტი ერთმანეთს არ ებრძოდნენ. უბრალოდ, მოთამაშე მართავდეს ხელოვნური ინტელექტის საწყის გადაწყვეტილებებს, ხოლო ინტელექტი ავტონომიურად ავითარებდეს მას. ასევე, შეიძლება ხელოვნური ინტელექტი და ადამიანი მოთამაშე საერთო მიზნის ქვეშ გაერთიანდნენ. ჩვენი ამოცანა ზემოთხამოთვლილთაგან ოქროს შუალედს იცავს. იგი გვთავაზობს რთული ამოცანის ოპტიმალურ გადაწყვეტას მოთამაშისა და ხელოვნური ინტელექტის თანამშრომლობით. ხელოვნური ინტელექტი თუ სისტემაა, რატომ არ შეიძლება მოთამაშე ამ სისტემის ნაწილი იყოს და ამავე დროს სისტემის არამონაწილე.

კვლევაში მოცემულია ერთერთი საინტერესო ვიდეოთამაშის ვერსია, ჭიანჭველების კოლონიზაცია, მათი ცხოვრების პრინციპი, საზრდოს მოპოვება და საცავში გადატანა, მოთამაშე იქნება ერთერთი ჭიანჭველა, რომელიც ამ ხელოვნური ინტელექტის სისტემაში მიიღებს მონაწილეობას.

დღესდღეობით დიდი კომპანიები ქმნიან ასევე დიდ და საინტერესო თამაშებს. ამ თამაშების ინტელექტუალური ბირთვი არის დახურული და საავტორო უფლებები დაცულია. კვლევის აქტუალურობა იმაში მდგომარეობს, რომ ჩვენი ხელოვნური ინტელექტის სისტემა იყოს ღია და ყველას მიუწვდებოდეს ხელი. ამ სისტემის ინტელექტუალური ბირთვის ცალკე გამოყოფა, როგორც უნიფიცირებული ფრეიმვორქი, საშუალებას მოგვცემს მსგავსი მიდგომით ამოვხსნათ რთული ამოცანები, არამარტო თამაშებში, არამედ სხვადასხვა სამეცნიერო საკითხებში. ასევე, ამ ნაშრომში განხილული მეთოდები შეიძლება გამოყენებული იქნას სხვადასხვა ვიდეოთამაშებზე მომუშავე დეველოპერების მიერ კონკრეტულ პროექტებში.

აღრეულ 2000 იანებში, ვიდეოთამაშების ინდუსტრია აღორძინდა და თითქმის ჰოლივუდის კინოპროდუქციას დაეწია. შემდეგ წლებში ვიდეოთამაშები გახდა მარკეტინგის ერთერთი ძირითადი სფერო, სადაც შეიძლებოდა დიდი და საინტერესო სერვისის, გართობის შეთავაზება. ვიდეოთამაშების განვითარებასთან ერთად, ადამინი ცდილობს მაქსიმალურად მიუახლოვოს ვირტუალური სამყარო რეალურს. თუმცა დღესდღეობით, ვიდეოთამაშებს მაინც ბევრი უკლია სრულყოფილებამდე. ვიდეოთამაშების ძირითადი შემადგენელი კომპონენტი არის ხელოვნური ინტელექტი, მისი ძირითადი მიზანია, თამაშის პრინციპიდან გამომდინარე ადამიანივით "იფიქროს".

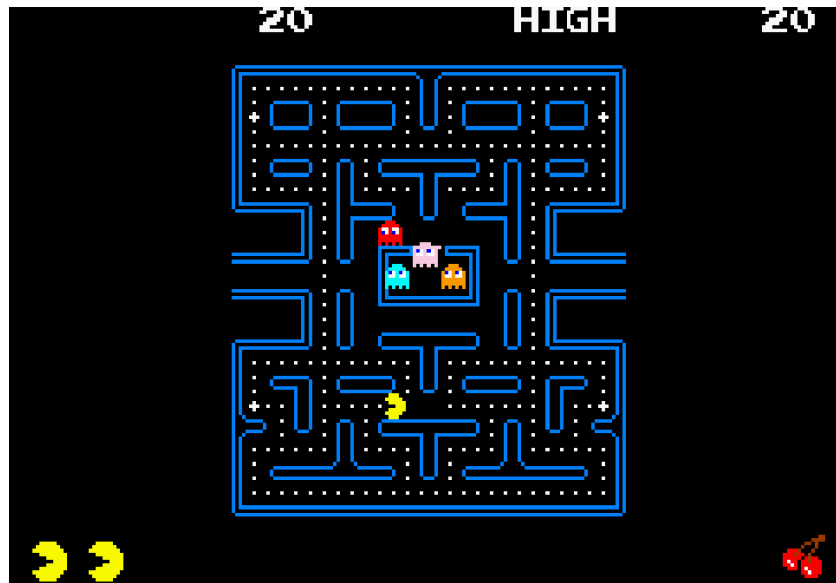
ბოლო წლებში თამაშები დიდი სისწრაფით განვითარდა. მიაღწია ვიზუალურად მაღალ დონეს, ასევე ბევრმა თამაშმა მოიპოვა უდიდესი აღიარება განსაკუთრებულად შესრულებისათვის. თამაშების ასეთ განვითარებასთან ერთად ვერ განვითარდა მათი ხელოვნური ინტელექტი, რათგან აქ მოცემული სისტემები დამყარებულია არაადაპტირებულ ტექნიკაზე [1]. ძირითადი სისუსტე არაადაპტირებული ინტელექტისა არის ის, რომ როდესაც მასში სისუსტეს აღმოაჩენს მოთამაშე, მაშინ მას ვეღარაფერი შეაჩერებს თამაშის მოსაგებად. ამის აღმოფხვრა შეიძლება მხოლოდ ადაპტირებული ხელოვნური ინტელექტით, როდესაც აგენტი სწავლობს გარემოში არსებულ სიტუაციას და აკეთებს არსებულ გარემოში ადაპტირებას დინამიურად. დღესდღეობით ადაპტირებული ხელოვნური ინტელექტი არ არის

სრულად განხორციელებული ვიდეოთამაშებში, რადგან ის მოითხოვს სპეციალურ ექსპერიმენტებს, დროს, მოქნილ სამართავ მექანიზმს.

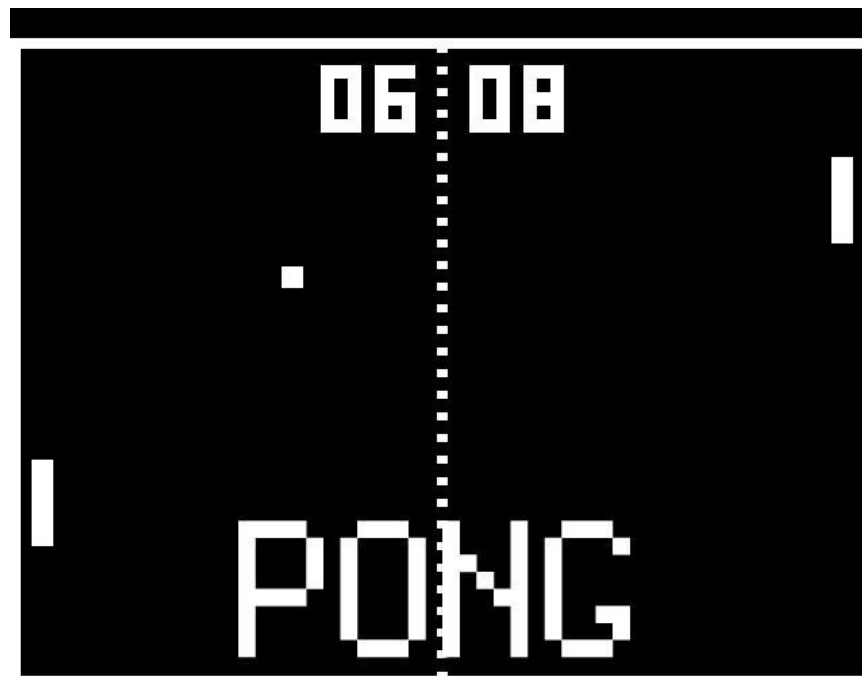
Abt (1970) განსაზღვრავს თამაშს, ორ დამოუკიდებელ გადაწყვეტილების მიმღებს შორის. ისინი მოქმედებენ რაღაც შეზღუდულ გარემოში. ამ განმარტებით გადაწყვეტილების მიმღები არის მოთამაშე და გარემო - ვიდეოთამაშის სივრცე, სადაც კონკრეტული თამაშის წესები მოქმედებს. მოქნილი ხელოვნური ინტელექტის სისტემის მეშვეობით, ასეთი თამაში მხოლოდ ადამიანს როდი შეუძლია, მასთან ერთად მონაწილეობს იღებენ ხელოვნური ინტელექტის მქონე აგენტები.

### **1.1 ხელოვნური ინტელექტის ძირითადი მიზანი ვიდეოთამაშებში**

ხელოვნური ინტელექტი ვიდეოთამაშებში, ძირითადად, არის გადაწყვეტილებათა მიმღები სისტემა. ამ სისტემით მოქმედებენ არამოთამაშე სუბიექტები და ობიექტები ადრეული თამაშები მაგ. Pac Man(სურ 1), Pong(სურ 2). წარმოადგენდნენ მარტივი ხელოვნური ინტელექტის ნიმუშებს. ამ თამაშებშიარსებული გარემო არ საჭიროებდა განსაკუთრებულ ალგორითმების მოფიქრებას. თუმცა თანამედროვე თამაშებმა წარმოშვა ისეთი სირთულეები, როგორცაა რეალური სამყაროს მოდელის დამოკიდებულება მოთამაშესა და გარემოზე Assassin's Creed(სურ 3), Farcry 3(სურ 4). თანამედროვე თამაშებში გარემო იყო მაქსიმალურად სრულყოფილი და მიახლოებული რეალურთან.



სურ. 1 pac man



სურ. 2 Pong



სურ. 3 Assassin's Creed



სურ. 4 FarCry 3

ხელოვნური ინტელექტის ძირითადი მიზანია შექმნას მოთამაშისთვის საინტერესო შეჯიბრის გარემო. მოთამაშისთვის ერთადერთი მიზანი კი თამაშიდან მიღებული სიამოვნებაა. Spronck [21] ახასიათებს თამაშების ხელოვნური ინტელექტის 7 ძირითად მიზანს, ეს მიზნები განსაზღვრავს ინტელექტის სირთულესაც.

1) **სამართლიანი თამაში** - ხელოვნური ინტელექტი უნდა ემორჩილებოდეს თამაშის ყველა წესს, არ უნდა იქცეოდეს ამ წესების გვერდის ავლით, რათა მოთამაშისთვის უცნობი გარემო არ შეიქმნას.



2) **ამოუცნობი ქცევა** - ხელოვნური ინტელექტის ნებისმიერი ქცევა უნდა იყოს მოთამაშისთვის მოულოდნელი. რატომაუნდა უინტერესო იქნებოდა თამაში, რომ ვიცოდეთ ჩვენი მოწინააღმდეგე რას მოიმოქმედებს.

3) **არაცხადი ქცევა** - სისტემის ნებისმიერ შემდეგი ქცევა არ უნდა იყოს აშკარა მოთამაშისთვის, არ უნდა აფროთხილებდეს მას.

4) **გარემოს გამოყენება** - სისტემა ნებისმიერ გარემოსთვის უნდა იყენებდეს ინტელექტუალურად მის სასარგებლოდ. საინტერესო და რთული სათამაშო იქნებოდა თუ მტერი ნისლიან გარემოში დაემალებოდა მოთამაშეს.

5) **თვითშესწორება** - ხელოვნური ინტელექტს უნდა შეეძლოს საკუთარი ქცევების კორექტირება და ცოდნის რედაქტირება. ნებისმიერ დროს უნდა შეეძლოს შაბლონის გადაკეთება და სხვა სტილით თამაში, ამ შემთხვევაში მოთამაშე რთულად აუღებს ადლოს მის პრინციპებს.

6) **კრეატიულობა** - ხელოვნური ინტელექტი უნდა გამოსავალს ნებისმიერი სიტუაციიდან, მათ შორის გაუთვალისწინებელიდანაც.

7) **ადამიანის მსგავსი ქცევა** - ხელოვნური აგენტის ქცევა მაქსიმალურად უნდა იყოს მიახლოებული მოთამაშის ქცევასთან.

ზემოთხამოთვლილი პრინციპებიდან, ვიდეთამაშებში იყენებენ პირველ ოთხ პრინციპს, დანარჩენი კი უფრო სამეცნიერო კვლევებისთვისაა.

## 12 ინტელექტუალურ აგენტთა თეორია

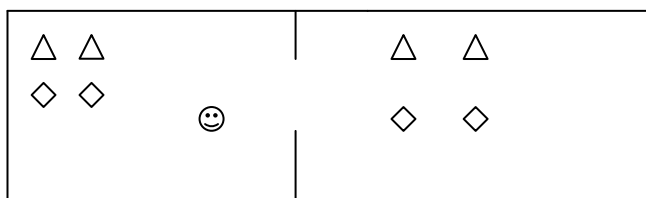
**ინტელექტუალური აგენტის ცნება.** აგენტად შეიძლება ჩაითვალოს ყველაფერი, რაც მოქმედებს (სიტყვა აგენტი ლათინური სიტყვაა და ნიშნავს აგერე - იმოქმედე), მაგრამ იგულისხმება, რომ კომპიუტერული აგენტები ჩვეულებრივი “პროგრამებისაგან” განსხვავებით ფლობენ კიდევ სხვა თვისებებს. ასე მაგალითად, მათ შეუძლიათ ფუნქციონირება ავტონომიურად, აქვთ გარემოს აღქმის უნარი, არსებობენ ხანგრძლივი დროის განმავლობაში, ადაპტურები არიან ცვლილებების მიმართ და რაც ყველაზე უფრო მთავარია შეუძლიათ თავის თავზე აიღონ სხვების მიერ დასმული მიზნების მიღწევა. **რაციონალურია** აგენტი, თუ მას შეუძლია მიაღწიოს საუკეთესო შედეგებს დეტერმინირებულ

გარემოში, ან თუ გარემო არაა დეტერმინირებული – საუკეთესო მოსალოდნელ შედეგებს. რაციონალურობის კონცეფცია შეგვიძლია გამოვიყენოთ ნებისმიერი აგენტის მიმართ, რომელიც მოქმედებს ნებისმიერ გარემოში, რომლის წარმოდგენაც ჩვენ შეგვიძლია. თავის მხრივ აგენტი შეიძლება იყოს ნებისმიერი სისტემა, რომელიც აღიქვამს გარემოს **გადამწოდების** საშუალებით და ზემოქმედებს ამ გარემოზე **შემსრულებელი მექანიზმების** საშუალებით. თუ განვიხილავთ ადამიანს აგენტის როლში, მას გააჩნია გადამწოდები - თვალები, ყურები და გრძნობის სხვა ორგანოები, რომელთა საშუალებითაც ის აღიქვამს გარემოს და ზემოქმედებს ამ გარემოზე ხელებით, ფეხებით და სხეულის სხვა ნაწილებით, რომლებიც წარმოადგენენ მის შემსრულებელ მექანიზმებს. რობოტს, რომელიც ასრულებს აგენტის როლს, გადამწოდებად შეიძლება ჰქონდეს ვიდეოკამერა, ინფრაწითელი მანძილმზომი და სხვა. მის შემსრულებელ მექანიზმებს კი წარმოადგენენ მოტორები, მექანიკური ხელი და სხვა. პროგრამული უზრუნველყოფაც შეიძლება წარმოვიგინოთ ინტელექტუალური აგენტის როლში. ამ შემთხვევაში გადამწოდის როლს ასრულებს კლავიატურა და ინფორმაციის შეყვანის სხვა საშუალებები, ხოლო გარემოზე ზემოქმედება გამოიხატება მონაცემების გამოყვანით ეკრანზე, ფაილზე ჩაწერით და ასე შემდეგ.

ტერმინს **აღქმა** ვიყენებთ იმისათვის, რომ აღვნიშნოთ სენსორული ინფორმაციის მიღება დროის ნებისმიერ კონკრეტულ მომენტში. **აღქმათა აქტების მიმდევრობა** ეწოდება იმი სასრულ ისტორიას, რაც საერთოდ როდესმე აღიქვა აგენტმა. ზოგადად **აგენტის მიერ დროის ნებისმიერ მომენტში არჩეული ქმედება შეიძლება დამოკიდებული იყოს აღქმათა აქტების მთელმიმდევრობაზე დასაწყისიდან ამ მომენტამდე**. თუ შესაძლებელია დავადგინოთ რომელ ქმედებას აირჩევს აგენტი ნებისმიერ შესაძლო აღქმათა აქტების მიმდევრობაზე, მაშინ შესაძლებელია აგენტის მეტ-ნაკლებად ზუსტი განსაზღვრა. მათემატიკურად ეს ნიშნავს, რომ რომელიღაც აგენტის ქცევა შეიძლება აღიწეროს **აგენტის ფუნქციის** საშუალებით, რომელიც აღქმის აქტების ნებისმიერ კონკრეტულ მიმდევრობას ასახავს რომელიმე ქმედებაზე.

აგენტის ფუნქციის დასადგენად შესაძლებელია შევადგინოთ ცხრილი, რომელიც აღწერდა ამ ფუნქციას, მაგრამ ეს ცხრილი იქნება პრაქტიკულად უსასრულო, თუ არ დავადგენთ აღქმათა აქტების მიმდევრობის მაქსიმალურ ზომას. თავისთავად ასეთი ცხრილის დადგენა არაა ცუდი, მაგრამ ეს იქნება აგენტის გარეგნული აღწერა. შინაგანი აღწერა მდგომარეობს იმის განსაზღვრაში, თუ აგენტის რა ფუნქცია ხორციელდება **აგენტის პროგრამის** საშუალებით. აგენტის ფუნქციისაგან განსხვავებით, რომელიც წარმოადგენს აგენტის აბსტრაქტულ, მათემატიკურ აღწერას, აგენტის პროგრამა არის კონკრეტული რეალიზაცია, რომელიც მოქმედებს აგენტის არქიტექტურულ ჩარჩოებში.

განვიხილოთ ძალიან მარტივი მაგალითი: დაუშვათ გარე სამყარო შედგება მხოლოდ ორი ( და ) ოთახისაგან, რომლებშიც დგას ოთხკუთხა და სამკუთხა პირამიდები. ამ გარემოში შეიძლება იმყოფებოდეს აგენტი, რომელსაც შეუძლია აღიქვას რომელ ოთახში იმყოფება თვითონ, დგას თუ არა ამ ოთახში პირამიდები და გაარჩიოს ერთმანეთისაგან ოთხკუთხა და სამკუთხა პირამიდები. შემსრულებელი მექანიზმების საშუალებით აგენტს შეუძლია იმოძრაოს გარემოში, აიღოს და გადაიტანოს ერთი ადგილიდან მეორეზე პირამიდები. აგენტის მიზანი ოთახში შეაგროვოს მხოლოდ ოთხკუთხა და ოთახში მხოლოდ სამკუთხა პირამიდები. ასეთი აგენტის ერთი უმარტივესი ფუნქცია შეიძლება მდგომარეობდეს შემდეგში:



სურ. 5 გარემო, რომელშიც მოქმედებს აგენტი

თუ არის ოთახში სამკუთხა პირამიდა, გადაიტანოს ის ოთახში, თუ არ არის ოთახში სამკუთხა პირამიდა, გადავიდეს ოთახში და ჩაატაროს იგივე პროცედურა ოთხკუთხა პირამიდებისათვის. შეგვიძლია შევადგინოთ ასეთი ფუნქციის ნაწილობრივი ცხრილიც თუ

არის A ოთახში სამკუთხა პირამიდა, გადაიტანოს ის B ოთახში, თუ არ არის A ოთახში სამკუთხა პირამიდა, გადავიდეს B ოთახში და ჩაატაროს იგივე პროცედურა ოთხკუთხა პირამიდებისათვის.

შეგვიძლია შევადგინოთ ასეთი ფუნქციის ნაწილობრივი ცხრილიც

აღქმათა ქმედება	აქტების	მიმდევრობა
[A, არ არის სამკუთხა პირამიდა] B ოთახში		გადადი
[A, არის სამკუთხა პირამიდა] B ოთახში		გადაიტანვის
[B, არ არის ოთხკუთხა პირამიდა] A ოთახში		გადადი
[B, არის ოთხკუთხა პირამიდა] A ოთახში		გადაიტანე
[A, არ არის სამკუთხა პირამიდა],[A,არ არის სამკუთხა პირამიდა]გადადი B ოთახში		
.....		
.....		
[ , არის სამკუთხა პირამიდა],[ , არ არის სამკუთხა პირამიდა] , [ , არ არის სამკუთხა პირამიდა] გადადი B ოთახში		

როგორც ამ ცხრილიდან ჩანს, შესაძლებელია მისი სხვადასხვანაირად შევსება, რაც თავისთავად გამოიწვევს აგენტის სხვადასხვა ქცევას. მაშინ ცხადია დგება კითხვა თუ როგორ უნდა შევავსოთ ცხრილი, რომ აგენტის ქცევა იყოს ინტელექტუალური? აქ შემოდის ინტელექტუალური ქცევისადმი ასეთი მიდგომის ყველაზე არსებითი ცნება – **რაციონალურობის** ცნება. **რაციონალური აგენტი** არის ისეთი აგენტი, რომელიც ასრულებს სწორ ქმედებებს, ანუ მის ფუნქციათა ცხრილში ყოველი ჩანაწერი არის სწორი. ცხადია რომ

სწორი ქმედებების შესრულება უნდა სჯობდეს არასწორი ქმედებების შესრულებას, მაგრამ რომელია სწორი და რომელი არასწორი ქმედება? ამ პრობლემიდან თავის დასაღწევად შემოგვაქვს კიდევ ერთი არაფორმარული ცნება - აგენტის **წარმატებული** ფუნქციონირების ცნება. იმისათვის კი, რომ შევაფასოთ, წარმატებულია თუ არა აგენტის ფუნქციონირება, საჭიროა რაიმე ფორმალური კრიტერიუმები, თორემ ყველა ეს განმატებები იქნება ცარიელი წყლის ნაყვა.

მაშასადამე, შეიძლება ვთქათ, რომ გარემოს, გადამწოდების, შემსრულებელი მექანიზმებისა და წარმატების კრიტერიუმების აღწერა წარმოადგენს იმ ამოცანის სრულ სპეციფიკაციას, რომლის გადაჭრაც უხდება აგენტს. როდესაც ჩვენ გვექნება ყველა ეს კომპონენტი, შევძლებთ უფრო ზუსტად განვმარტოთ რა იგულისხმება რაციონალურობის ქვეშ.

აგენტის წარმატებული ქცევის შეფასების კრიტერიუმებს წარმოადგენენ მისი **მუშაობის მახასიათებლები**. მას შემდეგ, რაც აგენტს მოვათავსებთ გარემოში, ის გამოიმუშავებს ქმედებათა მიმდევრობას მიღებულ აღქმათა შესაბამისად. ქმედებათა ეს მიმდევრობა აიძულებს გარემოს გაიაროს რაღაც მდგომარეობები. თუ ეს მიმდევრობა შეესაბამება სასურველს, მაშინ ცხადია აგენტი მუშაობს კარგად. შეესაბამება თუ არა მიღებული შედეგი სასურველს? ამის შესახებ შეიძლებოდა გვეკითხა თვით აგენტისათვის, მაგრამ ზოგიერთი აგენტი საერთოდ ვერ შესძლებს ამ კითხვაზე პასუხის გაცემას, ხოლო ზოგიერთმა მათგანმა შეიძლება ან თავი მოიტყუოს ან ჩვენ მოგვატყუოს რომ სწორედ ამ მიზნის მიღწევა სურდა. (გავიხსენოთ მელიის და ყურძნის არაკი). აქედან გამომდინარე განსაკუთრებით მნიშვნელოვანია თუ რა კრიტერიუმებით შევაფასებთ აგენტის მიერ შესრულებულ სამუშაოს. დაეუშვათ, ჩვენი მაგალითის შემთხვევაში კრიტერიუმად მივიღეთ გადატანილი პირამიდების რაოდენობა, რომელსაც აგენტი გადაიტანს ერთი ოთახიდან მეორეში რვა საათიანი სამუშაო დღის განმავლობაში. მაშინ ჩვენი აგენტი შეიძლება ასე მოიქცეს: გადაიტანოს სამკუთხა პირამიდები ოთახიდან ოთახში და ოთხკუთხა პირამიდები -დან -ში, შემდეგ კვლავ გადმოიტანს -დან -ში სამკუთხაპირამიდები და ასე შემდეგ. ჩვენი კრიტერიუმის მიხედვით

ის იქნება წარმატებული აგენტი და უნდა მიიღოს შესაბამისი ანაზღაურება. ამ მაგალითიდან კარგად ჩანს, რომ ასეთ მარტივ შემთხვევაშიც კი აუცილებელია აგენტის ქმედების შეფასების კრიტერიუმების სწორად შერჩევა, თუ გვინდა რომ მივიღოთ მართლაც რაციონალური ქმედება დასახული მიზნის მისაღწევად.

აგენტის ქმედების რაციონალურობასდროის ნებისმიერ მომენტში განსაზღვრავს შემდეგი ოთხ ფაქტორი:

მუშაობის მახასიათებლები, რომლებიც განსაზღვრავენ წარმატების კრიტერიუმებს;

აგენტის ცოდნა გარემოს შესახებ, რომელიც მან მიიღო ადრე;

ქმედებები, რომელთა შესრულებაც შეუძლია აგენტს;

აგენტის მიერ აღქმული აქტების მიმდევრობა დროის მოცემულ მომენტამდე.

ყველა ამ ფაქტორის გათვალისწინებით შეგვიძლია ჩამოვაყალიბოთ რაციონალური აგენტის განმარტება. აღქმის აქტების ყველა შესაძლო მიმდევრობებისათვის აგენტმა უნდა ამოირჩიოს ის ქმედება, რომელიც, როგორც მოსალოდნელია, მოახდენს მისი მუშაობის მახასიათებლების მაქსიმალურად მოცემული აქტების მიმდევრობით მიღებული ფაქტებისა და აგენტში ჩამონტაჟებული მთელი ცოდნის გათვალისწინებით.

ძალიან მნიშვნელოვანია განვასხვავოთ ერთმანეთისაგან რაციონალურობა და ყველაფრის ცოდნა. ყველაფრის მცოდნე აგენტმა ფაქტობრივად იცის თავისი ქმედების შედეგი (რაც ზოგიერთ გარემოში შეუძლებელია) და შეუძლია იმოქმედოს შესაბამისად. განვიხილოთ მაგალითი. დავუშვათ, აგენტმა სეირნობის დროს გადაწყვიტა გადავიდეს ქუჩის მეორე მხარეს. ის ხედავს რომ ქუჩაში არ მოძრაობს არცერთი ავტომობილი და როგორც რაციონალურმა აგენტმა, დაიწყო ქუჩის გადაკვეთა. ამ მომენტამდე ცოტა ადრე კი 10 000 მეტრის სიმაღლეზე თვითმფრინავს მოსწყდა კარები, რომელიც დაეცა ჩვენს აგენტს. იყო თუ არა აგენტის ქმედება რაციონალური? ალბათ არცერთი ადამიანი ქუჩაზე გადასვლის დროს არ იხედება მაღლა და არ ეძებს საშიშროებას ზევიდან. ეს მაგალითი გვიჩვენებს თუ რა განსხვავებაა რაციონალურ ქმედებასა და ყველაფრის ცოდნას შორის.

მეორე შემთხვევაში, თუკი აგენტი ისე გადადის ქუჩაზე, რომ წინასწარ არ გაიხედავს მარჯვნივ და მარცხნივ, მის მიერ აქამდე აღქმული აქტების მიმდევრობა ვერ უკარნახებს მას, რომ დიდი სისწრაფით უახლოვდება სატვირთო მანქანა. რაციონალურ აგენტს უნდა შეეძლოს ისეთ ქმედებათა შესრულება, რომელიც საშუალებას მისცემს მას შესაძლებლობის ფარგლებში წინასწარ განსაზღვროს მოსალოდნელი შედეგი. ანუ მას უნდა შეეძლოს **შეაგროვოს ინფორმაცია** გარემოს შესახებ და აგრეთვე გამოიკვლიოს **უცნობი სიტუაციები**.

### 13 ინტელექტუალურ აგენტთა ტიპები ვიდუოთამაშებში

რაციონალური აგენტის განმარტება გულისხმობს აგრეთვე, რომ აგენტი არა მარტო უნდა აგროვებდეს ინფორმაციას გარემოს შესახებ, არამედ **სწავლობდეს** კიდევ. სწავლის ქვეშ იგულისხმება რომ აგენტის საწყისი კონფიგურაცია შეიძლება შეიცავდეს რაიმე წინასწარ ცოდნას გარემოს შესახებ, მაგრამ თანდათანობით, აგენტის გამოცდილებასთან ერთად ეს ცოდნა იცვლება და განიცდის მოდიფიკაციას. არსებობს განსაკუთრებული შემთხვევები, როდესაც გარემო მთლიანად ცნობილია. ასეთ შემთხვევაში აგენტს აღარ სჭირდება არც ინფორმაციის შეგროვება და არც დასწავლა, მაგრამ ასეთი აგენტი იქნება ძალიან უსუსური. საკმარისია სულ პატარა ცვლილება გარემოში, რომ აგენტის ქმედებები უკვე აღარ იქნება რაციონალური.

როდესაც ჩვენ ვიხილავდით აგენტს, რომელსაც პირამიდები გადაჰქონდა ერთი ოთახიდან მეორეში, საჭირო გახდა მისი მუშაობის მახასიათებლების, გარემოს, გადამწოდებისა და შემსრულებელი მექანიზმების განსაზღვრა. გავაერთიანოთ ამ ფაქტორების აღწერა **პრობლემური გარემოს** ცნებაში. განვიხილოთ ახალი მაგალითი – ტაქსის ავტომატიზებული მძღოლის შექმნის პრობლემა. მანქანის ტარების პრობლემის სრული გადაჭრა ძალიან რთული ამოცანაა, რადგანაც უსაზღვროა იმ ყველანაირ სიტუაციათა რაოდენობა, რომლებიც შეიძლება წარმოიშვას მანქანის ტარების დროს და ამიტომ ყველაფერს წინასწარ ვერ გაითვალისწინებ.

პირველ რიგში განვსაზღვროთ **მუშაობის მახასიათებლები**, რომლებიც განსაზღვრავენ ტაქსის ავტომატიზებული მძღოლის

მოქმედებას მთლიანობაში: სასურველია წარმატებით მიაღწიოს დანიშნულების ადგილს; მოახდინოს საწვავის, ავტომანქანის დეტალების ცვეთის, მგზავრობის ხანგრძლივობის და/ან მგზავრობის ღირებულების მინიმიზაცია; იმოძრაოს ისე, რომ არ დაარღვიოს მოძრაობის წესები და არ შეუქმნას პრობლემები სხვა მძღოლებს; შეუქმნას მაქსიმალური უსაფრთხოება და კომფორტი მგზავრებს; მიაღწიოს მაქსიმალურ მოგებას. ცხადია, რომ ამ მიზნებიდან ზოგიერთი არაა მკაფიოდ ჩამოყალიბებული და აგრეთვე ზოგიერთი მათგანი მოდის წინააღმდეგობაში სხვებთან, ამიტომ საჭიროა რაღაც კომპრომისის გამონახვა.

განვიხილოთ, რისგან შედგება **გარემო, რომელშიც მოძრაობს მანქანა**. პირველ რიგში ესაა გზები დაწყებული ავტობანებიდან დამთავრებული სოფლის ორღობეებით. ამ გზებზე მძღოლს შეიძლება შეხვდნენ სხვა სატრანსპორტო საშუალებები, ფეხით მოსიარულენი, ცხოველები, მუშები, რომლებიც აწარმოებენ სარემონტო სამუშაოებს, საპატრულო პოლიცია. ტაქსის მძღოლს საქმე აქვს პოტენციურ ან რეალურ მგზავრებთან. მან უნდა გაითვალისწინოს აგრეთვე მეტეოროლოგიური პირობები. ავტომატიზებულ მძღოლს დასჭირდება **შემსრულებელი მექანიზმები**, რომელთა საშუალებითაც ის შესძლებს მართოს მანქანა და იქონიოს ურთიერთობა როგორც მგზავრებთან, ასევე მოძრაობის სხვა მონაწილეებთან. ამ მიზნების მისაღწევად მას დასჭირდება **გადამწოდები**. მათ შორის ერთი ან რამდენიმე ვიდეოკამერა, სპიდომეტრი, ოდომეტრი, აქსელომეტრი და ყველა ის სისტემები, რომლებიც შეატყობინებენ მანქანის ტექნიკურ გამართულობას. გარდა ამისა მას დასჭირდება აგრეთვე გარემოს ელექტრონული რუკა და ნავიგაციის გლობალური თანამგზავრული სისტემა თავისი ადგილმდებარეობის განსასაზღვრავად. აი ასეთი შეიძლება იყოს ავტომატიზებული მძღოლის პრობლემური გარემო. საერთოდ ცხადია, რომ რაც უფრო რთულია ამოცანა, რომელიც დგას აგენტის წინაშე, მით უფრო რთულია პრობლემური გარემოც. შევეცადოთ, დავახასიათოთ, როგორი შეიძლება იყოს პრობლემური გარემო.



**1. მთლიანად დაკვირვებადი ან ნაწილობრივ დაკვირვებადი.** თუკი გადამწოდები აგენტს აწვდიან სრულ ინფორმაციას გარემოს შესახებ დროის ნებისმიერ მომენტში, ასეთი გარემო იქნება მთლიანად დაკვირვებადი. სრული ინფორმაციის ქვეშ ვგულისხმობთ იმ ინფორმაციას, რომელიც აგენტს სჭირდება გადაწყვეტილების მისაღებად. ასეთი გარემო მოსახერხებელია იმით, რომ აგენტს არ სჭირდება შეინახოს თავის მეხსიერებაში გარემოს მდგომარეობა. გარემო შეიძლება იყოს ნაწილობრივ დაკვირვებადი იმ შემთხვევაში, როდესაც წარმოიშვება ხმაური ან როდესაც გადამწოდები ვერ აწვდიან ზუსტ ინფორმაციას აგენტს. ნაწილობრივ დაკვირვებადი იქნება გარემო მაშინაც, როდესაც აგენტს არ გააჩნია ინფორმაცია გარემოს რომელიმე მახასიათებელზე. მაგალითად, ჩვენს მიე განხილულ აგენტს, რომელსაც გადააქვს პირამიდები ერთი ოთახიდან მეორეში, გააჩნია მხოლოდ ლოკალური გადამწოდი, და ამიტომ არ შეუძლია განსაზღვროს არის თუ არა სამკუთხა პირამიდა მეორე ოთახში, ხოლო ავტომატიზებულ მძღოლს არ გააჩნია ინფორმაცია რა მანევრის შესრულებას აპირებენ მოძრაობის სხვა მონაწილენი.

**2. დეტერმინირებული ან სტოქასტური.** თუ გარემოს შემდეგი მდგომარეობა მთლიანად განსაზღვრულია მიმდინარე მდგომარეობით და აგენტის ქმედებით, მაშინ გარემო დეტერმინირებულია, წინააღმდეგ შემთხვევაში კი სტოქასტური. მთლიანად დაკვირვებად გარემოში აგენტს არ უხდება იმოქმედოს განუზღვრელობის პირობებში, მაგრამ თუ გარემო ნაწილობრივ დაკვირვებადია, მაშინ იქნება შთაბეჭდილება, რომ გარემო სტოქასტურია, რადგან აგენტს არ შეუძლია დააკვირდეს მის ყველა მახასიათებელს. ამიტომ ზოგჯერ უფრო მოსახერხებელია ვთქვათ, რომ გარემო დეტერმინირებულია ან სტოქასტურია აგენტის თვალსაზრისით. აშკარაა, რომ ასეთი განმარტების შემთხვევაში აგენტ – ტაქსისტის გარემო სტოქასტურია, რადგან მას არ შეუძლია ზუსტად განსაზღვროს სხვა მონაწილეთა ქცევა, ან შეიძლება მის ავტომობილს მოულოდნელად რაიმე გაუფუჭდეს.

**3. ეპიზოდური ან მიმდევრობითი.** ეპიზოდურ პრობლემურ გარემოში აგენტი აღიქვამს კონკრეტულ ეპიზოდს, დებულობს შესაბამის გადაწყვეტილებას და მოქმედებს. არსებითია, რომ მიღებული

გადაწყვეტილება არანაირ გავლენას არ ახდენს შემდეგ ეპიზოდზე. მაგალითად, დავუშვათ აგენტი ახდენს დეტალების გადაჩვენას კონვეირზე. აგენტი აკვირდება დეტალს რომელიც მის წინაშეა და თუ მას გააჩნია დეფექტი, გადადებს მას კონვეირიდან. ეს გადაწყვეტილება ვერანაირად ვერ განსაზღვრავს იქნება თუ არა შემდეგი დეტალი დეფექტით. ასეთ გარემოში აგენტს უადვილდება მოქმედება, რადგან არაა საჭირო გაითვალისწინოს რა გავლენას მოახდენს მისი გადაწყვეტილება პრობლემურ გარემოზე. განსხვავებით ასეთი გარემოსგან, მიმდევრობით გარემოში ყოველი გადაწყვეტილება მოქმედებს პრობლემური გარემოს შესაძლო მდგომარეობაზე და აგენტს უხდება ამის გათვალისწინება. ასეთი გარემოს მაგალითია ჭადრაკის თამაში.

**4. სტატიკური ან დინამიური.** თუ პრობლემური გარემო შეიძლება შეიცვალოს იმ პერიოდში, როდესაც აგენტი ირჩევს თავის ქმედებას, ასეთ გარემოს ეწოდება ამ აგენტისათვის დინამიური გარემო, წინააღმდეგ შემთხვევაში პრობლემური გარემო სტატიკურია. სტატიკურ გარემოში მოქმედება უფრო მარტივია, რადგან აგენტს არ სჭირდება გადაწყვეტილების მიღების მომენტში აკვირდებოდეს გარემოს და აგრეთვე არაა შეზღუდული დროში. თუ დროთა განმავლობაში გარემო არ იცვლება, მაგრამ იცვლება აგენტის მუშაობის მახასიათებლები, მაშინ ასეთ გარემოს უწოდებენ **ნახევრად დინამიურ** გარემოს.

**5. დისკრეტული ან უწყვეტი.** განსხვავება დისკრეტულ და უწყვეტ გარემოს შორის შეიძლება ეხებოდეს გარემოს მდგომარეობებს, დროის აღრიცხვის მეთოდებს და აგრეთვე აგენტის მიერ გარემოს აღქმასა და ქმედებებს. მაგალითად, ჭადრაკის თამაშის გარემო დისკრეტულია როგორც გარემოს მდგომარეობებით, აგრეთვე აღქმისა და ქმედებების მიხედვით, განსხვავებით ტაქსის ტარების გარემოსგან, რომელშიც გარემოს მდგომარეობები, დრო, აღქმა და ქმედებებიც უწყვეტია.

**6. ერთაგენტიანი და მულტიაგენტიანი.** განსხვავება ერთაგენტიან და მულტიაგენტიან გარემოებს შორის ერთი შეხედვით ძალიან მარტივია. მაგალითად, კროსვორდის ამოხსნის გარემო ერთაგენტიანი გარემოა, ხოლო ჭადრაკის თამაშის გარემო კი ორაგენტიანი, თუმცა აქაც არსებობს გარკვეული ნიუანსები. ჩვენ აღვწერეთ რა ნიშნების

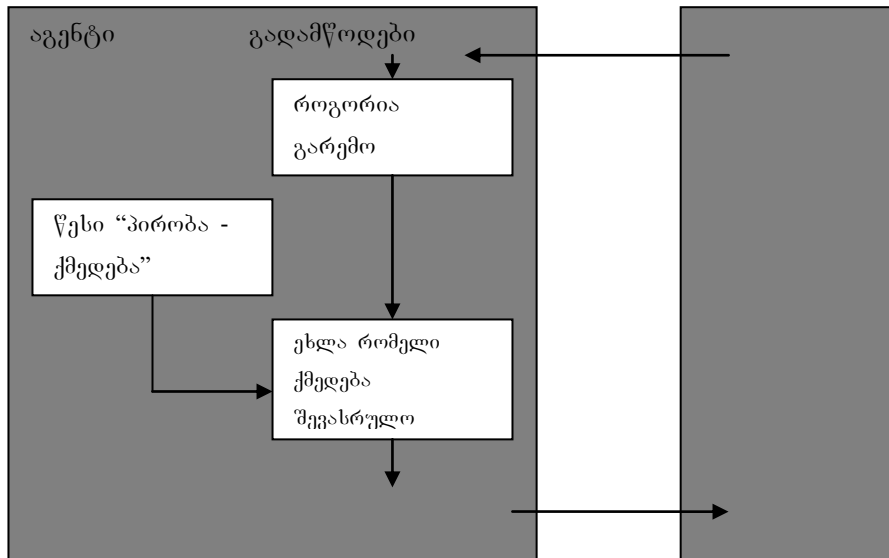
საფუძველზე შეიძლება ესა თუ ის არსი **შეიძლება** ჩავთვალოთ აგენტად, მაგრამ არ მიგვითითებია რომელი არსი **უნდა** განვიხილოთ როგორც აგენტი. უნდა ჩათვალოს თუ არა ტაქსის მძღოლმა სხვა სატრანსპორტო საშუალებების მძღოლები აგენტებად, თუ მოექცეს მათ როგორც უბრალოდ სტოქასტურად მოძრავ ობიექტებს. არსებითი განსხვავება მდგომარეობს იმაში, უნდა ჩათვალოს თუ არა ობიექტის ქცევა ისეთ ფაქტორად, რომელიც გავლენას ახდენს მისი მუშაობის მახასიათებლებზე. ჭადრაკის თამაშში მოწინააღმდეგე ცდილობს რა მოახდინოს თავისი მუშაობის მახასიათებლების მაქსიმიზაცია, ავტომატურად იწვევს აგენტის მუშაობის მახასიათებლების მინიმიზაციას. ამიტომ ჭადრაკის თამაშში არის **კონკურენტული** მულტიაგენტური გარემო. ტაქსის ტარების გარემოში კი დაჯახებათა თავიდან აცილება იწვევს ყველა აგენტის მუშაობის მახასიათებლების მაქსიმიზირებას, ამიტომ ასეთი გარემო არის ნაწილობრივ **კოოპერაციული გარემო** (ნაწილობრივ იმიტომ რომ სხვაპარამეტრებით ის შეიძლება იყოს კონკურენტული).

#### 14. აგენტთა ტიპები.

აქამდე ჩვენ აგენტების თვისებებს განვიხილავდით მხოლოდ მათი ქცევის ანალიზის მიხედვით. გადავიდეთ მათი ფუნქციონირების აღწერაზე. ხელოვნური ინტელექტის ამოცანა მდგომარეობს **აგენტის პროგრამის** შექმნაში, რომელიც ახდენს აგენტის ფუნქციის რეალიზაციას. იგულისხმება, რომ ეს პროგრამა უნდა მუშაობდეს გარკვეულ გამომთვლელ მოწყობილობაში ფიზიკური გადამწოდებით და შემსრულებელი მექანიზმებით. ამ კომპონენტებს მთლიანობაში უწოდებენ **არქიტექტურას**. აგენტის სტრუქტურა შეიძლება წაროვადგინოთ შემდეგი სახით:

$$\text{აგენტის სტრუქტურა} = \text{არქიტექტურა} + \text{პროგრამა}$$

ცხადია, რომ არჩეული პროგრამა უნდა შეესაბამებოდეს აგენტის არქიტექტურას. ასე მაგალითად, თუ პროგრამა გამოიმუშავებს ისეთ რეკომენდაციებს, როგორცაა “სიარული”, არქიტექტურაში გათვალისწინებული უნდა იყოს შესაბამისი მექანიზმი.



სურ. 6 რეფლექსური აგენტის სტრუქტურა

### 1.5. მარტივი რეფლექსური აგენტი.

ასეთი აგენტები ირჩევენ ქმედებას მხოლოდ მიმდინარე აღქმის აქტის საფუძველზე და არ ითვალისწინებენ წინაისტორიას. განხილული აგენტი, რომელსაც ერთი ოთახიდან მეორეში გადააქვს პირამიდები წარმოადგენს

სწორედ ასეთ აგენტს. მის პროგრამას აქვს შემდეგი სახე:

რეფლექსური აგენტის **ფუნქცია** [მდგომარეობა, სტატუსი]

**შეასრულე ქმედება**

**თუ** მდგომარეობა = ოთახი

**თუ** სტატუსი = არის სამკუთხა პირამიდა მაშინ **შეასრულე**

გადაიტანე -ში

**თუ არა** გადადი ოთახში

**თუ** მდგომარეობა = ოთახი

**თუ** სტატუსი = არის ოთხკუთხა პირამიდა მაშინ **შეასრულე**

გადაიტანე -ში

**თუ არა** გადადი ოთახში

როგორც ვხედავთ, ეს პროგრამა გაცილებით პატარაა იმ ცხრილთან შედარებით, რომელიც აღწერდა აგენტის ფუნქციას. ეს შემცირება განპირობებულია იმით, რომ უგულებელყოფილია აღქმათა აქტების

ისტორია, რის გამოც ყველა შესაძლო ვარიანტების რიცხვი დავიდა ოთხამდე.

რეფლექსური აგენტის მოქმედების საფუძველს წარმოადგენს პრინციპი: თუ სრულდება რაიმე პირობა, შეასრულე შესაბამისი ქმედება. შემოკლებით ეს შეგვიძლია აღვნიშნოთ როგორც წესი “პირობა – ქმედება”. ასეთი აგენტები მრავლად გვხვდება როგორც ტექნიკურ სისტემებში, ასევე ცოცხალ ორგანიზმებშიც. მათი ფუნქციონირება ძალიან მარტივია, მაგრამ მათ გააჩნიათ ძალიან შეზღუდული ინტელექტი (თუ გააჩნიათ საერთოდ). რეფლექსური აგენტის გადაწყვეტილებები მხოლოდ მაშინ იქნება სწორი, როდესაც მხოლოდ მიმდინარე აღქმის აქტი საკმარისია სწორი გადაწყვეტილების მისაღებად, ანუ თუ გარემო არის მთლიანად დაკვირვებადი. საკმარისია გარემო გახდეს ნაწილობრივ დაკვირვებადი და აგენტის ქმედებები აღარ იქნება სიტუაციის შესაბამისი. მაგალითად, განვიხილოთ ტაქსის ტარების პრობლემური გარემო. თუ წინ მიმავალი მანქანა იწყებს დამუხრუჭებას და ინთება მისი სამუხრუჭე ფარები, მძღოლმა ეს უნდა შენიშნოს და თვითონაც დაიწყოს დამუხრუჭება. ანუ შეიძლება ჩამოვაცალიბოთ “პირობა – ქმედების” შემდეგი წესი:

**თუ** ავტო წინ – ამუხრუჭებს **მაშინ** დაიწყე დამუხრუჭება დაუშვავთ, ამ პირობის აღქმა ხდება მხოლოდ მიმდინარე აღქმის აქტით. რადგანაც ზოგიერთ ძველი მარკის ავტომობილს სამუხრუჭე ფარები შეთავსებული აქვს სხვა სასიგნალო ფარებთან, მიმდინარე აღქმის ფაქტით შეუძლებელია დავადგინოთ, ამუხრუჭებს თუ არა წინ მიმავალი მანქანა. მარტივი რეფლექსური აგენტი, რომელიც მისდევს ამ მანქანას, ან დაამუხრუჭებს მაშინ როდესაც ამის აუცილებლობა არ იქნება, ან რაც უფრო უარესია, საერთოდ არ დაამუხრუჭებს.

### **1.6. მოდელზე დაფუძნებული რეფლექსური აგენტები.**

ნაწილობრივ დაკვირვებადი გარემოს შემთხვევაში კარგი იქნება თუ აგენტი აკვირდება აგრეთვე მიმდინარე აღქმის აქტის ისტორიას. ეს ნიშნავს, რომ აგენტს უნდა გააჩნდეს გარკვეული **შინაგანი მდგომარეობა**, რომელიც დამოკიდებულია აღქმის აქტების ისტორიაზე და ამიტომ ასახავს მიმდინარე მომენტში ზოგიერთ დაუკვირვებად

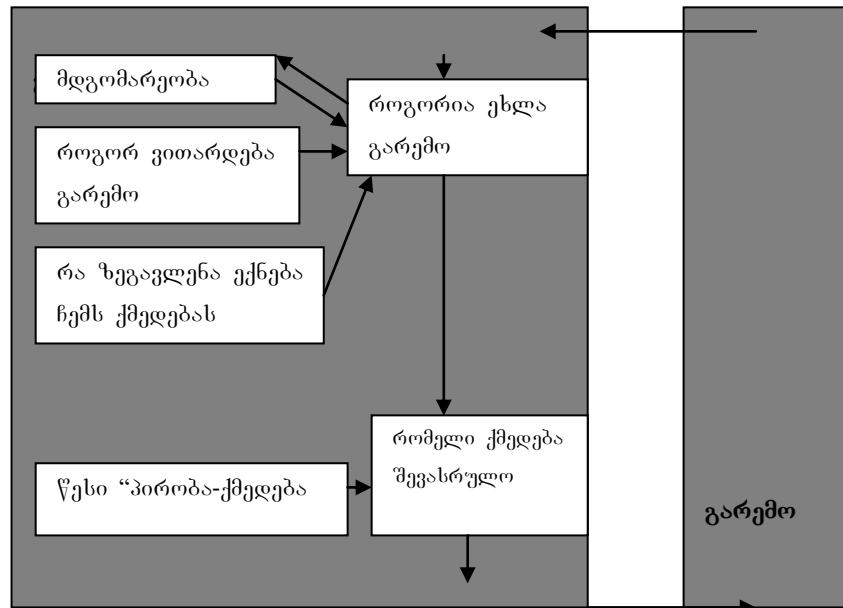
ასპექტს. ზემოთ განხილულ მაგალითში ამისათვის საკმარისია აგენტის შინაგანი მდგომარეობა ინახავდეს მხოლოდ წინა კადრს, რაც მისცემს აგენტს საშუალებას აღმოაჩინოს დამუხრუჭების დაწყების მომენტი.

იმისათვის, რომ უზრუნველყოთ დროთა განმავლობაში ამ შინაგანი ინფორმაციის განახლება, საჭიროა აგენტის პროგრამაში შევიტანოთ ორი სახის ცოდნა. პირველი, აგენტმა უნდა იცოდეს როგორ შეიძლება შეიცვალოს გარემო მისი ჩარევის გარეშე და მეორე, რა ცვლილებებს გამოიწვევს მისი ქმედებები გარემოში. ასეთ ცოდნას უწოდებენ **გარემოს მოდელს**,

ხოლო აგენტს რომელშიც გამოიყენება ასეთი მოდელი, უწოდებენ **მოდელზე დაფუძნებულ აგენტს**.

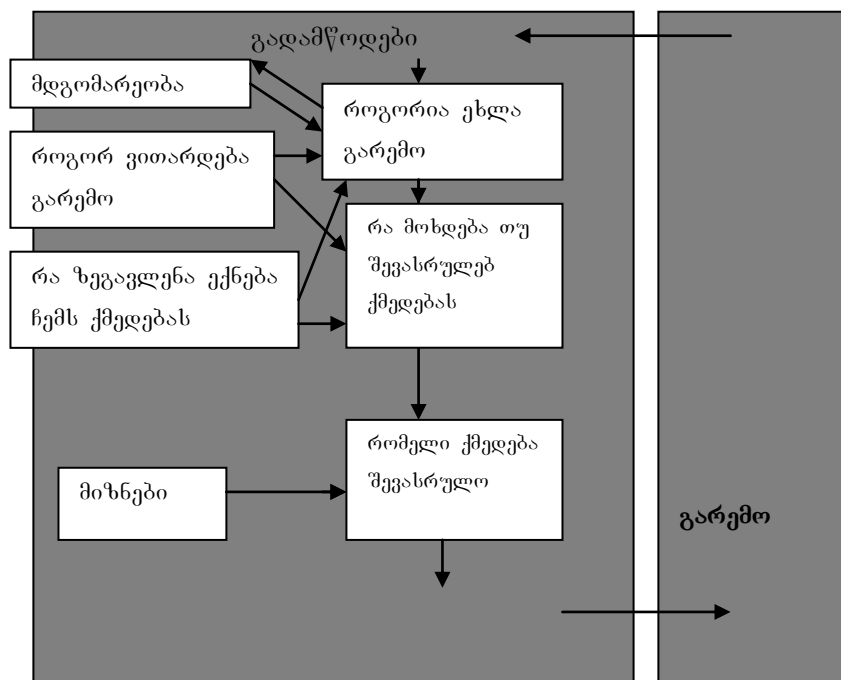
### 1.7. მიზანზე და მოდელზე დაფუძნებული აგენტები.

გარემოს ცოდნა ხშირად არ არის საკმარისი იმისათვის რომ აგენტმა მიიღოს სწორი გადაწყვეტილება. თუ ტაქსი დგას გზაჯვარედინზე, სწორი გადაწყვეტილების მისაღებად აუცილებელია აგენტმა იცოდეს თუ სად უნდა მივიდეს ის. აგენტს სჭირდება არა მარტო გარემოს მიმდინარე მდგომარეობის ცოდნა, არამედ ინფორმაცია **მიზნის** შესახებ, რომელიც აღწერს პრობლემური გარემოს სასურველ სიტუაციებს. აგენტის პროგრამას შეუძლია ამ ინფორმაციისა და შესაძლო ქმედებათა შედეგების კომბინირება ისეთი ქმედების ამოსარჩევად, რომელიც შესაძლებლობას მისცეს მიაღწიოს მიზანს. ზოგჯერ მიზნის მიღწევა ძალიან ადვილია, რადგან მის მისაღწევად საკმარისია ერთი ქმედება. უფრო რთულ და რეალურ შემთხვევებში, ეს ასე არ არის. მაშინ აგენტის პროგრამაში გათვალისწინებული უნდა იყოს ისეთი პროცედურები, რომლებიც მოახდენენ მიზნის **ძებნას** და ქმედებათა მიმდევრობის **დაგეგმვას** ამ მიზნის მისაღწევად.



სურ. 7 მოდელზე დაფუძნებული რეფლექსური აგენტი

ასეთ შემთხვევაში აგენტის გადაწყვეტილების ძიების პროცედურა არსებითად განსხვავდება რეფლექსური აგენტის გადაწყვეტილების ძიების პროცედურისაგან, რადგან აცაგენტს უხდება უპასუხოს ორ კითხვას: “რა მოხდება თუ მე მივიღებ ამ გადაწყვეტილებას?” და “მივადწევ თუ არა ამ ქმედების შედეგად მიზანს?”. რეფლექსურ აგენტებში ასეთი ინფორმაცია არაა წარმოდგენილი, რადგან აც მასში ჩამონტაჟებული წესები ამყარებენ პირდაპირ კავშირს გარემოს აღქმასა და ქმედებას შორის.



სურ. 8 მიზანზე და მოდელზე დაფუძნებული აგენტის სტრუქტურა

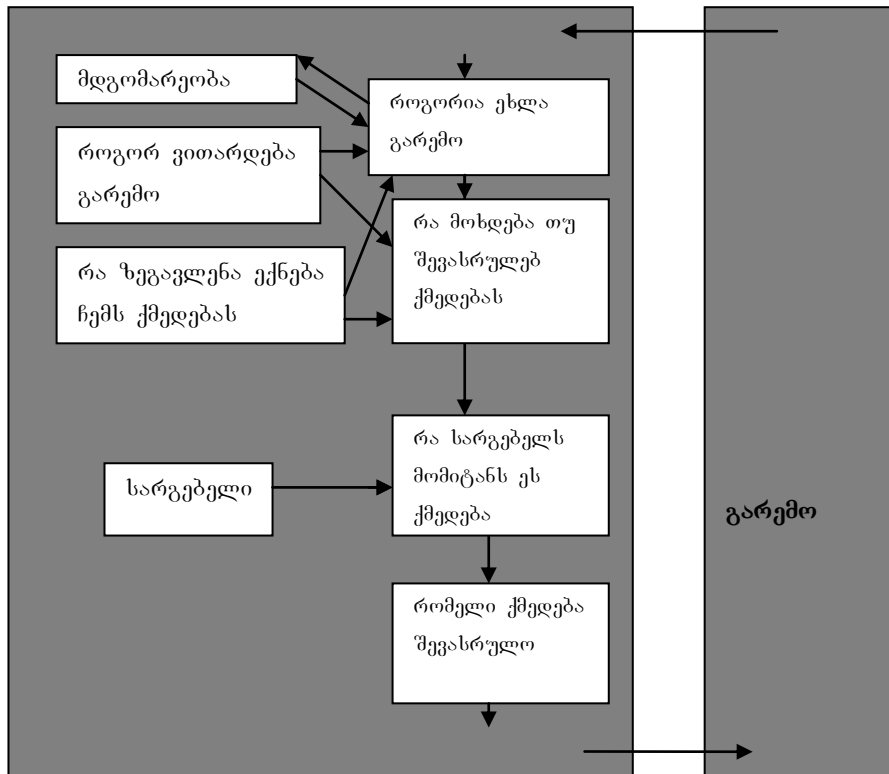
### 1.8. სარგებლიანობაზე დაფუძნებული აგენტები.

ხშირ შემთხვევაში კარგი გადაწყვეტილების მისაღებად საკმარისი არ არის მხოლოდ მიზნის მიღწევა. იგივე ტაქსის მართვის შემთხვევაში დანიშნულების ადგილს შეიძლება მივადწიოთ სხვადასხვა გზით, რომელთაგან ზოგი გზა იქნება უფრო გრძელი და დაგვეხარჯება უფრო მეტი დრო და საწვავი, ზოგი კი უფრო მოკლე და შესაძლებელია დროისა და საწვავის დაზოგვა. ამიტომ საჭიროა აგენტმა ამოიჩინოს ისეთი ქმედებების მიმდევრობა, რომლებიც უკეთესად დააკმაყოფილებენ აგენტის მუშაობის მახასიათებლების კრიტერიუმებს. ის მდგომარეობა პრობლემური გარემოსი, რომელიც უფრო

უკეთესია სხვა მდგომარეობებთან შედარებით აგენტის მუშაობის მახასიათებლების კრიტერიუმების დაკმაყოფილების თვალსაზრისით, განიხილება როგორც აგენტისათვის უფრო სასარგებლო მდგომარეობა.

სარგებლიანობის ფუნქცია ასახავს მდგომარეობას (ან მდგომარეობათა მიმდევრობას) ნამდვილ რიცხვთა სიმრავლეზე, რომელიც შეესაბამება აგენტის მუშაობის კრიტერიუმების დაკმაყოფილების შესაბამის ხარისხს (უნდა აღინიშნოს, რომ ასეთი ასახვის მოძებნა ზოგერთ შემთხვევაში ძალიან რთულია). სარგებლიანობის ფუნქცია საშუალებას გვაძლევს მივიღოთ რაციონალური გადაწყვეტილება იქ სადაც უძლურია მიზნის ფუნქცია. პირველ რიგში როდესაც გვაქვს კონფლიქტური მიზნები (მაგალითად ჩქარი სიარული, თუ მოძრაობის უსაფრთხოება) და მეორე იმ შემთხვევაში, როდესაც გვაქვს რამდენიმე მიზანი, რომელთაგანაც მიიღტვის აგენტი და რომელთაგან არცერთი არ შეიძლება იქნას მიღწეული ზუსტად. ასეთ შემთხვევაში სარგებლიანობის ფუნქცია წარმოადგენს იმ მოსახერხებელ მეთოდს, რომელიც საშუალებას გვაძლევს მიზნების პრიორიტეტების გათვალისწინებით შევაფასოთ რომელიმე მიზნის მიღწევის ალბათობა.





სურ. 9 სარგებლიანობაზე დაფუძნებული აგენტის სტრუქტურა

### 19. სწავლებადი აგენტები.

აქამდე ჩვენ ვიხილავდით აგენტების პროგრამებს, რომლებშიც გამოიყენება გადაწყვეტილების მიღების სხვადასხვა მეთოდები, მაგრამ არაფერი არ გვითქვამს, თუ როგორ უნდა შეიქმნას ასეთი პროგრამები. ჯერ კიდევ ა. ტიურინგმა გაანალიზა ასეთი პროგრამების შექმნის პრობლემა და მივიდა იმ დასკვნამდე, რომ საჭიროა რაიმე პროდუქციული მეთოდი, რომელიც გააადვილებს პროგრამების შექმნას. მანვე ასეთ მეთოდად მიიჩნია სწავლებადი მანქანის შექმნა და შემდეგ მათი სწავლება. დღეს სწორედ ასეთი მიდგომა დომინირებს ხელოვნურ ინტელექტში.

დასწავლადი აგენტის სტრუქტურა შედგება ოთხი კონცეპტუალური კომპონენტისაგან. ყველაზე მნიშვნელოვანია განსხვავება **დასწავლადკომპონენტსა** და **მუშა კომპონენტს** შორის. დასწავლადი კომპონენტი პასუხისმგებელია აგენტის მუშაობის გაუმჯობესებაზე, ხოლო მუშა კომპონენტი უზრუნველყოფს ქმედების შესრულებას. ფაქტობრივად, მუშა კომპონენტი არის ის, რასაც აქამდე განვიხილავდით როგორც აგენტს: ის ღებულობს აღქმით ინფორმაციას

და იღებს გადაწყვეტილებას რა ქმედება უნდა განხორციელდეს. დასწავლადი კომპონენტი იყენებს რა ინფორმაციას, რომელსაც ის იღებს *კრიტიკისსაგან* ეკუთვნის საშუალებით, რომელიც შეიცავს აგენტის მუშაობის შეფასებას, განსაზღვრავს თუ როგორ უნდა შეიცვალოს მუშა კომპონენტი რათა მისი მუშაობა გაუმჯობესდეს.

აგენტის მეოთხე ძირითად კომპონენტს წარმოადგენს **პრობლემათა გენერატორი**, რომლის ამოცანას წარმოადგენს ისეთი ექსპერიმენტების შეთავაზება მუშა კომპონენტისათვის, რომელთა საფუძველზეც შესაძლებელი იქნება ახალი, უფრო უკეთესი ქმედებების აღმოჩენა.

## თავი 2. ცოდნის წარმოდგენა და დამუშავება ვიდეოთამაშების ხელოვნური ინტელექტის სისტემაში

### 2.1. ცოდნის წარმოდგენის აუცილებლობა.

ჩვენ განვიხილეთ ისეთი მიზანზე დაფუძნებული აგენტები, რომლებიც პრობლემათა გადასაწყვეტად იყენებდნენ ძეხვის ალგორითმებს. როგორც ვნახეთ, ასეთი აგენტები არ იძლევიან საშუალებას გადავჭრათ ყველა პრობლემა, რადგანაც ეს ალგორითმები იყო ექსპონენციალური და ამოცანის ზომების ზრდასთან ერთად მათი ეფექტურობა მცირდებოდა. ჩვენ ესეა დავიწყებთ ისეთი აგენტების შესწავლას, რომლებიც მოქმედებენ ცოდნის საფუძველზე (შემოკლებით ცოდნაზე დამყარებულ აგენტი). ცოდნის წარმოდგენა და მსჯელობის პროცესები წარმოადგენენ ხელოვნური ინტელექტის ცენტრალურ საკითხებს. თავისთავად ცხადია, რომ ადამიანებმა ბევრი რამ იციან და გააჩნიათ განსჯის უნარი. ცოდნა და განსჯის უნარი ასევე მნიშვნელოვანია ხელოვნური აგენტებისათვის, რადგანაც უზრუნველყოფს წარმატებული ქცევის ფორმირებას, რომლის მიღწევაც სხვა გზით ალბათ შეუძლებელი იქნებოდა. ასე მაგალითად, თუ ამოცანათა ამომხსნელ აგენტს ეცოდინება თავისი ქმედებების შედეგები, მას შეეძლება წარმატებულად იმოქმედოს რთული გარემოს პირობებშიც და არ ეძებოს გზა ბათუმიდან თბილისამდე ბრმად, როგორც ამას აკეთებდა მიზანზე დამყარებული აგენტი. მაგრამ ამოცანათა ამომხსნელი აგენტების ცოდნა არის ძალიან სპეციფიკური და მოუქნელი. ჭადრაკის მოთამაშე პროგრამას შეუძლია გათვალისწინოს მუხის დასაშვები სვლები (იმ ფერის, რომელი ფერითაც ის თამაშობს), მაგრამ მას არ გააჩნია მრავალი სხვა სასარგებლო ცოდნა, მაგალითად, რომ არც ერთ ფიგურას არ შეუძლია ერთდროულად იდგეს ორ უჯრაზე. ცოდნაზე დამყარებულ აგენტებს შეუძლიათ ძალიან ზოგად ფორმაში გამოხატული ცოდნის გამოყენება, ამ ცოდნის კომბინირება და რეკომბინირება სხვადასხვა გარემო პირობების შესაბამისად.

ცოდნა და განსჯა თამაშობენ გადამწყვეტ როლს მაშინაც, როდესაც აგენტს უხდება მოქმედება ნაწილობრივ დაკვირვებად პრობლემურ

გარემოში. ცოდნაზე დამყარებულ აგენტს უნარი აქვს შეუთავსოს ერთმანეთს ცოდნა და მოცემულ მომენტში მიღებული აღქმა, რათა გამოავლინოს მიმდინარე მდგომარეობის დაფარული ასპექტები მანამაღ, სანამ ის დაიწყებს მოქმედებას. ძალიან მნიშვნელოვანია ისიც, რომ ასეთი აგენტები არიან ძალიან მოქნილები. მათ შეუძლიათ მიიღონ და შეასრულონ ახალი ამოცანები, რომლებიც ჩამოყალიბებულია ცხადად დასმული მიზნის სახით, გარდა ამისა მათ შეუძლიათ შეეგუონ იმ გარემოს ცვლილებას, რომელშიც ისინი მოქმედებენ თავისი ცოდნის განახლების გზით.

ცოდნაზე დამყარებული ნებისმიერი აგენტის ცენტრალურ კომპონენტს ცხადია წარმოადგენს მისი **ცოდნის ბაზა**. სწორედ ასეთი ბაზის ორგანიზება კომპიუტერულ სისტემაში წარმოადგენს ხელოვნური ინტელექტის ერთერთ ძირითად პრობლემას.

ისევე, როგორც პრობლემათა გადაწყვეტის დროს ჩვენ გვაინტერესებდა თუ როგორ წყვეტდა ასეთ პრობლემებს ადამიანი, აქაც რა თქმა უნდა საინტერესოა, თუ როგორაა ორგანიზებული ცოდნა ადამიანში.

## 2.2. სემანტიკური ქსელები.

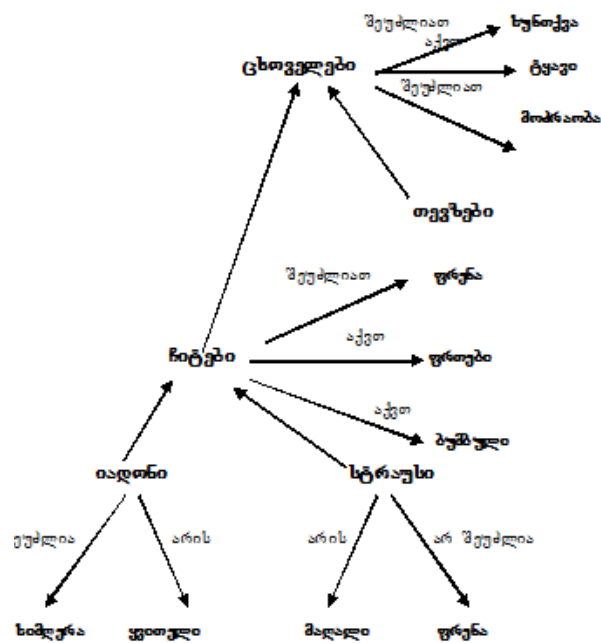
ფსიქოლიგიიდან ცნობილი ასოციონისტური თეორიის თანახმად, ობიექტის მნიშვნელობა განისაზღვრება სხვა ობიექტებთან ასოციაციის ტერმინებში. ამ თეორიის თანახმად, ობიექტების აღქმა ხდება ცნებების საშუალებით. ცნებები წარმოადგენენ გარემოს შესახებ ჩვენი ცოდნის მნიშვნელოვან ნაწილს და დაკავშირებულები არიან სხვა ცნებებთან ასოციაციების საშუალებით. ეს კავშირები წარმოადგენენ ობიექტის თვისებებს და ქცევას. მაგალითად, ცდების საფუძველზე ჩვენ ვახდენთ ცნება “თოვლის” ასოცირებას სხვა ცნებებთან, დავუშოთ ისეთებთან, როგორებიცაა: “სიცივე”, “თეთრი”, “თოვლის ადამიანი”, “ყინული” და ასე შემდეგ. ჩვენი ცოდნა თოვლის შესახებ და გამონათქვამის “თოვლი თეთრია” ჭეშმარიტება იქმნება ამ ასოციაციის ქსელიდან.

ასევე ფსიქოლოგიიდანაა ცნობილი, რომ ეს ასოციაციები ქმნიან იერარქიულ სტრუქტურებს. 1969 წელს კოლინსმა და კვილიანმა ჩაატარეს ცდები, რომლის არსი მდგომარეობდა იმაში, რომ ისინი ცდის

პირებს აძლევდნენ ისეთ შეკითხვებს ფრინველების თვისებების შესახებ, როგორცაა მაგალითად “იადონი ეს ჩიტია?” “შეუძლია იადონს ფრენა?” “სტრაუსი მაღალია?” და სხვა. ზომავენდნენ რა დროს, რომელიც სჭირდებოდათ ცდის პირებს პასუხების მოსაფიქრებლად. მათ აღმოაჩინეს, რომ იმ კითხვებზე პასუხის გასაცემად, რომლებიც ეხება მოცემული ობიექტის კონკრეტულ თვისებებს, ცდის პირებს დასჭირდათ უფრო ცოტა დრო, ვიდრე იმ კითხვებზე, რომლებიც ეხება უფრო ზოგად თვისებებს, დამახასიათებელს არა მარტო ამ კონკრეტული ობიექტისათვის. მაგალითად, კითხვას “იადონი ყვითელია?” სჭირდება გაცილებით ცოტა დრო, ვიდრე კითხვას “იადონს შეუძლია ფრენა?”, მაშინ, როდესაც ანალოგიური შეკითხვა “შეუძლია თუ არა ფრენა სტრაუსს?” მოითხოვს გაცილებით ცოტა დროს. ცდის ავტორები ამას ხსნიან იმით, რომ ადამიანები ინფორმაციას იმახსოვრებენ მაქსიმალურად აბსტრაქტულ დონეზე. იმის მაგივრად, რომ დაიმახსოვრონ იადონს შეუძლია თუ არა ფრენა, ისინი იმახსოვრებენ, რომ იადონი არის ჩიტი, ხოლო ჩიტებს (ძირითადად) შეუძლიათ ფრენა. ამ ცდის საფუძველზე მათ შემოგვთავაზეს სემანტიკური ქსელი, რომელიც ორგანიზებულია იერარქიული პრინციპით, და აღწერს ადამიანის თავის ტვინში ცოდნის შენახვის სქემას.

როგორც ამ ქსელიდან ჩანს, იმისათვის, რომ ცდის პირმა უპასუხოს შეკითხვას, შეუძლია თუ არა იადონს სუნთქვა, ადამიანმა თავის სემანტიკურ ქსელში უნდა გაიროს გზა ”იადონი” → “ჩიტი” → “ცხოველი” ← “სუნთქავს”. სემანტიკური ქსელების ეს პრინციპი ფორმალიზებული იქნა **მემკვიდრეობით სტრუქტურებში**. ეს სტრუქტურები საშუალებას გვაძლევენ დავიმახსოვროთ ინფორმაცია აბსტრაქციის ყველაზე მაღალ დონეზე, რაც ამცირებს ცოდნის ბაზის ზომებს და საშუალებას გვაძლევს თავიდან ავიცილოთ წინააღმდეგობები. მაგალითად, თუ ჩვენ ვაგებთ ცოდნის ბაზას ჩიტების შესახებ, მაშინ შეგვიძლია განვსაზღვროთ ყველა ჩიტისათვის დამახასიათებელი ისეთი ზოგადი თვისებები, როგორცაა “შეუძლიათ ფრენა”, “აქვთ ფრთები” და ა. შ. დავიმახსოვროთ ეს ინფორმაცია ჩიტების აბსტრაქციის დონეზე და მივცეთ შესაძლებლობა ჩიტების კონკრეტულ წარმომადგენელს მემკვიდრეობით მიიღოს ეს თვისებები.

ეს რა თქმა უნდა ამცირებს ცოდნის ბაზის ზომას, იმიტომ რომ ამ თვისებებს განვსაზღვრათ მხოლოდ ერთხელ და არა ყოველი კონკრეტული წარმომადგენლისათვის ცალ - ცალკე. ეს მოსახერხებელია აგრეთვე ცოდნის ბაზაში ცვლილებების შეტანის დროსაც. თუ ჩვენ ბაზას ვუმატებთ ჩიტების რომელიმე კონკრეტულ წარმომადგენელს ან მთლიანად ქვეკლასს, კლასისათვის აღწერილი თვისებები ავტომატურად გაგრძელდება ამ ქვეკლასზეც.



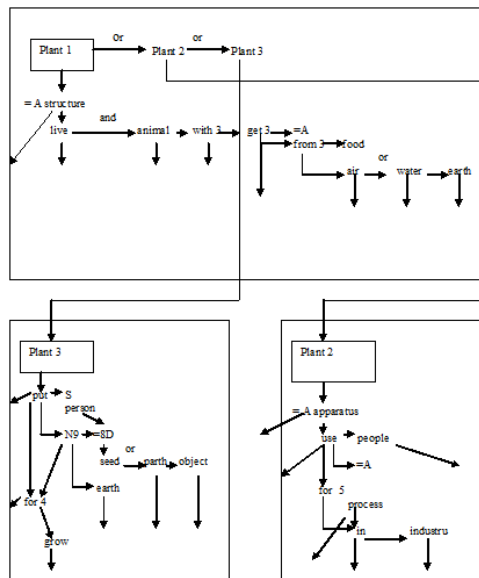
სურ. 10 კოლინის და კვილის მიერ შედგენილი სემანტიკური ქსელი

სემანტიკური ქსელების აღწერის ყველაზე მოხერხებული ფორმაა გრაფები რადგანაც შესაძლებელია ცნებებს შორის დამოკიდებულებების აღწერა კვანძებისა და რკალების საშუალებით. კვილიანის მიერ შედგენილ ცოდნის ბაზაში, რომელიც შეიქმნა სამოციანი წლების ბოლოს და რომელიც ორგანიზებული იყო ლექსიკონის პრინციპის შესაბამისად, გრაფის თითოეული კვანძი შეესაბამებოდა სიტყვებით აღწერილ ცნებას, ხოლო რკალების საშუალებით აღიწერებოდა ასოციაციური კავშირები სხვა ცნებებთან. ცოდნის ბაზა იყო ორგანიზებული სიბრტყეებით, სადაც თითოეული

სიბრტყე წარმოადგენდა გრაფს, რომელიც ახასიათებდა ერთ ცნებას. ამ ქსელიდან ერთერთი ცნების მაგალითი მოყვანილია სურ. 11-ზე.

პროგრამა იყენებდა ამ ბაზას მიმართების მოსაძებნად სიტყვათა წყვილს შორის. ამისათვის ის ახორციელებდა ძებნას სივანეში, სანამ არ იპოვნოდა საერთო ცნებას, ანუ თანაკვეთის კვანძს. გზები საერთო კვანძამდე წარმოადგენდნენ მიმართებებს ცნებებს შორის. მაგალითად, სურ. 12-ზე. ნაჩვენებია თანაკვეთის კვანძი, ორ ცნებას, **cry** (ტირილი) და **comfort** (მოსვენება) შორის. გამოიყენა რა ამ გზების თანაკვეთა, პროგრამამ დაასკვნა, რომ:

"ტირილი სხვა დანარჩენთან ერთად დაკავშირებულია სამწუხარო ხმების წარმოქმნასთან. მოსვენებას შეუძლია შეამციროს მწუხარება."



სურ. 11 სამი სიბრტყე, რომელიც წარმოადგენს სიტყვა "პლანტ"-ის სამ განმარტებას

Plant: 1. ცოცხალი სტრუქტურა, რომელიც არაა ცხოველი, ხშირად აქვს ფოთლები, ღებულობს საკვებს ჰაერიდან,

წყლიდან და მიწიდან.

2. მოწყობილობა, რომელიც გამოიყენება რაიმე საწარმოო პროცესში.

3. მოვათავსოთ (თესლი, მცენარე და ა.შ.) მიწაში გასაზრდელად.

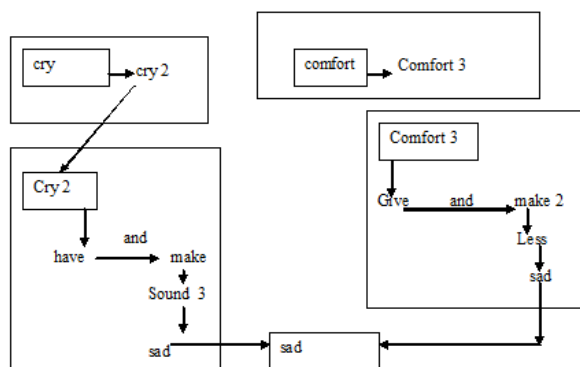
ავტორი თვლიდა, რომ ასეთი მიდგომა სემანტიკისადმი უზრუნველყოფდა სიტემის მიერ ბუნებრივი ენის გაგების შემდეგ შესაძლებლობებს:

1. ინგლისური ტექსტის აზრის გაგება თანაკვეთის კვანძების აგების გზით.

2. სიტყვის მნიშვნელობის ამორჩევა, წინადადების სხვა სიტყვებთან გადაკეთებულ შემოკლესი გზის ამორჩევის საფუძველზე. მაგალითად ინგლისურ წინადადებაში *“Tom went home to water his new plant”* ( “ტომი მოვიდა სახლში თავისი ახალი ნარგავების მოსარწყავად”), სისტემას შეუძლია ამოირჩიოს სიტყვა *“plant”*- ის მნიშვნელობა ცნებების *“water”* და *“plant”* თანაკვეთაზე დაყრდნობით.

3. შეკითხვაში მოცემული ცნებების ასოციაციებზე დაყრდნობით, სისტემა შესძლებს მოქნილად უპასუხოს შეკითხვებს.

მიუხედავად იმისა, რომ ამ და სხვა მისმა მსგავსმა სიტემებმა ნათლად აჩვენეს გრაფების გამოყენების ეფექტურობა ასოციაციური აზრის მოდელირების დროს, ისინი შეზღუდულნი იყვნენ მეტისმეტად ზოგადი ფორმალიზმის გამო. ცოდნის სტრუქტურირება როგორც წესი ხდებოდა სპეციფიკური დამოკიდებულებების ტერმინებში, ისეთი, როგორიცაა კლასი – ქვეკლასი, აგენტი – ზმნა – ობიექტი.



სურ. 12 ცებების cry და comfort თანაკვეთა.



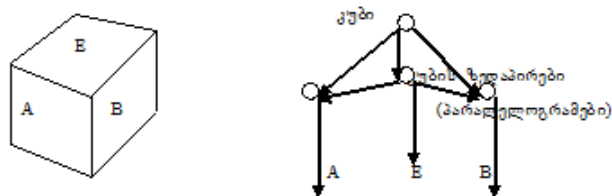
### 2.3. ფრეიმები.

განსხვავებული იდეა ცოდნის კომპიუტერულ სისტემაში წარმოდგენაზე წამოაყენა ცნობილმა ამერიკელმა მეცნიერმა მ. მინსკმა. მან შემოგვთავაზა ფრეიმების თეორია, რომლის ძირითადი მამოძრავებელი იდეა მდგომარეობს იმაში, რომ როდესაც ადამიანი ცდილობს შეიცნოს მისთვის ახალი სიტუაცია, ან ახლებურად შეხედოს მისთვის უკვე ცნობილ საგნებს, ის თავისი მეხსიერებიდან ამოიჩვენებს მონაცემთა რაღაც სტრუქტურას (სახეს, რომელსაც ავტორი უწოდებს ფრეიმს), იმ მიზნით, რომ მასში გარკვეული დეტალების ცვლილების გზით გახადოს ეს სტრუქტურა გამოსადეგი უფრო ფართო კლასის პროცესებისთვის.

ფრეიმი არის სტერეოტიპული სიტუაციის წარმოსადგენი მონაცემთა სტრუქტურა. თითოეულ ფრეიმთან ასოცირებულია სხვადასხვა სახის ინფორმაცია. მისი ერთი ნაწილი მიუთითებს, თუ როგორ უნდა გამოვიყენოთ ფრეიმი, მეორე – თუ რა შეიძლება მოჰყვეს მის შესრულებას და მესამე – რა უნდა მოვიმოქმედოთ, თუ ეს ჩვენი მოლოდინი არ გამართლდა. ფრეიმი შეიძლება წარმოვიდგინოთ როგორც ქსელი, რომელიც შედგება კვანძებისა და კავშირებისაგან ამ კვანძებს შორის. ფრეიმის "ზედა დონეები" მკაცრად განსაზღვრულია, რადგანაც შეესაბამებიან ცნებებს, რომლებიც ყოველთვის სწორია მოცემული შესაძლო სიტუაციებისათვის. შედარებით უფრო დაბალ დონეებზე არის მრავალი განსაკუთრებული წვერო – ტერმინალები ანუ უჯრედები, რომლებიც უნდა შეივსოს დამახასიათებელიმაგალითებით, ან მონაცემებით.

თითოეული ტერმინალის მიერ შეიძლება წამოყენებული იქნას პირობები, რომლებსაც უნდა აკმაყოფილებდნენ მისი დავალებები. მარტივი პირობები, მაგალითად, მოთხოვნა, რომ ტერმინალის დავალება იყოს რომელიმე სუბიექტი, ან გარკვეული ზომის საგანი, განისაზღვრება მარკერების საშუალებით. უფრო რთული პირობებით მოიცემა მიმართებები ცნებებს შორის, რომლებიც მოთავსებულნი არიან სხვადასხვა ტერმინალურ წვეროებში. სემანტიკურად ახლო ფრეიმების ჯგუფი შეიძლება გავაერთიანოთ ფრეიმთა სისტემაში.

განვიხილოთ უმარტივესი მაგალითი. დაუშვათ, ჩვენ გვინდა შევადგინოთ ფრეიმი, რომლის საშუალებითაც შევძლებთ ცოდნის ბაზაში აღვწეროთ კუბი ისე, რომ რა პოზიციიდანაც არ უნდა უყურებდეს ამ კუბს ხელოვნური ინტელექტუალური სისტემა, მან შესძლოს მისი ამოცნობა. სურ. 13-ზე. მოცემულია კუბის გამოსახულება რაიმე კონკრეტული წერტილიდან და შესაბამისი ფრეიმის სტრუქტურა. დაუშვათ, ესლა დამკვირვებელი იწყებს მოძრაობას მარჯვნივ, მაშინ A წახნაგი გაქრება და მის ნაცვლად გამოჩნდება წახნაგი B. თუ ესლა შევეცდებით ხელახლა აღვწეროთ კუბი, უნდა დავივიწყოთ ყველაფერი, რაც ვიცოდით და ხელახლა აღვწეროთ წახნაგი და აღვწეროთ წახნაგი C.



სურ. 13 კუბის გამოსახულება და მისი ფრეიმი

მაგრამ, რადგანაც ჩვენ ვიცით, რომ საგანი არ შეცვლილა, ამიტომ ჩვენ შეგვიძლია შევინახოთ ინფორმაცია B და E წახნაგების შესახებ აღვწეროთ მხოლოდ C წახნაგი. ამასთან, რომ არ დაგვარგოთ ინფორმაცია წახნაგის შესახებ, ჩვენ შეგვიძლია დავუკავშიროთ ის უხილავი წახნაგის ტერმინალს. შესაძლებელია გავაგრძელოთ ასეთი რთული სტრუქტურების აგება, რომელსაც ეწოდება ფრეიმთა სისტემა, როდესაც დაკვირვების წერტილები გვაქვს უფრო მეტი. მნიშვნელოვანია მივაქციოთ ყურადღება სიტუაციის ფრეიმთა სისტემით წარმოდგენის მთავარ თვისებას, რომელიც მდგომარეობს იმაში, რომ სხვადასხვა ფრეიმები, რომლების შედიან ერთსა და იმავე სისტემაში, გამოიყენებენ ერთსა და იმავე ტერმინალებს, რომლებიც შეესაბამებიან ერთსა და იმავე ფიზიკურ მახასიათებელს, რომელიც ჩანს დაკვირვების სხვადასხვა წერტილიდან. ეს გვაძლევს საშუალებას წინასწარ მოვაგროვოთ ინფორმაცია ერთ ადგილას ცნობილი ობიექტების თვისებების შესახებ დაკვირვების წერტილიდან დამოუკიდებლად. ეს განსაკუთრებით მომგებიანია რთული ობიექტების შემთხვევაში,

რადგანაც მათ როგორც წესი გააჩნიათ ბევრი ასეთი თვისება. ამის შედეგად შესაძლებელია მეხსიერების ეკონომია და მცირდება აგრეთვე აღქმის პროცესი, რადგანაც მეხსიერებაში უკვე გვაქვს საჭირო ინფორმაცია და დრო ისარჯება მხოლოდ მის "ამოღებაზე" მეხსიერებიდან.

ავტორის აზრით მეხსიერებაში ფორმირდება ფრეიმთა სისტემა არა ცალკეული ობიექტების გამოსახულებების, არამედ ჩვეულებრივ გამოყენებული "ძირითადი ფორმებისა", რომლებიც სხვადასხვა კომბინაციებში წარმოქმნიან ფრეიმთა სისტემებს ახალი სიტუაციებისათვის.

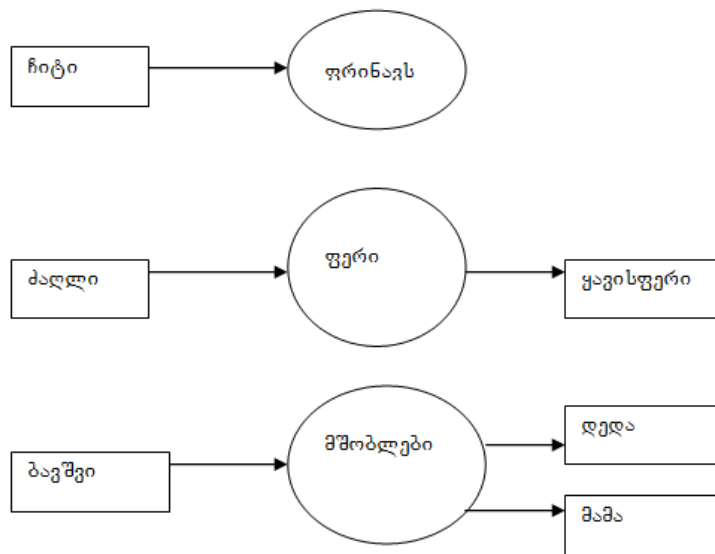
ასეთივე თვისებები ახასიათებს ფრეიმთა სისტემებს, რომლებიც აღწერენ არაეიზუალურ ცოდნას გარე სამყაროს შესახებ. შესაძლებელია ფრეიმთა სისტემების გაერთიანება იერარქიულ სტრუქტურებში, რაც საბოლოოდ მოგვცემს გარე სამყაროს მოდელს.

ფრეიმები აფართოებენ სემანტიკური ქსელების შესაძლებლობებს. ისინი გვაძლევენ საშუალებას წარმოვადგინოთ რთული ობიექტები არა დიდი სემანტიკური ქსელის, არამედ ერთიანი ფრეიმული სტრუქტურის სახით. მიუხედავად ამისა, რთული ცოდნის ბაზების ორგანიზების პრობლემები ასეთი მიდგომის დროს მაინც უნდა გადაიჭრას ძირითადად პროგრამისტის ინტუიციისა და გამოცდილების ხარჯზე.

#### **2.4. კონცეპტუალური გრაფები.**

ცოდნის ბაზების შექმნაში შემდეგი საფეხური იყო ქსელური ენების შექმნა ბუნებრივი ენის სემანტიკის და სხვა გარე სამყაროს მოდელირებისათვის ინტელექტუალურ სისტემებში. ჩვენ განვიხილავთ ქსელური ენების ერთ წარმომადგენელს – კონცეპტუალურ გრაფებს. **კონცეპტუალური გრაფი** წარმოადგენს სასრულ, შეკრულ, ორმხრივ მიმართულ გრაფს. გრაფის კვანძები წარმოადგენენ ცნებებს ან კონცეპტუალურ მიმართებებს. კონცეპტუალურ გრაფებში რკალების ჭდეები არ გამოიყენება. მიმართებები ცნებებს შორის წარმოიდგინება კონცეპტუალური მიმართებების კვანძებით. იმისათვის, რომ განვასხვავოთ ერთმანეთისაგან ცნებები და მიმართებები, სქემაზე პირველს აღნიშნავენ მართკუთხედებით, მეორეს კი – ელიფსებით. (სურ.

14). მოყვანილი კონცეპტუალური მიმართებები წარმოადგენენ სხვადასხვა  $n$ - არობის მიმართებებს. ჩიტი ფრინავს არის უნარული მიმართება, აქ ჩიტი არის ობიექტი, ხოლო ფრინავს – კონცეპტუალური მიმართება. ძაღლის ფერია ყავისფერი არის ბინარული მიმართება, ხოლო ბავშვის მშობლები არიან დედა და მამა კი – ტერანული მიმართება.

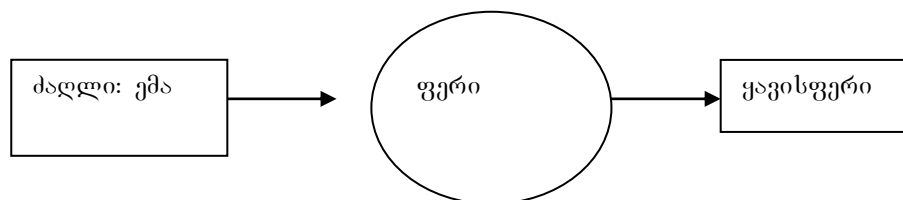


სურ. 14 სხვადასხვა  $n$ -არობის კონცეპტუალური მიმართებები

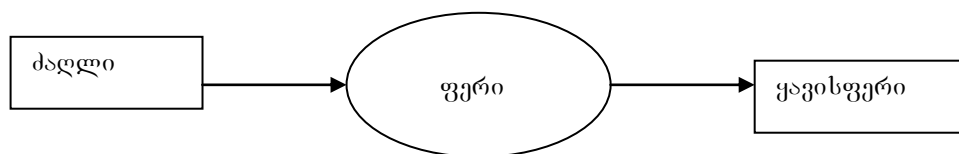
კონცეპტუალურ გრაფებში ცნებების კვანძები წარმოადგენენ ან გარე სამყაროში არსებული კონკრეტული ობიექტების ცნებებს ან მსჯელობის დროს გამოყენებულ ისეთ აბსტრაქტულ ცნებებს, რომლებიც გარე სამყაროში არ გვხვდება (მაგალითად პატრიოტიზმი, დალატი, სიყვარული და ასე შემდეგ).

სემანტიკური ქსელების შექმნისას, როგორც წესი ყურადღებას არ აქცევენ ისეთი მიმართებების განსაზღვრას, როგორებიცაა კლასი – ელემენტი, კლასი – ქვეკლასი, რის გამოც წარმოიშვა სემანტიკური არაერთგვაროვნება. მაგალითად, მიმართება ეგზემპლარსა და კლასს შორის განსხვავდება მიმართებისაგან კლასსა მის სუპერკლასს შორის. აქედან გამომდინარე ზოგიერთი თვისება ეკუთვნის ეგზემპლარს, ხოლო ზოგიერთი კი – კლასს. ცოდნის წარმოდგენას უნდა გააჩნდეს მექანიზმები ამ განსხვავების აღსაწერად. კონცეპტუალურ გრაფებში

თითოეული ცნება წარმოადგენს კონკრეტული ტიპის უნიკალურ ეგზემპლარს. თითოეული ცნების მართკუთხედი აღინიშნება ტიპის ჭდით, რომელიც განსაზღვრავს ამ კვანძით წარმოდგენილ კლასს ან ეგზემპლარის ტიპს. ტიპის ჭდე და ეგზემპლარი გამოყოფილია ორი წერტილით. კონცეპტუალური გრაფები გვაძლევენ საშუალებას აღვწეროთ აგრეთვე კონკრეტული, მაგრამ



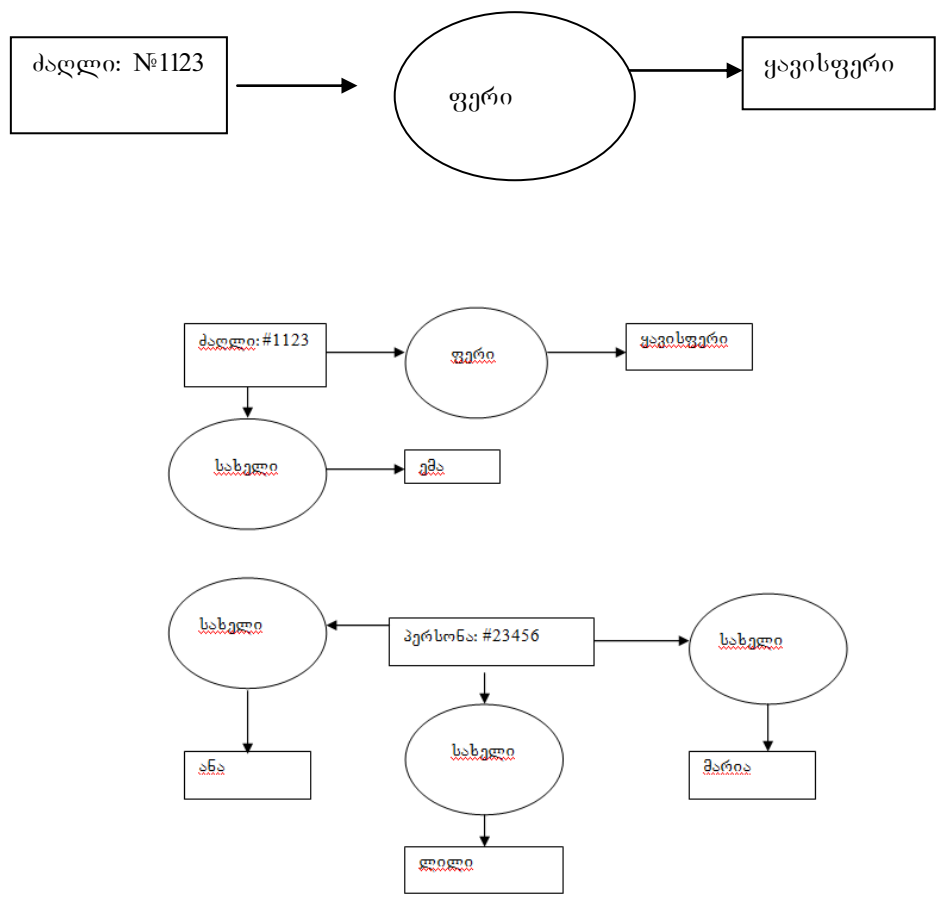
**კონცეპტუალური გრაფი, რომელიც გვიჩვენებს, რომ ძალა ემა ყავისფერია.**



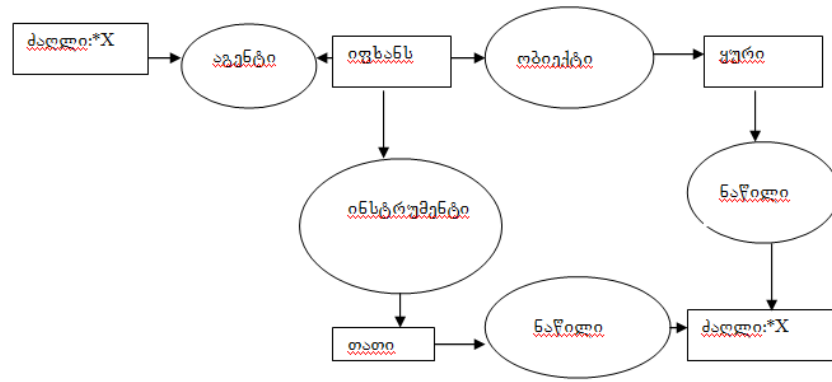
**კონცეპტუალური გრაფი, რომელიც გვიჩვენებს, რომ რომელიღაც არასპეციფირებული, ძალის ტიპის ობიექტი ყავისფერია.**

უსახელო ეგზემპლარები. ამისათვის, თითოეული ეგზემპლარის აღსაწერად მონაცემთა ბაზაში გამოიყენება უნიკალური დესკრიპტორი (აღმწერი), რომელსაც უწოდებენ მარკერს. ის წარმოადგენს რიცხვს, რომლის წინაც განთავსებულია სიმბოლო "#". მარკერი სახელისაგან განსხვავდება იმით, რომ ის არის უნიკალური, ეგზემპლარს შეიძლება არ ჰქონდეს საერთოდ სახელი, შეიძლება ჰქონდეს ერთი, ან რამდენიმე სახელი, მაგრამ მას ყოველთვის აქვს ზუსტად ერთი მარკერი. ეს გვაძლევს საშუალებას თავიდან ავიცილოთ სემანტიკური არაერთმნიშვნელობა, კერძოდ როდესაც გვაქვს ორი ობიექტი ერთი და იგივე სახელით.

სახელისა და მარკერის გამოყენების ნაცვლად შეიძლება გამოვიყენოთ განზოგადოებული მარკერი \* არაცსპეციფიცირებული ეგზემპლარების აღსანიშნავად, თუმცა ეს ინფორმაცია ხშირად გამოტოვებულია ცნებათა ჭდეებთან. კვანძი, რომელიც მოცემულია მხოლოდ ტიპის ჭდით dog, ეკვივალენტურია კვანძის dog:\*. კონცეპტუალური გრაფები საშუალებას გვაძლევენ აგრეთვე გამოვიყენოთ სახელობითი ცვლადები. ისინი აღინიშნება "ვარსკვლავით" და ცვლადის სახელით (მაგალითად \*X, \*foo). მაგალითად, წინადადება **"ძაღლი იფხანს ყურს თათით"** კონცეპტუალური გრაფის საშუალებით აღიწერება ასე



სურ. 15 კონცეპტუალური გრაფები, რომლებიც გვიჩვენებენ მარკერის და სახელის გამოყენების სხვადასხვა ვარიანტებს



სურ. 16 კონცეპტუალური გრაფი წინადადებისათვის "ძალი იფხვანს ყურს თათით"

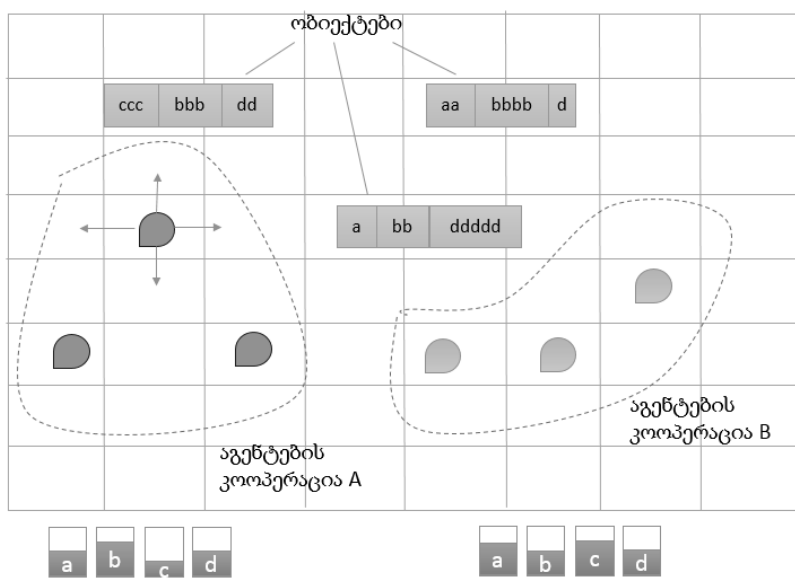
### თავი 3. თამაში "ჭიანჭველების ამოცანა"

#### 3.1 ამოცანის განსაზღვრა

დისერტაციის კვლევის საგანი, მულტიაგენტური გაერთიანებები განვიხილოთ კონკრეტულ მაგალითზე. მაგალითისთვის საჭიროა შევქმნათ ისეთი ვიდეოთამაში, სადაც მონაწილეობას მიიღებენ ინტელექტურად აგენტები. აგენტებს ექნებათ ცოდნის ბაზა, გადაწყვეტილებების გამოტანის მექანიზმი და შესაბამისად იმოქმედებენ.

ნებისმიერ ვიდეოთამაშს აქვს ერთი ძირითადი პრინციპი, იგი არის დელიბრირებული, ანუ გვაქვს ჩაკეტილი სამყარო, სადაც შეგვიძლია მოქმედება. ჩვენი ამოანაც დავაფუძვნოთ ამ პრინციპზე.

მოცემულია ორგანზომილებიანი სიბრტყე, სიბრტყეზე განლაგებულნი არიან ჭიანჭველა აგენტები. მათ შეუძლიათ ამ



სურ. 17 ჭიანჭველების თამაშის გრაფიკული პროტოტიპი

სიბრტყეზე მოძრაობა მხოლოდ ვერტიკალურად, ან ჰორიზონტალურად. სიბრტყეზე შემთხვევითად ჩნდებიან ობიექტები(ფოთლები). ობიექტები წარმოადგენენ შეზღუდული ანბანის შედგენილ მიმდევრობათა მიმდევრობას.(სურ.17)

ჭიანჭველა აგენტებს შეუძლიათ შემდეგი ტიპის მოქმედებები.

- შეზღუდული ზომის ფოთლის გადაადგილება
- ფოთლის დაჭრა იმ სიმბოლოზე რომელსაც იცნობს



- გამოყოფილი სიმბოლოს შენახვა საცავში
- **თამაშის მიზანია** გარკვეულ დროში, კოოპერაციაში მყოფმა ჭიანჭველა აგენტებმა რაც შეიძლება მეტისიმბოლოები დააგროვონ საცავში.

აგენტები, თავდაპირველად, სიბრტყეზე ჩნდებიან შემთხვევით პოზიციებზე. მათ შეუძლიათ როგორც ინდივიდუალური მოქმედება, ასევე კოოპერაციის შექმნა და კოოპერაციული გადაწყვეტილებები. კვლევის მიზანიც სწორედ ამაში მდგომარეობს, შევქმნათ აგენტთა ჯგუფური გადაწყვეტილებების მოდელი.

### **3.2 ერთაგენტიანი და მრავალაგენტიანი სისტემა**

ხელოვნური ინტელექტის ბევრი ამოცანა დამყარებულია ერთ აგენტზე. ამ შემთხვევაში, აგენტი თავად იღებს გადაწყვეტილებებს, მისთვის მეორადია სხვა, მსგავსი, ან არამსგავსი აგენტი, რომელთანაც ერთად შეიძლება რამე გაიზიაროს(ცოდნა, გადაწყვეტილება).

შესაძლებელია ერთაგენტიანი სისტემები შედგებოდეს რამდენიმეობიექტისაგან - სხვადასხვა მოქმედი, ცალკეული კომპონენტებისაგან. ერთაგენტიან სისტემაში თითოეული ობიექტი იღებს დაგალებებს დამოქმედებების ინსტრუქციებს ერთი ცენტრალური პროცესიდან.

ცენტრალიზებული პროცესი ახდენს ყველა ამ ობიექტების მოდელირებას.

ერთაგენტიანი სისტემა ვიდეოთამაშებში ხშირად გამოყენებადია. როგორც აღვნიშნეთ, თამაშის უანრობრივმა მრავალფეროვნებამ განსაზღვრა ხელოვნური ინტელექტის მასშტაბურობა. მაგალითად სურ 18 მოცემულია თამაში Call Of Duty Ghosts 2, სადაც გამოყენებულია ინდივიდუალური აგენტთა ამოცანა - ნებისმიერი სამხედრო, რომელიც მთავარი მოთამაშის მხარესაა, ან არ არის მის მხარეს, მოქმედებს ინდივიდუალურად. მათი ძირითადი მიზანია მოწინააღმდეგის განადგურება ინდივიდუალური ქცევის საფუძველზე.



სურ. 18 ბრძოლის სცენა თამაშიდან Call Of Duty Ghosts 2

მრავალაგენტიანი სისტემაც ერთი, საერთო ცენტრალური პროცესიდან იღებს დავალებებს, თუმცა ერთაგენტიანისგან განსხვავებით, ამ სისტემაში შემავალი ყველა აგენტისთვის ეს დავალება საერთოა. შესაბამისად, მათი მოქმედებაც, ცოდნაც და გადაწყვეტილებების გამომტანიც ერთმანეთთან არის დაკავშირებული. აგენტები კი არ არიან ერთმანეთისთვის მეორადი. ისინი ახდენენ ერთმანეთის მოდელირებას საერთო ცოდნის ქვეშ.

მრავალაგენტიანი სისტემის დიდი უპირატესობა მდგომარეობს იმაში, რომ ისინი მრავალნაკადად შეიძლება მუშაობდნენ, მათი წარმადობა გაცილებით ეფექტურია ინდივიდუალურ აგენტიან სისტემაზე.

ვიდეოთამაშების შემთხვევაში ეს ორი სისტემა ერთმანეთთან მკაცრად განსაზღვრულია - თამაშის ჟანრიდან გამომდინარე იქმნება ხელოვნური ინტელექტი. არსებობს ბევრი თამაში სადაც მრავალაგენტიანი გადაწყვეტილებებია მოცემული. მაგალითად სურ 19-ზე მოცემულია თამაში Stronghold Crusaders 2. ამ თამაშში არსებობს აგენტთა როლები, თვითოეულ როლს აქვს თავისი მიზანი და ქცევის დამახასიათებელი ნიშნები, მაგალითად, მშვილდოსანი ყოველთვის ისვრის მშვილდს, ცხენოსანი არასდროს ისვრის მშვილდს, იგი მხოლოდ ცხენიდან უტევს, უკიდურეს შემთხვევაში ჩამოდის ცხენიდან. შესაბამისი როლის აგენტები გაერთიანებულები არიან საერთო მიზნის ქვეშ და მათ საერთო ცოდნა გააჩნიათ ერთმანეთის ლოკაციაზე, სარდლის ბრძანებებზე და ა.შ.



სურ. 19 ვიდეოთამაში Stronghold Crusaders 2

### 3.3 ჭიანჭველების ამოცანის ერთაგენტიანი და მრავალაგენტიანი მოდელი

ჭიანჭველების ამოცანის იმპლემენტაცია, როგორც თამაში, რამდენიმენაირად შეიძლება. ძირითადი განსხვავება იქნება მოქმედ ინტელექტუალურ აგენტებში - ისინი შეიძლება ამოცანას ხსნიდნენ დამოუკიდებლად, ასევე კოოპერაციულად. განვიხილოთ თვითოეული თამაშის ასეთი განსხვავებული სცენარი.

ინდივიდუალური აგენტების შემთხვევაში, ყოველ ჭიანჭველას გააჩნია რანდენიმე მოქმედების ვარიანტი:

- მოძრაობა, ვერტიკალურად, ან ჰორიზონტალურად.
- გააჩნია რადიუსი, რითაც იგი პოულობს რაიმე ობიექტს.
- ობიექტის გარჩევის უნარი, არის თუ არა ის ფოთოლი.
- ფოთლის ვარგისიანობის დადგენა - ვარგისია თუ არა წასადებად
- თვითოეული ჭიანჭველა სცნობს გარკვეულ სიმბოლოს
- მოაჭრას შესაბამისი სიმბოლო თუ მოიპოვება იგი ფოთოლში
- გადაიტანოს სიმბოლო საცავში და დაბრუნდეს უკან.

ერთი შეხედვით ჩვენ წარმოგვიდგა რთული მოდელი ინდივიდუალური აგენტისათვის. სიტყვაში რთული, იგულისხმება აგენტის მოქმედების მასშტაბურობა, მისი ცოდნის ბაზის მრავალფეროვნება.

თვითოეული ჭიანჭველა თამაშს იწყებს საწყისი ცოდნით, ცოდნის ბაზაში გაწერილია წესები, რითაც იგი არჩევს გარკვეულ ობიექტს ფოთლისგან, ასევე იცის რას ნიშნავს ვარგისიანი ფოთოლი და რაც მთავარია სცნობს თავის სიმბოლოს. განვიხილოთ ერთგვანტვიანი ჭიანჭველათა ამოცანის სიმულაციის ერთი ნაბიჯი.

ჭიანჭველა ჩნდება შემთხვევითად გარკვეულ ადგილას სიბრტყეზე, ის იწყებს მოძრაობას ასევე შემთხვევითად. მას, როგორც ვთქვით გააჩნია, ობიექტის პოვნის გარკვეული რადიუსი. რაიმე ობიექტის პოვნის შემთხვევაში ჭიანჭველა მიმართავს ცოდნის ბაზას, შეადარებს ობიექტს ფოთლის პარამეტრებს და გამოიტანს გადაწყვეტილებას, ეს ობიექტი არის თუ არა ფოთოლი. უარყოფითი პასუხის შემთხვევაში, გააგრძელებს მოძრაობას. ხოლო დადებითი პასუხის შემთხვევაში გადადის შემდეგ ნაბიჯზე. ამ დროს ჭიანჭველამ ამ ოპერაციისთვის დახარჯა  $O(n)$  დრო,  $n$  - ამოცნობისათვის საჭირო ოპერაციების რაოდენობა.

ფოთლის პოვნის შემდეგი ნაბიჯია, მისი ვარგისიანობის დადგენა, შეიცავს თუ არა შესაბამის სიმბოლოს. სიმბოლოს შემოწმება ხდება ფოთლის ყველა შემავალი სიმბოლოს გადამოწმებით - თუ ფოთოლში ვერ მოიძებნა ასეთი ასო, ჭიანჭველა მიატოვებს ფოთოლს და წავა სხვა ფოთლის საძებნელად, პოვნის შემთხვევაში გადადის დაჭრისა და ტრანსპორტირების ეტაპზე.

საბოლოო ეტაპი ფოთლის დამუშავებისა არის სიმბოლოს მოჭრა და გადატანა საცავში. როდესაც ჭიანჭველა მოჭრის ნაპოვნ ასოს, იგი მაშინვე წაიღებს საცავში ამ სიმბოლოს, დაიმახსოვრებს ბოლო კოორდინატს სადაც ვარგისი ფოთოლი იპოვა, რათა ნაჭრის გადატანის შემდეგ დაბრუნდეს და გააგრძელოს ფოთლის დამუშავება.

დამახსოვრებულ კოორდინატზე დაბრუნების შემდეგ ჭიანჭველა თავიდან იმეორებს ფოთლის დამუშავების ბიჯებს, ეძებს სიმბოლოს, პოვნის შემთხვევაში იგი მოაჭრის და გადაიტანს და კვლავ დაბრუნდება. ასე ივლის რეკურსიულად მანა, სანამ ფოთოლში არ ამოიწურება შესაბამისი სიმბოლოები.

განხილული მაგალითის მიხედვით, ინდივიდუალურ აგენტზე დაფუძნებული ქცევებით, ჭიანჭველების ამოცანა რათქმუნდა

ამოიხსნება, თუმცა არ იქნება ეფექტური, რა ოპტიმიზაციაც არ უნდა გაუუკეთოთ ტექნიკურ ალგორითმებს, იდეა მაინც შეანელებს პროცესს. ჭიანჭველას ძალიან ბევრჯერ უხდება ერთი და იგივე ხანგრძლივი ბიჯების გამეორება ბევრჯერ. იგი არც ცოდნას ინახავს, არც გადასცემს არავის და შესაბამისად არც არავისგან იღებს. კარგი იქნებოდა ნებისმიერი ფოთლისთვის მას პირდაპირ მიეღო ინფორმაცია შესაბამისი სიმბოლოს არსებობის ან არ არსებობის შესახებ, რათა მას დაეზოგა სიმბოლოთა შემოწმების ოპერაცია, ასევე კარგი იქნებოდა დაჭრისას არა მხოლოდ ერთი არამედ რამდენიმე სიმბოლო გამოეცალკეებინა, ისევე როგორც გადაეტანა რამდენიმე სიმბოლო. მაგრამ ჭიანჭველა ეულად, მარტო მოქმედებს და ძველი ქართული მულტიფილმის მსგავსად "მარტო მუშაობს და აბა რა იცის როდის იქნება გზა".

ზემოთ ვახსენეთ ცოდნის გადაცემა ინტელექტუალურ აგენტებში, რასაც არ იყენებს ჭიანჭველათა ამოცანის ინდივიდუალური აგენტის მოდელი. იმისათვის რომ ეს ამოცანა უფრო მოქნილი და დროში აჩქარებული გავხადოთ, საჭიროა შემოვიღოთ მულტიაგენტური ქცევები, მულტიაგენტური ცოდნა და ცოდნის გადაცემა. სანამ ჭიანჭველათა მაგალითს განვიხილავთ მულტიაგენტური მხრიდან, განვიხილოთ ცოდნის გადაცემის მექანიზმი ვიდეოთამაშებში.

### **3.4 ცოდნის გადაცემათა ზოგიერთი ტიპი ვიდეოთამაშებში**

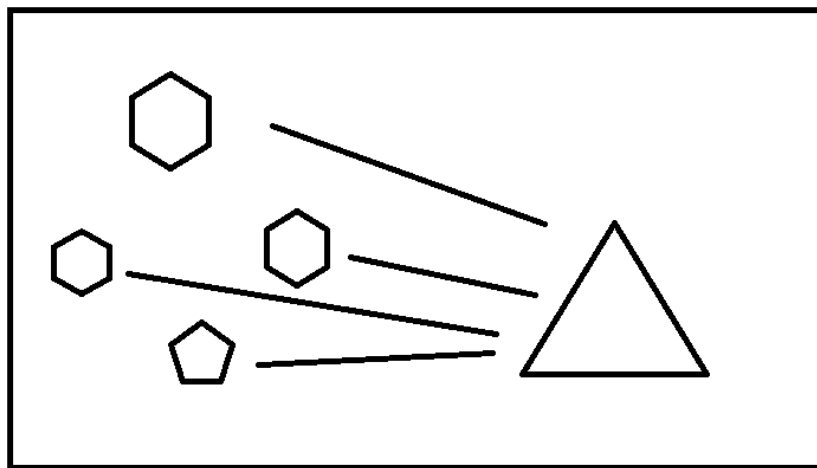
ვიდეოთამაშებში კონკრეტული ცოდნა წარმოდგენილია კონკრეტული საშუალებით - ხშირ შემთხვევაში ეს არის ცოდნის ბაზა. ცოდნის ბაზის განხორციელება მრავალნაირად შეიძლება. ხელოვნური ინტელექტში ცოდნის წარმოდგენის ყველაზე გავრცელებული ვარიანტები ზემოთ განვიხილეთ. ვიდეოთამაშებში ასევე შეიძლება ცოდნა წარმოდგენილი იყოს ხისტად, კოდში გაწერილი. ასევე შეიძლება გაწერილი იყოს გარკვეული წესებით (XML - Extensible Markup Language).

ნებისმიერი ინტელექტუალური აგენტისთვის, რომელიც მონაწილეობს მულტიაგენტურ სისტემაში მნიშვნელოვანია

იერთიერთოს სხვა აგენტებთან - გაუზიაროს ცოდნა და გამოცდილება. ზუსტად ეს განაპირობებს მრავალაგენტიანი სისტემის უპირატესობას ინდივიდუალურთან. რას ნიშნავს ცოდნის გადაცემა? რა გზით შეიძლება იყოს ცოდნა გადაცემული?

მულტიაგენტურ ვიდეოთამაშებში არსებობს რამდენიმე გავრცელებული მეთოდი ცოდნის გადაცემისა.

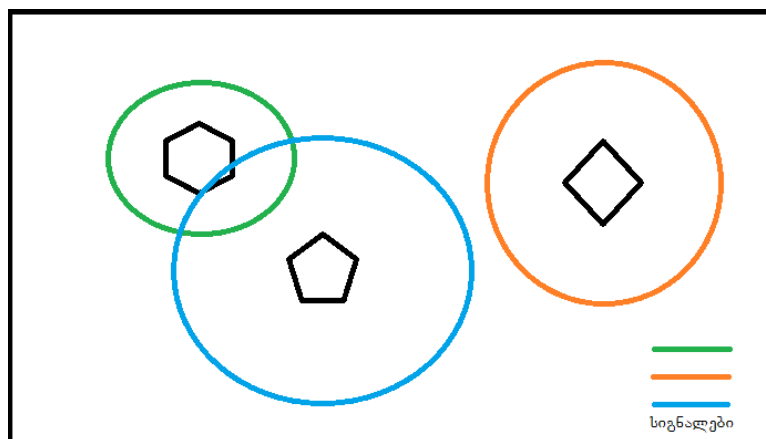
**საერთო გონება - "ყველანი ვფიქრობთ ერთსა და იმავეს"** ეს არის პროგრამული ერთეული, რომელშიც აღწერილია ყველა ცოდნისა და სემანტიკის ერთმანეთთან მიმართება. ნებისმიერი ინტელექტუალური აგენტი იმართება ერთი ცენტრალური გონიდან, მათი კოლექტიური გადაწყვეტილებები გამომდინარეობს ამ სისტემიდან. ამ მოდელის დადებითი მხარე მდგომარეობს მის სიმარტივეში, "საერთო გონების" მოდელი წინასწარ დგება, მან წინასწარ იცის თამაშის ყველა გამოვლენადი ქცევა, რაც შეიძლება აგენტს შეხვდეს და შესაბამისად, აგენტს უწევს მხოლოდ გადაწყვეტილების გამოტანა. ამ სისტემის დინამიურად ფორმირება არ ხდება, გონებას არ ემატება, ან არ აკლდება ცოდნა. რთულია მოქნილი მექანიზმის შექმნა, რამდენი აგენტი შეიძლება ეურთიერთებოდეს საერთო ობიექტს, საჭიროა ყველასთვის დროულად და სინქრონულად მუშაობა. (სურ 20)



სურ. 20 საერთო გონების მქონე აგენტთა მოქმედება

**სიგნალების გადაცემა დანარჩენ აგენტებთან - "დაიჭირე ჩემი ცოდნა"** სიგნალები არის გარკვეული სტრუქტურის პაკეტები. ისინი შეიცავენ ინფორმაციას შესაბამისი სტრუქტურის მიხედვით. ამ ინფორმაციის საფუძველზე აგენტები იღებენ ცოდნას.(სურ 21) პაკეტის

სტრუქტურა წინასწარ არის განსაზღვრული დეველოპერის მიერ თამაშის შესაბამისად. ამ მეთოდის უპირატესობა მდგომარეობს იმაში, რომ აგენტები გარკვეული რადიუსით გავრცელებულ სიგნალს პირდაპირ გადასცემენ სხვა აგენტებს, მიმღები აგენტები ამუშავებენ ამ პაკეტებს და იქცევიან შესაბამისად. სიგნალის გადაცემით ჩქარდება ცოდნის გავრცელება, თუმცა ნელდება პროცესი, ყველა აგენტმა შეიძლება ვერ დაიჭიროს ერთი სიგნალი, ხოლო ვინც დაიჭერს ის გაავრცელებს შესაბამის სიგნალს ხელახლა. ყველაზე უარყოფითი მხარე ამ მეთოდისა არის ის, რომ ცოდნის გადაცემის პროცესი არასრულია. ამ ალგორითმის განხორციელების ძირითადი მეთოდი არის რეკურსია, სიგნალი რეკურსიულად ვრცელდება, სანამ ყველა საჭირო აგენტი არ მიიღებს და დაამუშავებს, რეკურსია და დამუშავება ერთმანეთზეა დამოკიდებული. ამავე დროს, სიგნალების რადიუსი შეიძლება შეზღუდული იყოს - როდესაც გარკვეული სიგნალი ვერ მიაღწევს აგენტამდე, საჭიროა მისი თავიდან გაგზავნა, ან ისეთი აგენტის მიერ გაგზავნა რომელიც მიიღებს. რადგან წინასწარ არ არის ცნობილი რომელი აგენტები ინფორმირდებიან, შესაბამისად ყველა აგენტს უწევს სიგნალის გადაცემა. და ეს პროცესი არის რთული როგორც მესხიერების ასევე წარმადობის მხრივ.



სურ. 21 აგენტების მიერ სიგნალების მეშვეობით გაზიარებული ცოდნის გრაფიკული მოდელი

### 3.5 ჭიანჭველების ამოცანის მრავალაგენტიანი მოდელი

ჭიანჭველების ამოცანის ზემოთგანხილული ერთაგენტიანი სისტემა ვნახეთ, რომ ბევრი ეტაპებისგან შედგება. ეს ეტაპები ყოველთვის მხოლოდ ერთი აგენტის მიერ არის გასაკეთებელი. ჩვენ შეგვიძლია ეს

ამოცანა განვიხილოთ როგორც მრავალაგენტიანი თამაში, სადაც აგენტებს აქვთ ტავიანთი როლები, შეუძლიათ კოოპერაცია. ისინი მოქმედებენ როგორც ინდივიდუალურად, ასევე კოლექტიურად. კოლექტიური მოქმედების შემთხვევაში მათი გადაწყვეტილებები იქნება შესაბამისად, კოლექტიური, თუმცა ინდივიდუალურზე დაფუძნებით.

რადგან თამაშია ჩვენი ამოცანა, მივამსგავსოთ ბოლომდე თამაშს, შემოვიღოთ გარკვეულ აგენტთა როლები. ბევრი თამაშის ძირითადი ფუნქციური მექანიკაა გარკვეულ შინაარსობრივ როლზეა მორგებული. მოთამაშე ყოველთვის მოქმედებს რომელიმე დამახასიათებელი თვისებების მქონე მოთამაშით. როლები წინასწარ არჩევადია, შესაბამისი როლი ითვალისწინებს შესაბამისი თამაშის სტრატეგიას.

თამაშების ინდუსტრიაში ბევრი სათამაშო როლი ერთმანეთს ჰგავს, თუმცა შესაბამისი თამაშის ტიპიდან გამომდინარე მაინც აქვთ გარკვეული ინდივიდუალური თვისებები. ინდივიდუალური თვისებები ერთიანდება სიტყვაში "უნარჩვევა" (Skill). (სურ.22, სურ 23, სურ 24) მოთამაშეს თუ საშუალება აქვს უნარჩვევის წინასწარი არჩევა, შესაბამისად, ის ირჩევს თავისი სათამაშო სტრატეგიიდან გამომდინარე.

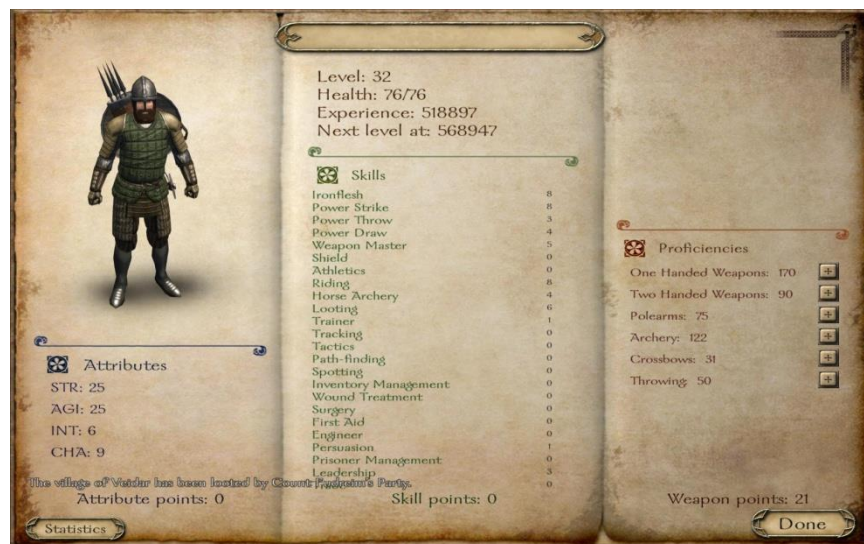


სურ. 22 თამაში Dark Souls პერსონაჟის უნარჩვევების განსაზღვრა თამაშის დაწყების წინ





სურ. 23 თამაში World Of Warcraft პერსონაჟის უნარჩვევების განსაზღვრა თამაშის აწყობის წინ



სურ. 24 თამაში Mount And Blade პერსონაჟის

### 3.6 ჭიანჭველების ამოცანის უნარჩვევებზე დაფუძნებული მრავალაგენტური მოდელი Skill Based Model

როგორც აღვნიშნეთ, თამაშებში მოთამაშისა და აგენტის სათამაშო როლი მის სტრატეგიას განსაზღვრავს. განვიხილოთ ჭიანჭველების ამოცანის მრავალაგენტური მოდელი. ერთაგენტისგან განსხვავებით, აქ ფუნქციურობა გადანაწილებული იქნება, შესაბამისად ასეთი ტიპის

მოდელი განსაზღვრავს ამოცანის სწრაფ წარმადობასა და მულტიინაკადიანობას.

სანამ ჭიანჭველათა მოაცანის მულტიაგენტურ ფუნქციურობას განვიხილავთ, ამოცანას შევხედოთ როგორც კლასიკური თამაში, სადაც როლები მნიშვნელოვანია. ჩვენი ამოცანისთვის განსაზღვროთ აგენტ-ჭიანჭველათა როლები დამათი დამახასიათებელი უნარჩვევები.

- როლი - **ჭიანჭველა მზვერავი (Walker)** - მისი მთავარი ფუნქციაა იაროს მდებლობა და ეძებოს ფოთლები. ჭიანჭველის ცოდნის ბაზა შედგება სასტარტო მონაცემებისაგან. შეუძლია ობიექტის გარჩევა ფოთლისგან და ფოთლის ვარგისიანობის დადგენა. მას აქვს ფოთლის აღმოჩენის გარკვეული რადიუსი, ეს რადიუსი არის დინამიური და დამოკიდებული შეიძლება იყოს მოძრაობის სიჩქარეზე. თუ ჭიანჭველა სწრაფია, რადიუსი შეიძლება იყოს პატარა, ხოლო თუ ჭიანჭველა ნელა დადის რადიუსს დიდი ექნება.

- როლი - **ჭიანჭველა ფოთლის მჭრელი (Rogue)** - ამ ჭიანჭველა აგენტის მთავარი ფუნქციაა ფოთლის დაჭრა შესაბამის ამოცნობად სიმბოლოზე, მათი ცოდნის ბაზა შეესაბამება მონაცემით, რომელიც ზუსტად ამ შესაბამის სიმბოლოს მიუთითებს. ჭიანჭველა მოძრაობს გარკვეული სიჩქარით. სცნობს მხოლოდ ერთ სიმბოლოს.

- როლი - **ჭიანჭველა გადამზიდი (Support)** - ამ როლის ჭიანჭველის ფუნქციაა დაჭრილი ფოთლის ნაჭრების გადატანა საცავში. მათი უნარებია - სიარულის სიჩქარე და ნაჭრების აწევის ძალა. ძალა, რომელიც განსაზღვრავს რამდენი ფოთლის ნაჭერი შეუძლია გადაიტანოს საცავში ერთდროულად.

როგორც ვხედავთ ნებისმიერი როლისათვის ზოგი უნარჩვევა საერთოა, თუმცა ეს საერთო უნარჩვევა რამდენად მნიშვნელოვანია თვითოეულისთვის ეს ცალკე საკითხია. ასევე არსებობს კონკრეტული, როლისათვის დამახასიათებელი უნარები. მოთამაშის , ან აგენტის თამაშის სტილის შესაბამისად შეიძლება მოვახდინოთ უნარჩვევების კლასიფიკაცია.

- **მოძრაობის სიჩქარე** - ნებისმიერი ჭიანჭველა მოძრაობს თამაშში, მათი გადაადგილების სიჩქარე გარკვეულ სისწრაფესთან არის

დამოკიდებული, ეს სისწრაფე შეიძლება განისაზღვროს სიდიდით, შესაბამისად ეს სიდიდე გარკვეული როლებისათვის მნიშვნელოვანია, მაგალითად მზვერავისათვის და გადამზიდისათვის, მათი მოძრაობა პირდაპირპროპორციულია მათი მუშაობის წარმადობასთან და შესაბამისად ამოცანის მოხსნის სისწრაფესთანაც.

- **მდებლობა ობიექტის აღმოჩენა** - ეს არის უნარი მხოლოდ მზვერავისათვის, შესაბამისად დანარჩენი როლის ჭიანჭველებს არ სჭირდებათ ეს ქცევა, რადგან მათი მოქმედების სტრატეგიაში არ შედის.

- **ფოთლის ვარგისიანად ცნობა** - ეს უნარიც მხოლოდ მზვერავისთვის იქნება განკუთვნილი, მხოლოდ მზვერავს შეუძლია ფოთლის შეფასება.

- **სიმბოლოს ამოცნობა და დაჭრა** - ეს უნარი განკუთვნილია მხოლოდ ფოთლის მჭრელებისათვის, მათ აქვთ უნარი უკვე ნაცნობ ფოთლს გაუკეთონ ანალიზი და და მოაჭრან შესაბამისი ამოცნობადი სიმბოლო.

- **მოჭრილი სიმბოლოს ტრანსპორტირება** - მოჭრილი სიმბოლოს ტრანსპორტირება შეუძლიათ მხოლოდ გადამზიდ ჭიანჭველებს, შესაბამისად მათი ეს უნარი ინდივიდუალურია და საჭიროა მხოლოდ ამ როლის ჭიანჭველებისათვის.

კონკრეტულ ამოცანაში, ჭიანჭველათა როლები და უნარჩვევები ერთმანეთთან მჭიდროდ არის დაკავშირებული, შეიძლება ითქვას, ამ პრინციპით არის გადანაწილებული მთლიანობაში, ამოცანის ამოხსნა. მზვერავები დადიან ველზე და ეძებენ ობიექტებს თავიანთი ძებნის რადიუსის მიხედვით, რაც მთავარია ისინი მხოლოდ ეძებენ, ანუ მათი ფუნქცია მხოლოდ მისი შესაბამისი კლასიდან გამომდინარეობს. ისევე როგორც ფოთლის მჭრელი მხოლოდ ფოთლს იცნობს და ჭრის შესაბამის სიმბოლოს. ხოლო გადამზიდი ამარაგებს საცავს დაჭრილი სიმბოლოებით.

თუ ჩვენ ჭიანჭველების ამოცანას განვიხილავთ როგორც მრავალაგენტიან მოდელს, ბუნებრივია, ჩნდება აუცილებლობა ამ აგენტთა ერთმანეთთან ურთიერთქმედებაზე, საჭიროა აგენტები

კლასიკური მრავალაგენტიანი სისტემის გაგებიდან ერთმანეთს ცოდნას უზიარებდნენ - უნდა გამოვიყენოთ რომელიმე მეთოდი ცოდნის გადაცემისა. ზემოთ განვიხილეთ შესაბამისი მეთოდები ვიდეოთამაშებში ცოდნის გადაცემისა. აღვწეროთ შედარებით ოპტიმალური და ჩვენს ამოცანაზე მორგებული მეთოდი - **ცოდნის მარკირების ამოცანა**,

### **3.7 ცოდნის მარკირების ამოცანა, როგორც ცოდნის გადაცემის**

#### **საშუალება ჭიანჭველების ამოცანაში**

ჭიანჭველების ამოცანაში, როგორც მრავალაგენტიან თამაშში საჭიროა ცოდნის გადაცემა. ამ ამოცანაში მონაწილე აგენტები დაყვავით როლებად. შესაბამისად, თვითოეული როლის მქონე აგენტმა მისი მოქმედების შესახებ სხვა აგენტს უნდა აცნობოს. კონკრეტული ამოცანიდან გამომდინარე ყველაზე კარგი ვარიანტი იქნებოდა ცოდნის მარკირების ამოცანა.

ცოდნის მარკირების ამოცანის პრინციპი ასეთია, სიბრტყის(ან სივრცის) გარკვეულ პოზიციაზე  $(X,Y)$  შეინახოს გარკვეული ინფორმაცია, ცოდნა  $K$ . ამ მეთოდის უპირატესობა იმაში მდგომარეობს, რომ ინტელექტუალურ აგენტს პირდაპირი გზით მიეწოდება ცოდნა ბაზაში. უფრო კონკრეტულად კი, თუ აგენტს უწევს რაიმე სიგნალის ან ინფორმაციის გაანალიზება და დასწავლა, მას ეს ინფორმაცია პირდაპირი გზით მიეწოდება ცოდნის ბაზაში. ცოდნაში პირდაპირი გზით მიწოდებული ინფორმაცია, რომელიც გამოტოვებს ანალიზის ეტაპს, იმდენჯერ არის სწრაფი რა დროც გაანალიზებას სჭირდება.

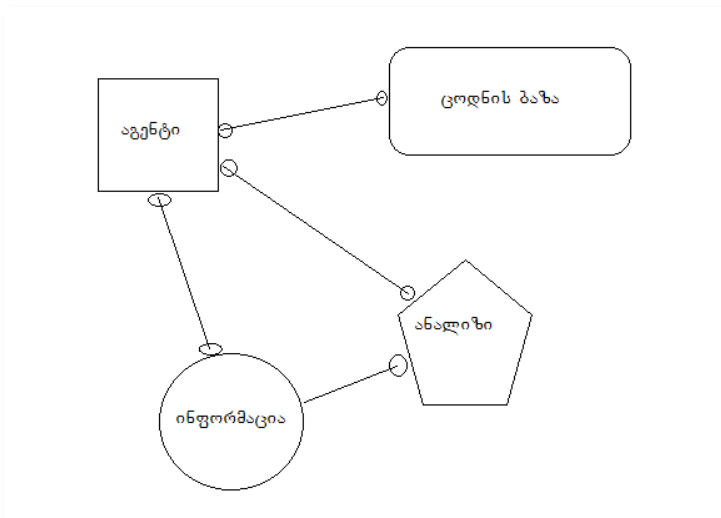
ცოდნის პირდაპირი გზით გადაცემის უკეთ გასაგებათ განვიხილოთ მაგალითი, ელენემ ჩაიფიქრა რიცხვი  $N$ . სალომე მას ეკითხება თუ რა რიცხვი ჩაიფიქრა, ელენე ეტყვის რამდენიმე მინიშნებას ამ ამოცანაზე:

- რიცხვი ლუწია.
- რიცხვი ორნიშნაა.
- რიცხვი ათზე იყოფა უნაშთოდ.
- რიცხვი თხუთმეტზეც იყოფა უნაშთოდ.
- რიცხვი არ არის მეტი ორმოცზე.

სალომე ამ რამდენიმე მინიშნების მოსმენის შემდეგ იფიქრებს,

- რიცხვი ლუწია? ესე იგი იყოფა ორზე.
- რიცხვი ორნიშნაა? მეტია, ან ტოლი 10-ზე და ნაკლებია ან ტოლი 98-ზე.
- თუ რიცხვი ათზე უნაშთოდ იყოფა, ესე იგი მეტია, ან ტოლი 10-ზე და ნაკლებია, ან ტოლი 90-ზე.
- ხოლო თუ თხუთმეტზეც იყოფა უნაშთოდ, ესე იგი გამოცნობის ქვედა ზღვარი იზრდება და პირობაც იცვლება - რიცხვი მეტია, ან ტოლი 30-ზე და ნაკლებია ან ტოლი 90-ზე.
- თუ რიცხვი არ არის 40-ზე მეტი მაშინ ეს რიცხვი ყოფილა 30.

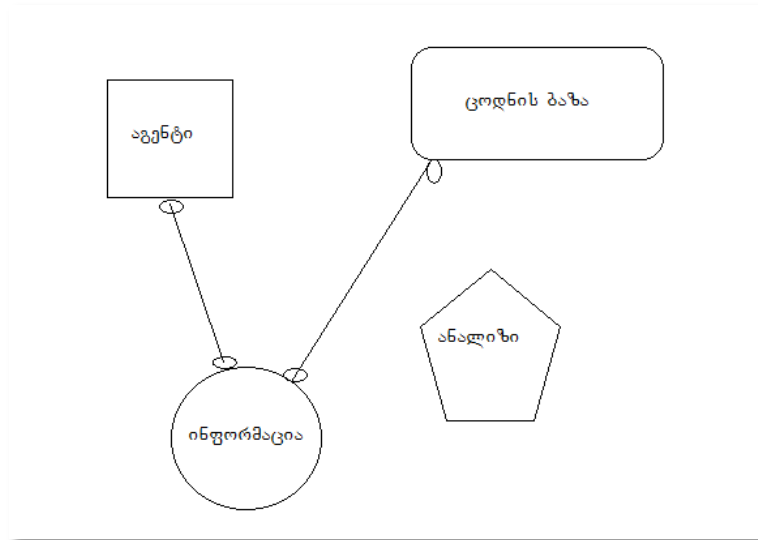
ეს პროცესი რომ სიტყვიერად აღვწეროთ, გამოდის, რომ ელენეს მიერ ჩაფიქრებული ინფორმაცია, კონკრეტულად კი რიცხვი 30, არაპირდაპირი გზით მიეწოდა სალომეს. სალომემ მხოლოდ ელენეს გამოცანებზე დაყრდნობით გააკეთა ანალიზი და დაასკვნა რა რიცხვიც ჩაიფიქრა. (სურ 25) მოცემულია აგენტის მიერ არაპირდაპირი გზით მიღებული ცოდნა, როდესაც მას იწვევს გაანალიზოს ინფორმაცია.



სურ. 25 ცოდნის მიწოდება არაპირდაპირი გზით

სულ სხვა, გაცილებით მარტივი სურათი გვექნებოდა რომ მომხდარიყო ცოდნის პირდაპირი გზით გადაცემა, ჩვენი კონკრეტული მაგალითი ასეთი სახის იქნებოდა - ელენე ჩაიფიქრებდა რიცხვს K, ხოლო სალომეს პირდაპირ ეტყოდა პასუხს და მას აღარ დასჭირდებოდა დროის დახარჯვა მინიშნებების გამოცნობაზე, მათ

გაანალიზებაზე. (სურ 26) ანალოგურად ხდება ხელოვნურ ინტელექტშიც როცა გვაქვს მსგავსი შემთხვევა, მექანიზმი ზოგავს ბევრ დროს ცოდნის გადაცემის პირდაპირ გადაცემაზე.



სურ. 26 ცოდნის მიწოდება პირდაპირი გზით

### 3.8 ცოდნის მარკირება ჭიანჭველების ამოცანაში მზვერავი ჭიანჭველის შემთხვევაში

ზემოთგანხილული საკითხებიდან დავასკვნით, რომ ზოგიერთ ამოცანაში საჭიროა გვექონდეს მულტიაგენტური მექანიკა. ჩვენ უკვე ვიცით რას ნიშნავს მულტიაგენტურ ვიდეოთამაშში ცოდნის გადაცემა, ვიცით რა უპირატესობა აქვს ინტელექტუალურ აგენტთა როლებად და კლასებად დაყოფას. ჭიანჭველების ამოცანა განვიხილოთ როგორც მულტიაგენტური ვიდეოთამაში, სადაც ძირითადი წარმადობის მექანიკა მდგომარეობს ცოდნის მარკირებით გადაცემაში.

ჭიანჭველა მზვერავი მოძრაობს ველზე ნებისმიერი მიმართულებით. მისი რადიუსისა და შესაბამისად პოულობს ობიექტებს და იწყებს მათ გარჩევას ფოთლისგან. ხდება ცოდნის ბაზაში ფაქტებისა და პირობების გადამოწმება. განვიხილოთ ერთი სიმულაციური ბიჯი მზვერავი ჭიანჭველის მოქმედებისა, როდესაც ის პოულობს ფოთოლს. მზვერავის ცოდნის ბაზა დაახლოებით ასე გამოიყურება:

**ფაქტები:**

**ფაქტი1:** ფოთოლი მწვანეა

**ფაქტი2:** ფოთოლი მსუბუქია 0.01 გრამზე

**პირობები:**

**პირობა1:** თუ ნაპოვნი ობიექტი მწვანეა, შეამოწმე ახალი პირობა მეორე ფაქტზე

თუ არა, დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე იმის შესახებ, რომ ეს ობიექტი არ არის ფოთოლი. გააგრძელე ძებნა.

**პირობა2:** თუ ნაპოვნი ობიექტი 0.01 გრამზე მსუბუქია, ის ფოთოლია, დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე რომ ის ფოთოლია. თუ არა - დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე იმის შესახებ, რომ ეს ობიექტი არ არის ფოთოლი და გააგრძელე ძებნა.

მზვერავის მოქმედების თანმიმდევრობა მკაცრადაა განსაზღვრული, მან თუ იპოვა ნებისმიერი ობიექტი იგი თავის ცოდნის ბაზაზე დაფუძნებით იწყებს ანალიზს, არის თუ არა იგი ფოთოლი. გარკვევის შემთხვევაში იგი ტოვებს ცოდნას კონკრეტულ ადგილას, რომ ფოთოლი უპოვია, ხოლო არ პოვნის შემთხვევაში შესაბამისად იქცევა, მხოლოდ არგუმენტი არის უცნობი ობიექტი, ანუ არ უპოვია ფოთოლი.

ცოდნის დატოვებისას შესაბამისი მზვერავი უმარტივეს საქმეს უკვე შემდეგ მომავალ მზვერავს, რომელიც კონკრეტული მიზნით მოადგება ამავე ობიექტს, მას შეუძლია მხოლოდ შეამოწმოს მისი წინამორბედის დატოვებული ნიშანი, არის თუ არა ფოთოლი ეს ობიექტი. დადებითი და უარყოფითი პასუხის შემთხვევაში აღარაფერს შეამოწმებს, რადგან მან უკვე იცის ინფორმაცია ამ ობიექტზე, გააგრძელებს გზას. მზვერავის მოქმედების ფსევდო აღწერა შემდეგნაირად გადაკეთდება:

**პირობა 1ა. თუ ნაპოვნი ობიექტთან უკვე არის მარკირებული ცოდნა მის რაობაზე, გააგრძელე გზა, თუ არა და გადადი პირობა 1-ზე.**

**პირობა1:** თუ ნაპოვნი ობიექტი მწვანეა, შეამოწმე ახალი პირობა მეორე ფაქტზე

თუ არა, დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე იმის შესახებ, რომ ეს ობიექტი არ არის ფოთოლი. გააგრძელე ძებნა.

**პირობა2:** თუ ნაპოვნი ობიექტი 0.01 გრამზე მსუბუქია, ის ფოთოლია, დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე რომ ის ფოთოლია. თუ არა - დატოვე ცოდნა კონკრეტულ ადგილას კონკრეტულ ობიექტზე იმის შესახებ, რომ ეს ობიექტი არ არის ფოთოლი და გააგრძელე ძებნა.

როგორც ვხედავთ, ცოდნის მარკირების გამოყენებით ჭიანჭველა მზვერავს ძალიან შეიძლება გაუმარტივდეს საქმე. იგი ამ მექანიზმის გამოყენებით დაზოგავს მთელ რიგ ოპერაციებს, რომელიც კავშირშია ობიექტის ანალიზთან ცოდნის ბაზაზე დაყრდნობით, ხოლო ცოდნის მარკირებით, მას პირდაპირ მიეწოდა დამუშავებული ინფორმაცია.

### **3.9 ცოდნის მარკირება ჭიანჭველების ამოცანაში ფოთლის მჭრელი ჭიანჭველის შემთხვევაში**

რაც შეეხება ჭიანჭველა ფოთლის მჭრელებს, მზვერავების მსგავსადაც ორგანიზმილებაში მოქმედებენ. მათი ფუნქციაა ფოთლის დაჭრა, თუმცა რა იციან, რომ ობიექტი ფოთოლია. მათი ამ ცოდნის შესავსებათ გვაქვს ორი ვარიანტი:

- 1) ობიექტის პოვნის შემთხვევაში გააკეთონ მისი ანალიზი მზვერავის შესაბამისად
- 2) კონკრეტულ ობიექტთან მიმართებაში იხელმძღვანელონ მზვერავის მიერ მარკირებული ინფორმაციით.

ცხადია, პირველი ვარიანტი არ გამოდგება. ჩვენ ვთქვით, რომ განვიხილავთ ჭიანჭველების ამოცანის მულტიაგენტურ ვარიანტს. ჭიანჭველები დაყვავით კლასებად, თვითოეულს გააჩნია თავისი როლი და მთავარი ფუნქცია. ფოთლის მჭრელს უბრალოდ, არ შეუძლია ფოთლის ამოცნობა, იგი მხოლოდ სიმბოლოებს სცნობს. გვრჩება მეორე ვარიანტი, რომელიც საუკეთესო გამოსავალია - ფოთლის მჭრელი მოქმედებს მზვერავის ინფორმაციაზე დაყრდნობით. განვიხილოთ მზვერავის ცოდნის ბაზა, ფაქტები და პირობები:

#### **ფაქტები:**

**ფაქტი 1 :** ვცნობ მზვერავის დატოვებულ ინფორმაციას(ობიექტი ფოთოლია თუ არ არის ფოთოლი)



**ფაქტი 2 :** სიმბოლო, რომლის ამოცნობაც შემიძლია გამოიყურება როგორც T

**პირობები:**

**პირობა 1:** თუ ობიექტი მარკირებულია ცოდნით გადადი პირობა 2-ზე, თუ არა, გააგრძელე გზა სხვა ფოთლის ძიებაში.

**პირობა 2:** თუ მარკირებული ობიექტი ფოთოლია გადადი პირობა 3-ზე, თუ არა გააგრძელე გზა სხვა ფოთლის ძიებაში.

**პირობა 3:** შეამოწმე, თუ ფოთოლი შეიცავს სიმბოლო T, მთავარი სიმბოლო და დატოვე ინფორმაცია ამის შესახებ, კონკრეტულად, დარჩა თუ არა შესაბამისი სიმბოლო ისევ ფოთოლში, სულ რამდენი სიმბოლო არის მოჭრილი. თუ არ შეიცავს ფოთოლი სიმბოლოს, დატოვე ინფორმაცია ამის შესახებ, რომ არ შეიცავს. გააგრძელე გზა.

ფოთლის მჭრელიც, შეიძლება ითქვას, დამოკიდებულია მზვერავის შენახულ ინფორმაციაზე. ობიექტის პოვნისას, იგი იმ შემთხვევაში გადადის მოქმედებაზე, თუ მან მზვერავის დატოვებული ცოდნა აღმოაჩინა კონკრეტულ ობიექტზე, თუ ვერ აღმოაჩინა აგრძელებს გზას. საინტერესოა ისიც, რომ სიმბოლოს მოჭრის თუ არ მოჭრის შემთხვევაში, მჭრელი ტოვებს ცოდნას მის კონკრეტულ მოქმედებაზე. ეს ცოდნა სხვა ფოთლის მჭრელს გამოადგება, მას შეუძლია პირდაპირ გაიგოს არის თუ არა მისი შესაბამისი სიმბოლო ამ ფოთოლში. ამ ფაქტზე დაფუძნებით, ფოთლის მჭრელის ქცევა იცვლება შემდეგნაირად:

**პირობა 1:** თუ ობიექტი მარკირებულია ცოდნით გადადი პირობა 2-ზე, თუ არა, გააგრძელე გზა სხვა ფოთლის ძიებაში.

**პირობა 2:** თუ მარკირებული ობიექტი ფოთოლია, გადადი **პირობა 2.1-ზე**, თუ არა გააგრძელე გზა სხვა ფოთლის ძიებაში.

**პირობა 2.1 :** შეამოწმე, თუ არსებობს წინა მჭრელების ინფორმაცია საკუთარ სიმბოლოზე, ფოთოლში არსებობს სიმბოლო? გადადი პირობა 3-ზე, არ არსებობს? გააგრძელე გზა.

**პირობა 3:** შეამოწმე, თუ ფოთოლი შეიცავს სიმბოლო T, მთავარი სიმბოლო და დატოვე ინფორმაცია ამის შესახებ, კონკრეტულად, დარჩა

თუ არა შესაბამისი სიმბოლო ისევ ფოთოლში. თუ არ შეიცავს, დატოვე ინფორმაცია ამის შესახებ, რომ არ შეიცავს. გააგრძელე გზა.

წინა კოლების ცოდნაზე დაფუძნებით, მას შეუძლია დაზოგოს სიმბოლოს შემოწმების ოპერაციები. იგი პირდაპირი გზით მიიღებს შეტყობინებას თავისი სიმბოლოს ფოთოლში არსებობის შესახებ.

### **3.10 ცოდნის მარკირება ჭიანჭველების ამოცანაში ფოთლის გადამზიდი ჭიანჭველის შემთხვევაში**

ჭიანჭველა-გადამზიდი აგენტის ფუნქცია მარტივია - იპოვოს მოჭრილი ფოთლის ნაჭერი და წაიღოს საცავში, თუმცა მის ქცევაში საჭიროა გარკვეული ოპტიმიზაცია. განვიხილოთ ამ როლის აგენტის ცოდნა:

#### **ფაქტები:**

**ფაქტი 1:** ნებისმიერი მოჭრილი ფოთლის ზომა არ აღემატებაD-ს.

**ფაქტი 1.1:** მოჭრილი ფოთლის ნაჭერი შედგება ერთი სიმბოლოსგან.

**ფაქტი 2:** ყველა ფოთლის ნაჭერს, რომელსაც აქვს ერთი სიმბოლო, შეიძლება გადაიტანოს საცავში.

**ფაქტი 3:** შეუძლია გადაიტანოს ფოთლის ნაჭერი(ნაჭრები) თუ ჯამში იწონის(იწონიან)m-მასას.

#### **პირობები:**

**პირობა 1:**თუ ნაპოვნი ობიექტის ზომა ნაკლებიაD-ზე. ის ფოთლის ნაჭერია, გადადი პირობა 2-ზე, თუ არა, გააგრძელე გზა სხვა ობიექტის ძიებაში.

**პირობა 2:** თუ ფოთლის ნაჭერი შეესაბამება წადების პირობას, ნუ ის მსუბუქია, გადავიდეს პირობა 3-ზე, თუ არა და წაიღოს ერთი ნაჭერი საცავში და ნაჭრის დაბინავების შემდეგ დაბრუნდეს, ან არ დაბრუნდეს ამ ადგილას.

**პირობა 3:** მოეცმულ ფოთლის მიდამოში შეამოწმოს ყველა არსებული ფოთლის ნაჭრებიდან რამდენის წადება შეუძლია.

მე-2 და მე-3 პირობები კითხვის ნიშანს ჰბადებს. რას ნიშნავს უკან დაბრუნება, ან რატომ უნდა დაბრუნდეს, ან არ უნდა დაბრუნდეს გადამზიდლისამუშაო ადგილას? ამ ქცევას განაპირობებს მისი წინა

ქცევა - დარჩა კი წასადები სიმბოლოები? თუ დარჩა, უნდა დაბრუნდეს მათ წასადებად, ხოლო თუ ბოლო ნაჭერი წაიღო საცავში, მისი დაბინაების შემდეგ უნდაგააგრძელოს ძებნა სხვა ნაჭრებისთვის. შეიძლება ვიფიქროთ, რომ ნებისმიერ, უკან დაბრუნებულ ჭიანჭველა გადამზიდს დარჩენილი ნაჭრები არ დახვდეს, ანუ იგი კონკურენციის მსხვერპლი გახდეს, თუმცა ამ პრობლემის ოპტიმიზაციაც შეიძლება. თუ ნებისმიერი ჭიანჭველა-გადამზიდი დატოვებს ცოდნას იმის შესახებ, დარჩა თუ არა სიმბოლოები ფოთლის მიდამოში, ნებისმიერ შემდეგ მომსვლელს პირდაპირი გზით გადაეცემა ეს ცოდნა. შესაბამისად, ისინი უკვე გააგრძელებენ სხვა ნაჭრების ძებნას და არა არსებულ ზონაში დარჩენილი ნაჭრების მოძებნას.

მე-3 პირობას რაც შეეხება, იმისათვის, რომ დავაოპტიმიზიროთ პირობის ფუნქცია, საჭიროა გადამზიდს გარკვეული სახით დახვდეს ცოდნა უკვე მოჭრილი ნაჭრების რაოდენობის შესახებ. იგი პირდაპირ მიმართავს შემოწმების ფუნქციას კონკრეტული რაოდენობით და არ დაიწყებს ხელახალ ძებნას. ამ ცოდნას კი, რასაკვირველია, პირდაპირი გზით მიიღებს წინამორბედი ფოთლის მჭრელიდან. როგორც ვახსენეთ, ნებისმიერი ფოთლის მჭრელი ტოვებს ინფორმაციას კონკრეტულ ადგილას მოჭრილი სიმბოლოების შესახებ.

#### თავი 4. ობიექტთა გამოცნობა ხელოვნურ ინტელექტში

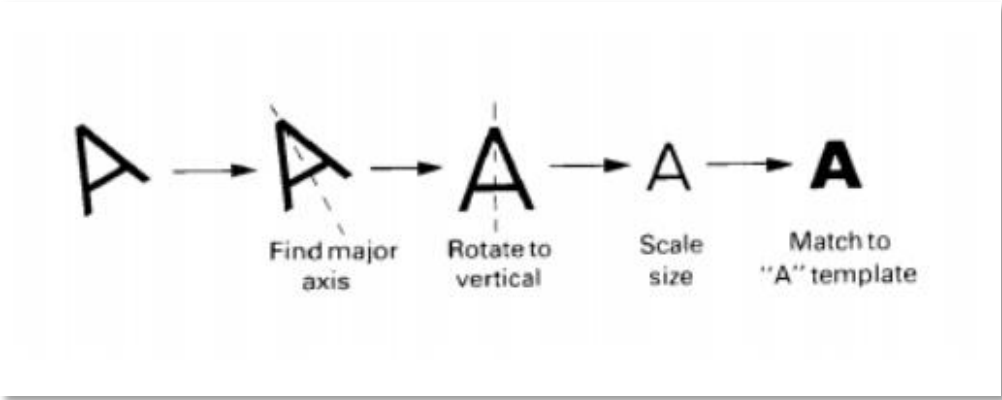
ჭიანჭველების ამოცანაში ფოთლის მჭრელი ჭიანჭველის მოქმედების ძირითადი მექანიკა, სიმბოლოთა ამოცნობაში მდგომარეობს. სიმბოლოებისგან შედგება ფოთლები, რომლის ამოცნობაც და დაჭრაც ხდება. განვიხილოთ სიმბოლოთა ამოცნობის ამოცანა.

ყველაზე მარტივი ახსნა თუ როგორ ხდება ასო-ციფრული ელემენტების ამოცნობა არის

შაბლონური დამთხვევის მეშვეობით. ყოველიზნასოსთვის ან ციფრისთვის იარსებებს შაბლონი, რომელიც ინახებაგრძელვადიან მეხსიერებაში. შემომავალი ნიმუში შედარდება შენახულ

შაბლონების სიმრავლესთან და თუ იარსებებს საკმარისი დამთხვევა მაშინნიმუშს მიენიჭება შესაბამისი შაბლონის შესაბამისი კლასი. ესეთსტრუქტურაში, ნიმუშის ზომაში ან კუთხეში მცირედი ცვლილებამოგვარდება სტანდარტიზაციის და ნორმალიზაციის პროცესების

მეშვეობით.მაგალითად: შესაძლოა მოხდეს ყველა ნიმუშის ატრიალება ისე, რომ მათი აბსცა გახდეს ვერტიკალური. დამატებით შესაძლოა გახდესსაჭირო სურათის წინასწარი დამუშავება / გასუფთავება, რადგან როგორცადამიანებს ისევე ცხოველებს შეუძლიათ ნიმუშის დაზიანებული ანგაურკვევლი ნაწილების ამოცნობა / შევსება (სურ. 27).



სურ. 27 სანამ დაიწყება შაბლონის დამთხვევა, ნიმუშს შესაძლოა შეუცვალონ ორიენტაცია, ან ზომა.

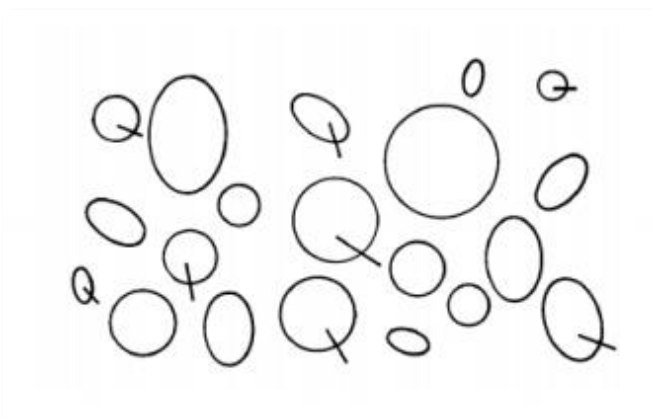
ამ ტიპის შაბლონური დამთხვევის სქემა იმ შემთხვევაში იმუშავებდა თუ ნორმალიზაციის პროცედურა იქნებოდა საკმარისი შედეგების ნიმუშების ცალსახად გარდასაქმნელად. სამწუხაროდ აღნიშნულის

მიღწევა არის თითქმის შეუძლებელი, ასო-რიცხვითი ელემენტების მარტივ შემთხვევაშიც კი. ასო A -ს შაბლონი შესაძლოა უფრო მეტად დაემთხვეს ასო R-ის შაბლონს ვიდრე თავს საკუთარ შაბლონს და პირიქით (სურ. 28).



სურ. 28 სურათზე ნაჩვენებია თუ როგორ შეიძლება ასო A-ს შაბლონი დაემთხვეს ასო R-ს შაბლონს და პირიქით

ის ხაზი რომელიც ასო Q-ს ანსხვავებს ასო O-სგან შესაძლოა განთავსებული იყოს რამოდენიმე წერტილში (სურ. 29).



სურ. 29 ასო Q და ასო O-ს სიმბოლოთა ვარიანტები რომელთა გარჩევაც ხდება

შაბლონური დამთხვევის სქემა მუშაობს ძალიან წარმატებულად თუ ამოსაცნობი სიმბოლოები არის სტატიკურად (ხისტად) გაწერილი. შედეგად კომპიუტერები, რომლებსაც ევალებათ ჩეკის ბოლოში დაწერილი ანგარიშის ნომრების წაკითხვა მარტივად ახორციელებენ ამ ოპერაციას მათში ჩაწერილი შაბლონებთან შედარებით. ამ მაგალითში

რამე სახის გაურკვევლობის თვიდან ასაცილებლად სიმბოლოებს მუდამ აქვთ 38ერთი ფორმა და არიან ერთმანეთისგან მაქსიმალურად განსხვავებულები. სიმბოლოები, რომლებსაც ადამიანი ცნობს არ არიან ესე ერთმანეთისგან განსხვავებულები. თუ ჩვენ დაეფიქრებით თუ როგორ ახერხებს ადამიანი A-ს და R-ს ან Q-ს და O-ს ერთმანეთისგან გარჩევას დაინახავთ რომ უნდა არსებობდეს გარკვეული კრიტიკული თვისებები, რომლითაც ამ სიმბოლოებს ერთმანეთისგან განვასხვავებდით. ხაზი რომელიც კვთავს ასო Q-ს არის ძალიან მნიშვნელოვანი O-სგან განმასხვავებელი ელემენტი. ამავედროულად ამ ელემენტების განსხვავებისას წრის ფორმას უფრო ნაკლები დატვირთვა აქვს. შესაძლოა მოდელი, რომლითაც ხდება თვისებების კომბინაციების დადგენა უფრო წარმატებული იყოს ვიდრე შაბლონებზე დაფუძნებული მოდელის. თვისებების ანალიზის მოდელი იყო ძალიან პოპულარული ფსიქოლოგიასა და კომპიუტერულ მეცნიერებებში 1960-იანი წლების განმავლობაში. ამ დროს შეიქმნა სავარაუდოდ ფსიქოლოგიისთვის ყველაზე წარმატებული სელფორიჯის პანდემონიუმის სისტემა, რომელიც თვიდან წარმოადგენდა კომპიუტერულ პროგრამას, რომელსაც შეეძლოა მორხეს სიგნალების ამოცნობა, მაგრამ შემდგომში პოპულარობა ჰპოვა როგორც ასო-რიცხვითი სიმბოლოების ამომცნობმა მოდელმა, რომელიც შექმნა ნეისერმა (1967) და ლინდეიმ და ნორმანმა (1972).

სისტემა შედგება სხვადასხვა რაოდენობის და სხვადასხვა კლასის „დემონებისგან“. ჩვენი მიზნებისთვის ყველაზე მნიშვნელოვანი არიან თვისების დემონი და შემეცნებითი დემონი. თვისების დემონი არჩევითან პახუბს როცა მას მიეოდება კონკრეტული ლოკალური კონფიგურაცია (მარჯვენა კუთხე, ვერტიკალური ხაზი...). ხოლო შემეცნებითი დემონი, წარმოადგენს კონკრეტულ ასოებს და თვისებების დემონთან ეძებს თვისებების კონკრეტულ კომბინაციას. შედეგად შემეცნებითი დემონი, რომელიც წარმოადგენს ასო H -ს შესაძლოა ეძებდეს ორ ვერტიკალურ და ერთ ჰორიზონტალურ ხაზს და დამაბით ითხოვდეს ოთხი მართი კუთხის არსებობას. რაც უფრო მეტი საჭირო თვისებების მოძიება მოხდება, უფრო ხმამაღლა შეძლებს შემეცნებითი დემონი იმის

ყვირილს, რომ ის წარმოადგენს დამთხვევის უმაღლეს ალბათობას. ხოლო არჩევანის გამკეთებელი დემონი ირჩევს ისეთ შემეცნებით დემონს, რომელიც ყველაზე ხმა მაღლა ყვირის. შედეგად, ამ სისტემაში თითოეული სიმბოლო წარმოდგენილია როგორც კრიტიკული თვისებების მქონე სიმრავლე და ყოველი სურათის პროცესირება ხდება იერარქიული სტრუქტურით აბსტრაქციის სხვადასხვა დონეებზე. ამ მოდელში ბარლოუ და სხვები გულისხმობდნენ რომ ტვინის მარტივი უჯრედები ასრულებდნენ პანდომინუმის სისტემის თვისების დემონების მოვალეობას და აწვდიდნენ ინფორმაციას სხვა უჯრედებს, რომლებიც შესაბამისად მოექცეოდნენ გადაწოდებულ აბსტრაქტულ პარამეტრებს. ესეთი ჰიპოთეტურ უჯრედებს ეძახდნენ „ბებია უჯრედებს“ ან „ყვითელი ვოლცვაგენის დეტექტორებს“, რათა ყურადღება გაემხავილებინათ იმ აბსტრაქტულ მასტიმულირებელ გარემოებებზე, რომლებზეც ისინი მოქმედებდნენ. პანდომინუმის სისტემას შეუძლია სხვადასხვა თვისებას მიანიჭოს სხვადასხვა რანგი, იმისდა მიხედვით თუ რამდენად კარგად ჯდება ეს თვისებები კონკრეტულ ნიმუშებში. ამ სისტემის გარდა არსებობენ კიდევსხვა სისტემები, რომლებიც ზემოთ აღწერილი სისტემის ანალოგიურად სწავლობენ რანგის ცვლილებას სტიმულსა და სტიმულზეგანხორციელებული რეაქციის მიხედვით. პანდომინუმის ტიპის სისტემას ასევე შეუძლია გარკვეული შინაარსიდან შესაბამისი დასკვნების გამოტანა. მაგალითისთვის სურათ 30-ზე ნაჩვენებია მაგალითი, თუ როგორ მოქმედებს შინაარსი ადამიანის ასოების აღქმის უნარზე.



სურ. 30 ადამიანის ასოების აღქმის უნარის შინაარსი

იმისდა მიხედვით თუ რომელი ასოების გარემოცვაშია მოქცეული, ერთი და იგივე ფორმა შესაძლოა იქნას დანახული როგორც ასო A ასევე ასო A -ად. პანდომინუმის სისტემაში, ამ ფაქტებზე დაყრდნობით ჩვენ შეგვიძლია ზედა - დონის დემონებს უფლება დავართოთ რომ

გამოაცოცხლონ ის ქვედა დონის დემონები, რომლებიც შეესაბამებიან შესაბამის ნიმუშს რათა მათ უფრო ხმამაღლა დაუძახონ გადაწყვეტილების მიმღებ დემონს. თუმცა როგორც ადამიანის ნიმუშის და ობიექტების ამოცნობის ზოგადი მოდელი პანდომონიუმის სისტემა არ არის დამაკმაყოფილებელი. ძირითადად იმიტომ რომ ის დაფუძნებულია ნიმუშებში არსებული თვისებების სიმრავლის აღწერებზე, რომლებიც თავის მხრივ არიან მინი-შაბლონების ანალოგიები. ერთ-ერთი საბაბი რის გამოც პანდომონიუმი იყო პოპულარული იყო მისი თავსებდაობა ნეუროფსიქოლოგიასთან. თუმცა ჩვენ უკვე ვნახეთ რომ ცალი უჯრედი ვერ ჩაითვლება „თვისების დამდგენად“. ამას მართალია არ აქვს მნიშვნელობა წმინდა ფსიქოლოგიური და კომპიუტერული მეცნიერებების თვალთაზრისით, მაგრამ არსებობს სხვა პრობლემები. თვისებების აღმწერ სიას არ შეუძლია სრულად გადმოსცეს დანახული დეტალების სრული სტრუქტურა. შესაბამისად, პანდომონიუმის სისტემას შესაძლოა სიმბოლოები შეეშალოს უფრო იშვიათად გამოყენებად სიმბოლოებთან. მაგალითად : E და  $\Sigma$  ანუ ისეთი ტიპის შეცდომები, რომლებიც ზოგადად ადამიანებს არ შეეშლებოდათ. ამასთან ერთად, პანდომონიუმის სისტემა არ ითვალისწინებს ერთი ნიმუშის სხვადასხვა მაგალითის ინფორმაციას ანუ გადაწყვეტილების მიმღები დემონის შედეგი მუდამ იქნება ერთი იმისდა მიუხედავად ასო A -ს რომელი ვერსია იყონაჩვენები. საჭიროა არსებობდეს გზა, რომლითაც ჩვენ შევძლებდით განგვესხვავებინა სხვადასხვა ნიმუშების სხვაობა და ამასთან ერთად შევძლებდით ერთიდაიგივე ტიპის ნიმუშების დაჯგუფებას.

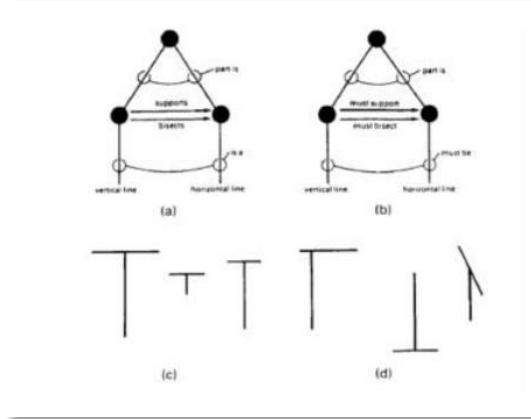
#### 4.1 სტრუქტურული აღმწერები

ზოგადი და მოხერხებული ფორმატი ადამიანის ნიმუშის და ობიექტების ამოსაცნობად მოცემულია სტრუქტურული აღმწერი ენებისმეშვეობით. სტრუქტურული აღმწერები არ ეხმარებიან თეორიას რომელიცადგენს თუ როგორ ხდება ცნობა, ისინი უბრალოდ გვაძლევენ გამოხატვისსწორ ტიპს, რომელზეც შესაძლებელია აღნიშნული თეორიის აგება. სტრუქტურული აღმწერები გვაძლევენ თეორიების



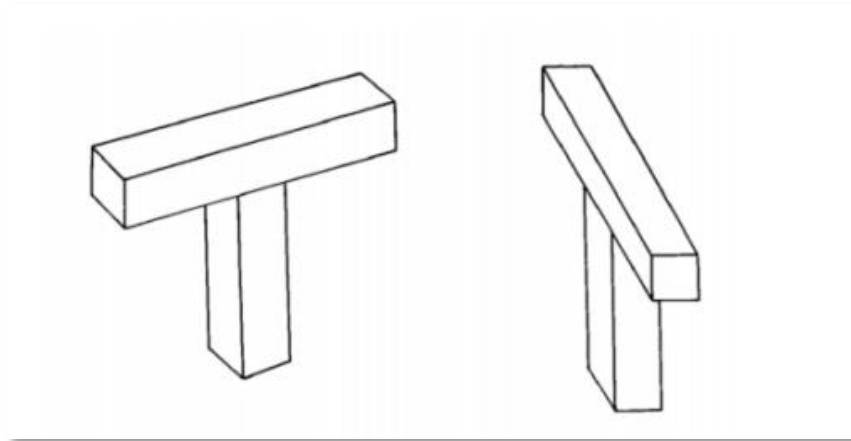
სიმრავლესკონკრეტული კონფიგურაციის შესახებ. ესეთი თეორიები აღწერენკონფიგურაციის კომპონენტის ბუნებას და წვრილმანდებიან ამ ნაწილების

სტრუქტურული განლაგებებით. შესაბამისად ასო T-ს სტრუქტურული აღწერა, შესაძლოა გამოიყურებოდეს შემდეგნაირად:



სურ. 31 ასო T -ს სტრუქტურული აღწერა

სტრუქტურული აღწერის ენის გამოყენებით შესაძლებელია კონკრეტული კონცეფციის და კატეგორიის მოდელის აგება, რომელიც შეძლებს ნებისმიერი შემომავალი ობიექტისთვის შესაბამისი კლასის დადგენას. ესეთი მოდელები ავლენენ სტრუქტურის აუცილებელ თვისებებს, მაგრამ შესაძლოა ნაკლებ ყურადღებას უთმობდეს სხვა დეტალებს. შესაბამისად ასო T-ს მოდელი შესაძლოა გამოიყურებოდეს როგორც სურათი 31 ბ ნახაზი, ხოლო სურათი 30 ც-ზე ნაჩვენებია ის მოდელები, რომლებიც შესაძლოა კატეგორიზირდეს როგორც ასო თ, ხოლო იგივე ნახაზის ც-ში ნაჩვენებია მოდელები, რომლებიც არ გამოდგება ასო T-ს აღწერად. სტრუქტურული აღმწერების მორგება უფრო მარტივია ობიექტების ამოცნობაზე ვიდრე შაბლონების ან თვისებების გამოსახულებებზე. ობიექტის სურათი შესაძლოა აღწერილი იყოს სხვადასხვა დონის აბსტრაქციის მქონე სტრუქტურული აღმწერების მიხედვით. თითოეული ეს დონე წარმოადგენს აღწერის შესაძლო „დომენებს“. აღნიშნული მაგალითის განსახილველად ავიდეთ სურათი 32.



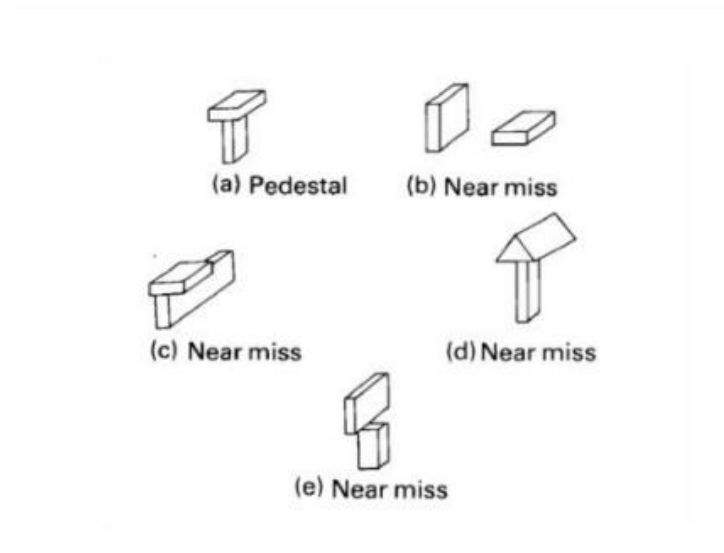
სურ. 32 ასო T-ს სამგანზომილებიანი ფიგურა

ეს სურათი შესაძლოა აღწერილი იქნას სხვადასხვა უნიკალური დომეინის მიხედვით, რომელთა ზოგადი კატეგორიზირება შესაძლებელია მოხდეს როგორც „2 განზომილებიანი“ ან „3 განზომილებიანი“. 2 განზომილებიანი აღმწერი გადმოსცემს მიცემულ სურათს და ამ სურათის აღწერა შესაძლოა მოხდეს ძალიან აბსტრაქტული და ზოგადი ტერმინებით. ის შესაძლოა აღიწეროს როგორც სხვადასხვა სიკაშკაშის მქონე წერტილების ერთობლიობა, ან როგორც ხაზების ერთობლიობა და ა.შ. მნიშვნელობა არ აქვს თუ როგორი და რა დონის „2 განზომილებიანი“ აღწერა იქნება მოცემული ზემოთ ნაჩვენები ფიგურების, ამ აღწერების მიხედვით ზემოთ ნაჩვენები ორი ფიგურის გამოსატყულება იქნება ერთმანეთისგან ძალიან განსხვავებული. ეს მხოლოდ “3 განზომილებიანი” აღწერის მოდელშია შესაძლებელი რომ ამ ორ ფიგურას შორის მოხდეს ტოლფარდობის დაფიქსირება, რადგან ამ ტიპის აღმწერები გადმოსცემენ ობიექტის წყობის და აგებულების აღწერებს. შესაბამისად, ზემოთ ნაჩვენები 2 ნახაზიერთამენთის ანალოგიურია მხოლოდ ობიექტების აღწერის დონეზე, რომელიც არ არის დამოკიდებული დაკვირვების წერტილზე. შედეგად ერთი და იგივე ობიექტის სხვადასხვა ვიზუალს სურათების დომეინში ექნება განსხვავებული სტრუქტურული აღწერა, მაგრამ იქნებიან ანალოგიურები ობიექტების დომეინში. ჩვენი პრობლემა ეხლა მდგომარეობს იმაში რომ გავარკვიოთ როგორ უნდა მოხდეს სტრუქტურული აღწერების აგება 3D დონეზე, მათი შედარება და სასურველი შედეგის მიღება. ვინსტონმა 1975 წელს, წარმოადგინა

ადრეული ილუსტრაცია იმისა თუ როგორ არის შესაძლებელი სტრუქტურული აღმწერების გამოყენება ობიექტის ამოცნობისას. აღნიშნულით მან გვაჩვენა თუ როგორ შეიძლება მაგალითებზე სწავლების გზით მოხდეს ობიექტებზე საერთო წარმოდგენის ჩამოყალიბება. მან შექმნა პროგრამა რომელსაც შეეძლო ესწავლა მარტივი სათამაშოების სტრუქტურის ცნობა (სურ.33). კომპიუტერულ პროგრამას გადაეწოდებოდა თითოეული ფორმის სათამაშოს ნიმუში და ასევე ამ ფორმასთან მიახლოებული ნიმუშები. სწავლების პროცესი იყო შემდგომი: თავდაპირველად მას გადაეწოდებოდა შესასწავლი სათამაშოს ასლი ნიმუში მას პროგრამა მიაჩვენებდა შესაბამის სტრუქტურულ აღწერას



სურ. 33 სათამაშოების ნიმუში, რომელთა ცნობაც შეეძლო ვინსტონის პროგრამას



სურ. 34 ტრენინგის მიმდევრობა

შემდეგში პროგრამას გადაეწოდებოდა პედესტალი ფორმასთან მიახლოებული ფორმის მქონე ფიგურები, რომლებიც არ წარმოადგენენ პედესტალს (სურ: 34 ნახაზი b). მნიშვნელოვანი ამ გადაცემულ

სათამაშოებში იყო ის, რომ ყველა ფიგურა შედგებოდა ორი ნაწილისგან: აგურისგან და დაფისგან. შესაბამისად სისტემას პედესტალის ფორმის მქონე ობიექტის და არა-პედესტალის ფორმის მქონე ობიექტების შედარებით, უკვე შეეძლო აეგო პედესტალის მოდელი, რომელშიც ნაჩვენები იქნებოდა თუ რა პირობებს უნდა აკმაყოფილებდეს ერთმანეთან მიმართებაში ფიგურის ნაწილები, რათა ფიგურას დაერქვას პედესტალი. ობიექტს რომ დაერქვას პედესტალი სავალდებულოა რომ „მისი ერთი ნაწილი მყარდებოდეს მისი მეორე ნაწილისგან“, „ჰორიზონტალურ მდგომარეობაში მყოფი დაფა უნდა იდგეს ვერტიკალურ მდგომარეობაში მყოფ აგურზე“.

ჩვენ სპეციალურად გაჩვენეთ სწავლების მაგალითი პედესტალის ნიმუშზე, რადგან მას გააჩნია ასო T-სთან მიახლოებული 3 განზომილებიანი ფორმა, შესაბამისად ასო T-ს 3 განზომილებიანი სტრუქტურული აღწერა იქნება პედესტალის აღწერასთან ძალიან მიახლოებული მხოლოდ ერთი სხვაობით. პედესტალის შემთხვევაში მისი შემდგენი ნაწილებიც თავის მხრივ წარმოადგენენ 3 განზომილებიან ობიექტს, ხოლო ასო t-ს შემთხვევაში მისი შემდგენები არიან ჰორიზონტალური და ვერტიკალური ხაზები.

დავუბრუნდეთ ვინსტონის პროგრამას. პედესტალის ამოცნობის იდენტური პროცესის გამოყენება შესაძლებელია მოხდეს სხვა ფიგურების, მაგალითად სურათზე ნაჩვენები სახლის ან არკის სათამაშოს ობიექტების დასადგენად.

შედეგად ვხედავთ რომ ვინსტონის ობიექტი ოპერირებს ობიექტების დომეინში. მას შეუძლია მიიღოს აგურის ან დაფის პროექცია და მათ მიანიჭოს შესაბამისი კატეგორია. თუმცა თავად აგურის და დაფის სტრუქტურული აღწერა უნდა იყოს განსაზღვრული პროგრამის სხვა დონეზე. სწორედ ეს დაბალი დონის პროგრამები, რომლებიც ამუშავებენ აღწერებს და გადასცემენ ინფორმაციას შემსწავლელ პროგრამას წარმოადგენენ ვინსტონის სისტემის პრობლემას. ჩვენ გვჭირდება უკეტესი გზა, რომლითაც ვინსტონის ტიპის პროცედურებისთვის შევძლებდით საკმაირის რაოდენობის სტრუქტურული აღწერების მომარაგებას, რომელიც არ იქნებოდა შეზღუდული ხელოვნური სამყაროს ფარგლებში. ამისთვის ჩვენ უნდა

დავუბრუნდეთ ობიექტების ამოცნობის ფუნდამენტალურ პრობლემას. გავისხენოთ რომ ობიექტების პროექცია

რეტინაზე დამოკიდებულია დაკვირვების წერტილზე. შესაბამისად თუ ჩვენ დაეყრდნობოდით დამკვირვებლის კუთხიდან აგებულ ობიექტის აღწერაზე, ჩვენ დაგვჭირდებოდა ბევრი სხავდასხვა დაკვირვების წერტილების შესაბამისი აღწერების შენახვა.

#### 4.2 ობიექტების გრაფებად წარმოდგენა და ამოცნობის სხავდასხვა მიდგომები

პათერნი[15] არის მრავალჯერ გამოყენებადი პრობლემის გადაჭრის ხერხი. პათერნები წარმოდგენენ შემდეგი სახის აღწერებს:

<პათერნი> ::= <<თვისება><მნიშვნელობა>?<ვალიდურობა>>\*

<<მიმართება><მნიშვნელობა>?<ვალიდურობა>>\*

<მიმართება> ::= <ISA მიმართება>|<კომპოზიციის მიმართება> |

<აგრეგირების მიმართება>

<თვისება> ::= თვისების აღმნიშვნელი ტერმი

<მნიშვნელობა> : :=  $R^n$  | დისკრეტული სიმრავლე

<ვალიდურობა> ::= {true,false} | [0,1]C R

<კომპოზიციის მიმართება> ::= მთელი-ნაწილი მიმართება

<აგრეგირების მიმართება> ::= ასოცირების მიმართება

მოცემულია გარკვეული რაოდენობის კლასების (ცნებების) სიმრავლე  $\{C_1, C_2, C_3, \dots, C_n\}$ . თითოეული  $C_i$  კლასისთვის გვაქვს გარკვეული რაოდენობის პათერნები  $\{P_{i1}, P_{i2}, P_{i3}, \dots, P_{ik}\}$  გამოსაცნობი ობიექტის აღწერა უნდა მოიძებნოს გამოსაცნობ  $X$  ობიექტთან ყველაზე ახლო მდგომი პათერნი და შესაბამის კლასის ინდექსი:

$$i^* = \min \text{Dist}(X, P_{i1});$$

$$1 \leq i \leq n$$

$$1 \leq j \leq ik$$

მანძილის გამოსათვლელად თითოეული პათერნი წარმოდგება მრავალშრიანი გრაფის სახით. თითოეულ შრეში თავსდება გრაფი რომელიც შედგენილია ასოცირების მიმართებით შედგენილი კლასების წვეროებისგან და თვისებების მნიშვნელობების წვეროებისგან. ხოლო

ყოველ მომდევნო შრეში თავსდება მიმდინარე შრის კლასის წევრობის შემადგენელი კლასების წევროებისგან, მათი თვისებათა მნიშვნელობების და მიმართებით დაკავშირებული კლასებისგან. ანალოგიურად ხდება გამოსაცნობი წევროების წარმოდგენა .

გამოცნობის პროცესი ამ შემთხვევაში წარმოადგენს ერთი ერთი დონის შრის ობიექტის და პათერნის გრაფების შეთავსების ამოცანას, რომელიც ექვივალენტურია ორი გრაფის მაქსიმალური საერთო ქვეგრაფის მოძებნის ამოცანისა. რთული ობიექტების გამოცნობისთვის არის შემდგომი მიდგომები:

1. კლასიკური ქვევიდან -ზევით, როცა განისაზღვრება ჯერ ქვედა დონეზე ობიექტის შემადგენელი ნაწილების მათ პატერნებთან სიახლოვე და შემდგომ მშობელი ნაწილების სიახლოვე და ასე შემდეგ სანამ არ მივაღწევთ საწყის ობიექტს.

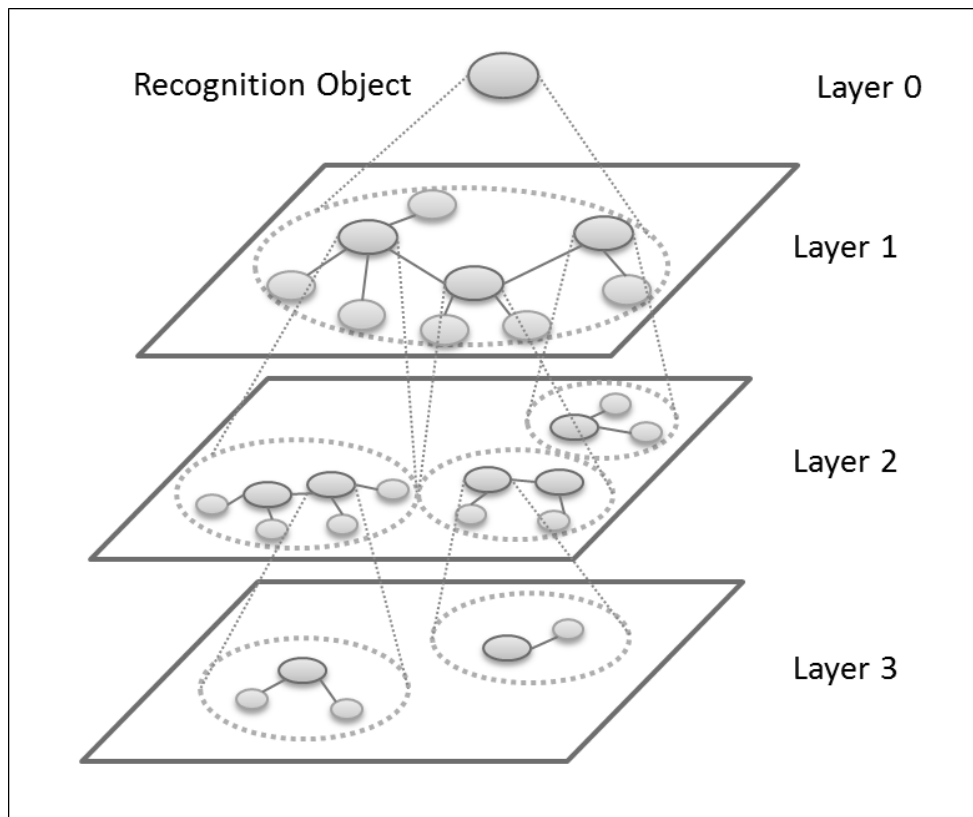
$$M(x, Y) = \min_{t_{ij} \in U, c_i \in Y, T_{ci}} (Dist(g(x), g(t_{ij})) + \sum_{x_i \in S(x)} M(x_i, \sigma(c_i)))$$

სადაც  $T_{c_i} = \{t_{i1}, t_{i2}, t_{i3}, \dots, t_{ik}\}$  არის ნიმუში  $c_i$  კლასის, ხოლო  $Dist(g(), g())$  ფასდება როგორც ორი გრაფის შედარება და ეს ამოცანა არის NP-Complexity.  $\sigma(c_i) = \{c_{i1}, c_{i2}, c_{i3}, \dots, c_{im}\}$  კლასების სია  $c_i$ ,  $g(x)$  გრაფი, რომელიც შედგება თვისებებით და კავშირებით  $X$ -ის რეკურსიის დაწყების პირობა  $M(o, C)$ , სადაც  $C = \{c_1, c_2, c_3, \dots, c_m\}$  არის სია ნასწავლი კლასების. ხოლო რეკურსიის დასრულების პირობაა  $\sigma(c_i) = \emptyset$ . ოპტიმალური მნიშვნელობა აკმაყოფილებს ტოლობას  $opt\{M(o, C)\} = M(o, \{c_i^*\})$ .

რთული ობიექტის და განუზღვრელობების შემთხვევაში გადასატრედი ამოცანა ძალიან ბევრ გამოთვლით რესურსებს თხოულობს და პრაქტიკულად უმეტეს შემთხვევაში შეუძლებელია მისი გამოყენება.

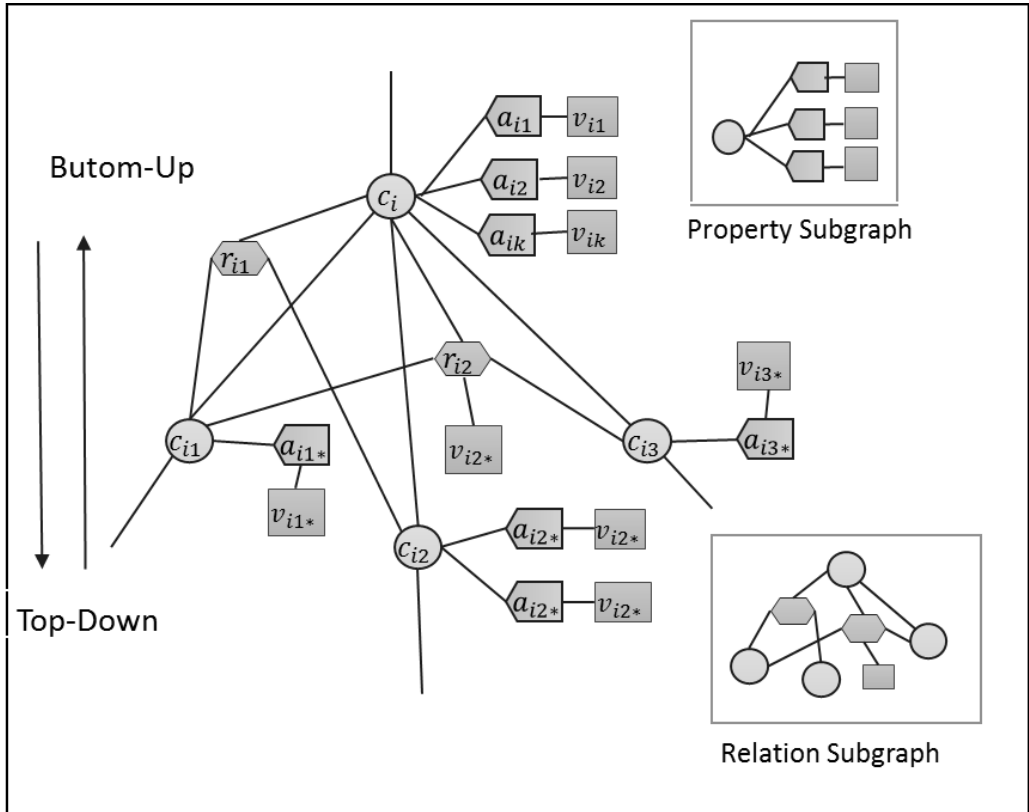
მეორე მიდგომაა ზევიდან-ქვევით(სურ 35), როცა ზედა დონეზე გამოითქმებაჰიპოთეზა და შემდგომ ეს ჰიპოთეზა მოწმდება ობიექტის შემადგენელი ნაწილებზე, შემდგომ მათ შემადგენელ ნაწილებზე და ასე შემდეგ სულ მცირე ნაწილებამდე. ეს მეთოდი მოითხოვს აპრიორულ ცოდნას საწყისი ობიექტის შესახებ, რათა სწორედ შედგეს

ჰიპოთეზა , მაგრამ რთული ობიექტებისთვის ამ აპრიორული ინფორმაციის მოპოვება როგორც წესი რთულია.



სურ. 35 ზევიდან-ქვევით ტიპის გამოცნობის არქიტექტურა

კომბინირებულია მიდგომა(სურ 36) , როცა გამოიყენება ერთდროულად ორივე ზევით მოყვანილი მეთოდი. ეს მეთოდი ხასიათდება შემდეგი თვისებებით: ქვევიდან-ზევითგამოცნობის მეთოდი უზრუნველყოფს მაღალ პარარელიზმს , ამიტომ შესაძლებელია მრავალ აგენტური მოდელის გამოყენება ზევიდან-ქვევით გამოცნობისმეთოდი მოითხოვს ცნებებისსემანტიკასთან მუშაობას და დასკვნების მექანიზმის გამოყენებას. ამიტომ აგენტებს უნდა ჰქონდეთ ონთოლოგიაზე - დაფუძნებული არქიტექტურა, დასკვნების კეთების და სემანტიკური სიახლოვის განსაზღვრის მექანიზმებით.



სურ. 36 კომბინირებული ტიპს გამოცნობის არქიტექტურა



## **თავი 5. არგუმენტაციის მექანიზმი ვიდეოთამაშებში.**

ვიდეოთამაშების ხელოვნურ ინტელექტში დიდი როლი უჭირავს კოლექტიურ გადაწყვეტილებებს. რამდენიმე აგენტის გაერთიანებამ შეიძლება სხვა სახე მისცეს თამაშს. მრავალაგენტიან სისტემასთან მოთამაშის ინტერაქცია ბევრად საინტერესო გამოწვევაა. ამ შემთხვევაში აგენტები ერთმანეთთან ურთიერთქმედებენ და იღებენ საერთო გადაწყვეტილებას. ეს გადაწყვეტილება კი გამომდინარეობს მათი ინდივიდუალური გადაწყვეტილებებიდან, რომლებსაც პირობითად, არგუმენტები ეწოდება. არგუმენტები დაფუძნებულია ცოდნის ბაზაზე, აგენტის პირად გამოცდილებაზე, რომელიც მან მოქმედების დროს დააგროვა. აღვწერთ არგუმენტაციის ტიპები, განვიხილოთ კონკრეტული არგუმენტაციის მექანიზმი ჭიანჭველათა ამოცანაში, სადაც არგუმენტაცია ჭიანჭველა-აგენტთა გაერთიანებათა მთავარი მექანიზმია.

### **5.1 არგუმენტაციის მექანიზმი ხელოვნურ ინტელექტში**

არგუმენტებით სწავლება ერთ-ერთი ძირითადი მეთოდია ხელოვნურ ინტელექტში, არგუმენტების ლოგიკაზე დაფუძნებული მოდელი უმთავრესია კომპლიქტების მოგვარების, მიზნის მიღწევის და გადაწყვეტილების მიღებისთვის. არსებითი დიალექტიკური ხასიათი ამოდელისა გადმოგვცეს დამფუძნებლებმა დიალოგზე დაფუძნებული არგუმენტების ფორმალიზაციისთვის, მაგალითად, როცა აგენტი ცდილობს დაარწმუნოს რწმენაში სხვა პერსონა ის უკვე დარწმუნებულია ჭეშმარიტებაში, ან როცა აგენტი მიზანმიმართულია რადაცის გაკეთებისკენის აწარმოებს მოლაპარაკებებს, ბოლოდროინდელი კვლევითი პროექტია ვითარებს ზოგად მოდელს არგუმენტაციისას, რომელსაც დაფუძნებულია დასკვნასა და გადაწყვეტილებაზე [8,11]. ბევრი თეორიული და პრაქტიკული დეველოპმენტი დაფუძნებულია სემინალური არგუმენტების თეორიაზე. უნგ არგუმენტაციის თეორია მიმართულია გრაფისკენ, რომელიც შედგება არგუმენტების ნაკრებისგან არგს და ბინარული შეტევა დ თანდაკავშირებით, ამ ფრეიმვორკის არგუმენტები განისაზღვრება როგორც სხვადასხვა სემანტიკა სადაც სენამტიკის

არჩევანი უტოლდება სხვადასხვახარისხის სეპტიციზმის. საკითხის გაფართოება განისაზღვრება „მისაღებიარცხვებით“ როცა არგუმენტი  $X \in \text{Args}$  ამბობს რომ მისაღებია  $S \in \text{Args}$  მაშინ როცა  $y$  უტევს  $x$  თითონ გამოდის დამარცხებული  $z$  არგუმენტით „მაგალითად თუ  $s_i$  არის მაქსიმალური წარმოდგენს ყველა შემავალარგუმენტს რომელიც აკმაყოფილებს  $s_i$  მაშინ როცა  $s_i$  ამბობს რომ ისიქნება გაგრძელება რჩეული სემანტიკის. ბოლო წლების ინტენსიურსწავლებას აქვს მრავალი მიმართულება, ის შეიძლება აისხნასარგუმენტების ფარგლებში აბსტრაქტული ხასიათით და კოდირებისსინტუციით. ხაზგასმული და ზემოთ აღნიშნული ლოგიკა კრავსარგუმენტებს არგს და შეტევა  $R$  ზე რჩება განუსაზღვრელი, ამგვარადსაშუალება გვეძლევა ლოგიკის სხვადასხვა ფორმალიზაციისვის. თეორიის დასკვნების შემდეგ შეიძლება განისაზღვროს პრეტენზიები არგუმენტებისა,არგუმენტები ძირითადად არის მტკიცებულება კანდიდატის დასკვნისა-არგუმენტი მოიცავს გაგებად ლოგიკას. დანკის თეორია ამიტომ შეიძლებაგაგებული იქნას სემანტიკად, მონოტონურ მსჯელობად. ამ გაგებით მიზანიარაა მხოლოდ სწორი დასკვნის გაკეთება არამედ დაუპირისპირდებოპონენტს, სად არის აპლიკაცია რომელიც გამოითვლის ლოგიკას,რაციონალობა ნიშნავს დაამკვიდრო სისწორის სტანდარტი. მართლაც ბევრილოგიკა პროგრამირების შეესაბამება დაგის თეორიას, დაგის სემანტიკასშეიძლება მრავალი გაგრძელება ქონდეს განსაკუთრებით არგუმენტებისთეორიას, თუმცა შეიძლება აღმოვჩნდეთ პრობლემის წინაშე როცა ისმენერთის არგუმენტს და უნდა მიიღო გადაწყვეტილება , რომელსაც არ აქვს გაგრძელება[9]. ილუსტრაციისთვის განვიხილოთ არგუმენტები, და ამინდის ამინდის პროგნოზი

1 : დღეს იქნება მშრალი ამინდი ლონდონში მაშინ როცა  $sy$  იმ გამოაცხადაა  $m_{sy} = A$

1: დღეს იწვიმებს ლონდონში მაშინ როცა CNN მა გამოაცხადა ნალექი =  $B$  და  $B$  აყენებენ ურთიერთგამომრიცხავ არგუმენტებს და ეპაექრებიანერთმანეთს (სემანტიკური პროლა)დაგის თეორიის თანახმად გვაქვს ორი ვარიანტი  $A$  და  $B$  რომელთაგანარცერთი არაა

გამართლებული, არსებობს პრობლემის გადაჭრის ერთი გზა, უპირატესობა მიანიჭო რომელიმე არგუმენტს, ზოგიერთი ამ მომენტშირთავს და ამართლებს დაგის თეორიას. დაგის თეორია ივრცობა უპირატესობის მინიჭების პრინციპით, ასე რომ შეტაკებისას X ის Y ზეწარმატებულია იმ შემთხვევაში თუ  $Y_i$  ვერ ჯობია X ს, დაგის ფრეიმვორკი დაფუძნებულია ღირებულებებზე და ღირებულებების მნიშვნელობაზე. ამინდის მაგალითი როლი რომელიც განვიხილეთ ზემოთ და და სუპირატესობა რადგან BBC უფრო სანდოა ვიდრე CNN, B e არ იყო წარმატებული მოიერიშე ვიდრე ამიტომ ამ ბრძოლის მაგივრად შეიძლება დავტოვოთ მხოლოდ {A} ეს მაგალითი არის ილუსტრაცია პრობლემის გადაჭრის არგუმენტებით. ხშირ შემთხვევაში აუცილებელია ინფორმაცია იმის სავარაუდოდ თუ ვინ იქნება გამარჯვებული, თუმცა ინფორმაცია უპირატესობის შესახებ იყოს არასწორი, შეფასება შეიძლება შეესაბამებოდეს კონტექსტს, რომელსაც აფასებს სანდო წყარო სხვადასხვა კრიტერიუმების მიხედვით, ამისათვის აუცილებელია სიტუაციის შეფასება სხვადასხვა კრიტერიუმებით, ამდენად გჭირდება კომფლიქტის უპირატესობის ინფორმაცია რომ ამტკიცო ზემოთ მოყვანილი ფაქტები. ილუსტრაციისთვის გავაგრძელოთ ზემოთ მოყვანილი მაგალითი ამინდის შესახებ :

2: მაგრამ BBC არის უფრო სანდო ვიდრე CNN = C

3: მაგრამ CNN ი უფრო კომპეტენტურია ამინდის პროგნოზში ვიდრე BBC =

C1

4: ზოგადი შედარება უფრო მნიშვნელოვანია ვიდრე ვიდრე პროგნოზის სანდოობასთან დაკავშირებული: E

C ს არგუმენტი არ ეწინააღმდეგება B ს თუმცა ის გამოხატავს არგუმენტს A სუპირატესობას B ზე , თუმცა C გამოხატავს უპირატესობას B ს A-ზე , C და C1 ეპაექრებიან ერთმანეთს რაც მათ გამოხატეს ურთიერთსაწინააღმდეგო წინადადებები, საბოლოოდ მოითხოვს C1 უპირატესობას C ზე დამხლოდ C1 ი ამარცხებს C ს,

ამგვარად ჩვენ გვაქვს სკეპტიკურისამართლიანი არგუმენტი C1 ის რომელიც მოითხოვს B ს უპირატესობას Aზე და B არის სკეპტიკურად მართალი .ამ თავში ჩვენ გვსურს განვაგრძოთ დაგის თეორია რომ ყურადღებაშივაქციოთ არგუმენტებს, გვერდი დაყოფილია შემდეგნაირად .განვიხილავთ დაგის თეორიას და არგუმენტებს, იმ გვერდს რომელსაც ჩვენმიყვებით ადვილად სწავლებას არგუმენტების ფორმალიზაციას.დაგის არგუმენტების თეორია გრძელდება რომ ინტეგრაცია მოხდესუპირატესობა არგუმენტებს შორის. გაფართოებული თეორია იცავს დაგისაბსტრაქტულ მიდგომას.არგუმენტები რომლებიც გამოხატავენ უპირატესობას არის მარტივიკვანძი გრაფაში, უპირატესობა აპლიკაციის არის აბსტრაქტული თვისება,ახალი შეტაკების განსაზღვრა რომელიც დაკავშირებულია არგუმენტებისუპირატესობასთან , შეტაკება არგუმენტებს შორის რომელიც იქნებაუპირატესობის მოთხოვნის საგანი, ამგვარად არ არის საგაღდებულიოცოდე უპირატესობის განსაზღვრა . უპირატესობის არგუმენტებიგამოხატავს საწინააღმდეგო უპირატესობას ოპონენტებს შორის, შემდეგჩვენ განვაგრძობს დანგის სტატიებს ასე რომ არგუმენტებს და შეტაკებებსჭირდება აღდგენა. ვრცელ სემანტიკები სემანტიკის გავრცობისთვისგანისაზღვრება ზუსტად ისე როგორც როგორც დანგის თეორია, ზოგიერთიზოგადი შედეგი რომელიც ახლავს ვრცელ სემანტიკს დანგის თეორიაშიასევე ნაჩვენებია ვრცელ ფრეიმვორკში. ჩვენი მიზანია საფუძველიჩავუყაროთ აპლიკაციის პირდაპირ მოდიფიკაციას და გაავითაროთდანგის არგუმენტებზე დაფუძნებული უპირატესობა. ვრცელი თეორიაგვაწვდის ზოგად პარამეტრებს ვისწავლოთ არგუმენტების უპირატესობა დაუფრო ზოგადად შეტაკებები, ამგვარად ზემოთ ნახსენები ფრეიმვორკიშეიძლება განვაგრძოთ და მხედველობაში მივიღოთ არგუმენტებირომელიც გამომდინარეობს პრობლემის გადაწყვეტილებიდან. აქ ჩვენგანვიხილავთ არგუმენტების ფრეიმვორკის ორ ძირითად კლასს.გაფართოებული თეორია შექმნილია იმისთვის რომ გააერთიანოსფრეიმვორკი რომელიც ავითარებს დანგის თეორიის უპირატესობისპრინციპს. ეს სექცია აღწერს როგორ შეიძლება გაფართოვდესიერარქიულად არგუმენტებზე დამყარებული

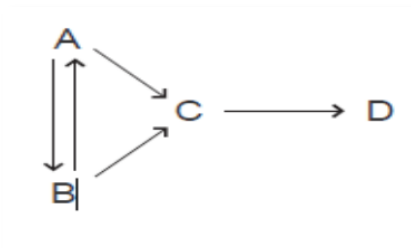
მნიშვნელობები. გაფართოებული თეორია არის შემოთავაზებული როგორც ობიექტის დონეობრივი ალაგებს ფაქტებს პრიორიტეტების მიხედვით, ასეთი ფორმალური გავრცელება ობიექტის დონის ენის არგუმენტების კონსტრუქციისთვის წესებთან და პრიორიტეტებთან ერთად [10]. არგუმენტები კომპლიქტის შესახებ პრიორიტეტები შეიძლება ჩამოაღიბდეს სხვა ფაქტებიდან გამომდინარე, ასეთი მიდგომა აზოგადებს ფორმალიზაციას და უპირატესობები შეიძლება დამყარებული იყოს

წესების პრიორიტეტებზე არამედ არგუმენტის სიძლიერეზე.

## 5.2 კლასიკური არგუმენტაცია

ამ თავში განვიხილავთ დანგის არგუმენტების თეორიას [4,15]:

1. დანგის არგუმენტების ფრეიმვორქი არის პირობა  $AF = (Args; R)$  სადა  $Args$  არის არგუმენტების ერთობლიობა და  $R \subseteq Args \times Args$ . პირველ ფიგურაში ნაჩვენებია ისრები  $x$  და  $y$  - კენ სადაც  $(X; Y) \in R$  (სურ. 37)



სურ. 37 დანგის არგუმენტაციის ფრეიმვორქის მაგალითი

დანგი შემდეგ განიხილავს არგუმენტების წვდომას, ფუნქციების თვისებებს და ფრეიმვორქის დასაშვებ გავრცელებას.

2.  $AF = (Args; R)$ ,  $S \subseteq Args$  და  $A; B; C; \dots$ : აღნიშნავს არგუმენტს  $Args$  ში.

- 1. სარისთვის უფალი კონფლიქტი  $A; B \in S$

ეს არ არის შემთხვევაროცა  $(A; B) \in R$

- $A$  ს შეუძლია პატივი სცეს  $S$  თუ  $B \in Args$ : თუ  $(B; A) \in R$  შემდეგ შესაძლებელია  $C \in (C; B)$

• ფუნქციის ტისებები  $AF$ , აღნიშნავს  $FAF$  განსაზღვრულია როგორც:  $FAF : 2Args \rightarrow 2Args$  ( $S$ ) =  $fA \setminus A$  წვდომადია w.r.t.  $S_g$

- თუ  $S$  თავისუფალი კონფლიქტი, მაშინ  $S$  არის დასაშვები პირობა  $AF$

თუ ყოველი უმენტიში  $\omega$  ვდომადია  $S$  (i.e.,  $S \mu FAF(S)$ )

3.  $AF = (Args; R)$ ,  $S$  კვეფუნქციაა  $Args$  ის, და  $F$  ფუნქციაა  $AF$ . როცა :

- $S$  არის ფუნქცი რომელიც მისაღებია ( $S = F(S)$ )
- $S$  არის სტაბილური ფუნქცია თუ  $8B = 2 S, 9A \ 2 S$  და  $(A; B) \ 2 R$
- $S$  არის ფუნდამენტური ფუნქცია თუკი  $S \mu FAF(S)$

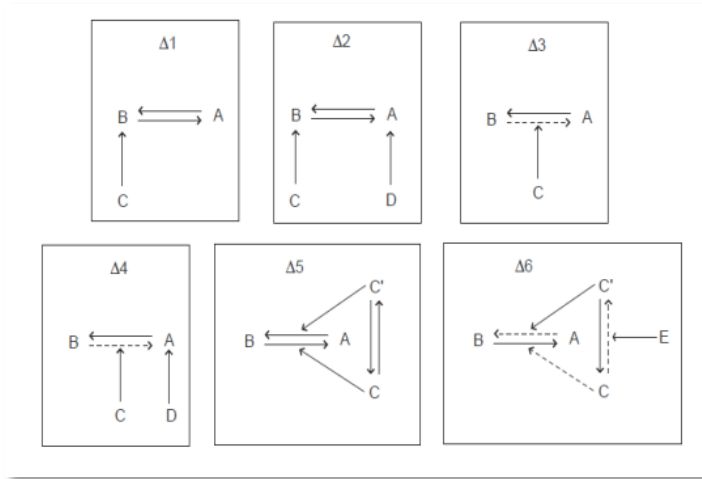
$S \in$  (სრულყოფილი, უპირატესი, სტაბილური, ფუნდამენტური)

არგუმენტი გულისხმობს სემანტიკას თუკი ის ეკუთნის ფუნქციას.

არგუმენტი არის სემანტიკური თუ ის ეკუთნის მხოლოდ ერთს და არა ყველას ერთად.

### 5.3 მნიშვნელობაზე დამოკიდებული არგუმენტაცია

ამ თავში ჩვენ გვსურს განვაგრძოთ დაგის თეორია რომ ყურადღება მივაქციოთ არგუმენტებს, გვერდი დაყოფილია შემდეგნაირად, განვიხილოთ დაგის თეორიას და არგუმენტებს, იმ გვერდს რომელსაც ჩვენ მიუყვებით აღვწერთ სწავლებას არგუმენტების ფორმალიზაციას. გავიხსენოთ პირველი სექციის ამინდის პროგნოზის მაგალითი სადაც  $A$  და  $B$  სემანტიკური შეტევებია  $((A; B), (B; A) \in R)$   $A$  ა უპირატესია  $B$  ზე(რასაც ითხოვს არგუმენტი  $C$ ), ვერ ეპაექრება  $A$  ს, პარალელურად  $A$  ვერასოდეს ამარცხებს  $B$  ს, თუ ჩვენ შევაფასებთ შესაძლებლობას მაშინ  $A$  პირობა დასაშვებია და  $B$  არა. (სურ 38) მოცემულია დენგის თეორიის ვარიანტები.



სურ. 38 დენგის თეორიის გავრცობის რამდენიმე ვარიანტი

#### 5.4 სემანტიკური შეთავსების მოდელი არგუმენტაციის მექანიზმისათვის

შეთავსები მოდელის ასაგებად [13] დაუშავთ, რომ ობიექტი, ან ცნება წარმოდგება სასრული რაოდენობის ფენების სახით. ნულოვან-საწყის ფენაში მოთავსებულია თვითონ ობიექტი (ცნება) თავისი ატრიბუტებით (თვისებებით და მიმართებებით), ხოლო ყოველ შემდეგ ფენაში თავსდება წინა ფენის ობიექტის შემადგენელი ნაწილები-ქვეობიექტები (ცნების შემადგენელი ქვეცნებები) თავიანთი ატრიბუტებით. შემოვიტანოთ შეთავსების ფუნქცია  $\varphi: g(O) \times g(T) \rightarrow R$ , რომელიც განსაზღვრავს გამოსაცნობი ობიექტებისა და ცნებების პატერნების სიახლოვეს აქ  $g: X \rightarrow \{G_1, G_2, \dots\}$  არის ოპერატორი რომელიც გამოსაცნობ ობიექტს ან კლასის პატერნს ასახავს მათი აღმწერი გრაფების სიმრავლეში. განვსაზღვროთ  $\varphi$  ფუნქციის მნიშვნელობების მიღების პირობები თვისებების და მიმალტებების შემთხვევაში:

1. თვისებების შეთავსება:  $\varphi((g^p(o), g^p(t))) \geq 0$  თუ

$$f_p(g(t)) \subseteq f_p(g(o)) \& \forall p \{p \in f_p(g(t)) \& (f_v(p_0) \subseteq f_v(p_t))\}$$

ობიექტის თვისებების სიმრავლე სრულად მოიცავს პათერნის შესაბამის სიმრავლეებს და პათერნის თოთეული თვისების მნიშვნელობებისთვის სრულდება კონკრეტული პრედიკატებით განსაზღვრული პირობა ობიექტის შესაბამის მნიშვნელობებთან მიმართებაში. მაგ: თუ პათერნისთვის განსაზღვრულია თვისება „სიგრძე

$< 10^{-6}$  და ობიექტს გააჩნია ეს თვისება მაშინ შეთავსება , განხორციელდება თუ თვისების მნიშვნელობა ნაკლები იქნება  $10^{-6}$  ზე , მაგ. 5. ამ შემთხვევაში გვაქვს სრული განსაზღვრულობა (*certainty*).

2. მიმართებების შეთავსება:  $\varphi((g^r(o), g^r(t))) \geq 0$  თუ

$$f_r(g(t)) \subseteq f_r(g(o)) \&\forall r \{if \exists p((r, p) \in g(t)) \& (f_v(p_o) \subseteq f_v(p_t))\}$$

Certainty ( $\sigma(t), \sigma(o)$ )

სადაც  $\sigma$ - ოპერატორია , რომელიც ნებისმიერ ობიექტს უკვშირებს მისი ქვეობიექტების სიმრავლეს. ობიექტების მიმართებები უთავსდებიან პათერნის მიმართებებს თუ პათერნის ქვეობიექტების სიმრავლესრული განსაზღვრულობით უთავსდება ობიექტის ქვეობიექტების სიმრავლეს, პათერნის მიმართებების სიმრავლეს სრულად ფარავს ობიექტების მიმართებების სიმრავლე და პათერნის ნებისმიერი მიმართების თვისების მნიშვნელობის დაკმაყოფილება ხდება თუ მას თვისება გააჩნია.

3.  $\varphi((g(o), g(t))) \leq 0$  სხვა შემთხვევაში , როცა პათერნი ნაწილობრივ უთავსდება ობიექტს ან საერთოდ არ უთავსდება და ამ შემთხვევაში ვამბობთ რომ გვაქვს განუზღვრელობა (*uncertainty*).

#### 5.4 თვისობრივი არგუმენტების გენერირება

თვისობრივი არგუმენტაციის გენერაცია[4] იწყება მაშინ , როცა ობიექტის თვისებების შეთავსების აგენტება თავიანთი საქმე დაამთავრეს. არგუმენტაციის დასაგენერირებლად გამოიყენება

$\varphi((g^i(o), g^p(t)))$  ფუნქცია და დარდება მათი თვისებების სიმრავლეები.

ქვევით ცხრილ 1 -ში მოყვანილია c და h-ის მნიშვნელობები . თუ მოხდა პათერნთან სრული თანხვედრა მაშინ , ყველაფერი გარკვეულია და მის კონფიდენციალობის დონე არის ჩერტაინტყ და სხვა აგენტები ვერ შეეკამათებიან ამ არგუმენტს. მაშინ როცა ხდება ნაწილობრივი შეტავსება და შეთავსების ფუნქცია დადებითია მაშინ ცონფიდენციალობა გაურკვეველია და შესაძლებელია ამ

არგუმენტაციის გაპროტესტება.

---

<b>C</b>	<b>+(h)</b>
----------	-------------

---



<b>შეთავსებად ია</b>	თუ ობიექტის ყველა თვისება $\omega$ თანხვედრაშია პათერნ $\tau$ -ს თვისებებთან და შესაბამისად, მნიშვნელობებიც ემთხვევა და $\varphi(g^p(\omega), g^p(t)) \geq 0$
<b>შეუთავსება დია</b>	თუ ობიექტის ყველა თვისება $\omega$ მსგავსია პათერნ $\tau$ -ს თვისებებთან, ან / და შესაბამისად, მნიშვნელობებიც არ ემთხვევა და $(g^p(\omega), g^p(t)) \geq 0$
<b>C</b>	<b>-(h)</b>
<b>შეუთავსებე ლია</b>	სხვა შემთხვევაში $\varphi(g^p(\omega), g^p(t)) < 0$

ცხრილი 1 თვისობრივი არგუმენტების შეთავსებადობა

### 5.5 მიმართებითი არგუმენტების გენერირება

მიმართებითი არგუმენტაციის პროცესი იწყება მაშინ თუ თვისობრივი არგუმენტ გენერირებულია და ის განიცდის შეტევას, ამასთან ყველგამოსაცნობი ობიექტის ყველა ქვეობიექტი. უნდა იყვეს პატერნებთან შეთავსებული. ქვევით ცხრილ-ში მოყვანილია არგუმენტების კონფიდენციალობის და კონტრ არგუმენტების წამოყენების პირობები. აქ არგუმენტაციის გამოსათვლელად გამოიყენება  $\varphi(g^r(\omega), g^r(t))$  ფუნქცია. (ცხრილი 2)

<b>C</b>	<b>+(h)</b>
<b>შეთავსებად ია</b>	თუ ობიექტის ყველა თვისება $\omega$ თანხვედრაშია პათერნ $\tau$ -ს თვისებებთან და შესაბამისად, მნიშვნელობებიც ემთხვევა და $\varphi(g^r(\omega), g^r(t)) \geq 0$
<b>შეუთავსება დია</b>	თუ ობიექტის ყველა თვისება $\omega$ მსგავსია პათერნ $\tau$ -ს თვისებებთან, ან / და შესაბამისად, მნიშვნელობებიც არ ემთხვევა და $(g^r(\omega), g^r(t)) \geq 0$
<b>C</b>	<b>-(h)</b>
<b>შეუთავსებე ლია</b>	სხვა შემთხვევაში $\varphi(g^r(\omega), g^r(t)) < 0$

ცხრილი 2 მიმართებითი არგუმენტების შეთავსებადობა

### 5.6 აგენტების მიერ სტრუქტურული არგუმენტების გენერაცია

სტრუქტურული აგენტები არგუმენტების გენერაციას განიხილავენ ცნებების ონტოლოგიური იერარქიის კონტექსტში, როცა მოცემული ობიექტი შეიძლება შეუთავსდეს იერარქიის ნებისმიერ ცნებას. რა თქმა უნდა არგუმენტების გენერირება იწყება ზევიდან ქვევით და ალგორითმი უნდა ჩავიდეს რაც შეიძლება ღრმად. თუ რომელიმე დონეზე ვერ მოხერხდა შეთავსება თვისობრივად და მიმართებრივად და არის გაურკვეველობა, მაშინ იერარქიის დაბალ საფეხურებზე მდგომი ცნებები უნდა მოინიშნოს როგორც არა გამოცნობადი, უდავო კონფიდენციალობით და მათთვის არგუმენტები აღარ უნდა დაგენერირდეს. თუ იერარქიის რომელიმე დონეზე გვაქვს განუზრვრელობის კონფიდენციალობა და კონტრ არგუმენტი არსებობს ამ დონეზე, მაშინ ეს ცნებაც უნდა მოინიშნოს როგორც არა გამოცნობადი, უდავო კონფიდენციალობით და არგუმენტაცია უნდა გაგრძელდეს მშობელი ცნებისთვის. (ცხრილი 3)

აგენტის დანიშნულება	კ	ტი	რაოდენობა
	ოდი	პი	
ობიექტების გამოყოფა	A	Sla	m
	O	ve	
⋮ თვისებების მნიშვნელობების გამოთვლა	A	Sla	m+1
	P	ve	
⋮ ობიექტის თვისებების შეთავსება პათერნის თვისებებთან	A	Sla	m+n
	PM	ve	
⋮ ობიექტის მიმართების შეთავსება პათერნის მიმართებებთან	A	Sla	m+n
	RM	ve	
⋮ ობიექტის სტრუქტურის შეთავსება პათერნის	A	Sla	m+n
	SM	ve	
⋮ არგუმენტების გენერატორები	A	Ma	m+n
	A	ster	
⋮ გადაწყვეტილების მიმღები	A	Gu	1
	DM	ru	

### ცხრილი 3 დაგენერირებული სტრუქტურული არგუმენტების ცხრილი

პირველ რიგში მუშაობას იწყებენ თვისებების გამომთვლელი და ობიექტების გამომყოფი აგენტები. თვისებების გამომთვლელი აგენტები მუშაობას იწყებენ იმ შემთხვევაში, როგორც კი რაიმე ობიექტი არსებობს ან გამოიყოფა. თავის მხრივ, თვისების შეთავსების აგენტები მაშინ აქტიურდებიან, როდესაც თვისებების მნიშვნელობები გამოთვლილია. ამის შემდეგ აქტიურდებიან თვისებების მიხედვით არგუმენტაციის გენერატორი აგენტები. თუ მოცემული კლასისთვის თვისებები გამოთვლილია და არსებოს კონტრ არგუმენტი თვისებების მიხედვით ამ კლასის შესახებ, მაშინ აქტიურდებიან მიმართებათა შეთავსების აგენტები, რომელთა შემდეგაც იწყებენ მუშაობას მიმართებების მიხედვით არგუმენტაციის გენერატორი აგენტები. თუ მოცემული კლასისთვის კონტრ არგუმენტები მაინც ჩნდება მაშინ იწყებენ მუშაობას შესაბამისად სტრუქტურის შემთავსებელი და არგუმენტების გენერატორი აგენტები.

მას შემდეგ რაც თითოეული არგუმენტების გენერატორი აგენტი დააგენერირებს თავისი არგუმენტების სიმრავლეს, აგენტები ცვლიან თვიანთ არგუმენტებს. როცა ყველა აგენტისთვის ცნობილი გახდება სხვა აგენტების არგუმენტები მაშინ ისინი აგენერირებენ შეტევათა სიმრავლეს. შეტევა ჩნდება მაშინ, როცა ობიექტის შეთავსების მნიშვნელობა პატერნთან უკეთესია ვიდრე სხვისა და  $h$  - ის მნიშვნელობა კონფლიქტურია, მაგალითად არგუმენტ  $(m_1(o, t_1), P, certainty, +)$  -ზე ხორციელდება შეტევა არგუმენტ  $(m_2(o, t_2), P, certainty, -)$  - ის მიერ თუ  $m_1$  და  $m_2$  წარმოადგენენ ერთიდაიგივე ობიექტის სხვადასხვა პატერნთან შეთავსებას, ან  $(m_2(o, t_1), R, certainty, -)$  - ს შემთხვევაში თუ გვაქვს ობიექტის ერთიდაიგივე პატერნთან შეთავსება.

როდესაც არგუმენტების და შეთავსების სიმრავლეები წარმოქმნილია, აგენტებმა უნდა გამოითვალონ უპირატესი გაფართოება, აუდენციის და კონფიდენციის მნიშვნელობიდან გამომდინარე. არგუმენტების სიმრავლე გლობალურად სუბიექტურად მისაღებია, თუ თითოეული ელემენტი ჩანს რომელიმე აგენტის სასურველ გაფართოების

სიმრავლეში. არგუმენტების სიმრავლე გლობალურად ობიექტურად მისაღებია, თუ თითოეული ელემენტი ჩანს ყველა აგენტის სასურველ გაფართოების სიმრავლეში. თუ არგუმენტი არის არც სუბიექტურად და არც ობიექტურად მისაღები მაშინ ის განიხილება როგორც უსაფუძვლო.

#### **თავი 6. კოლექტიური გადაწყვეტილებების მიღება ვიდეოთამაშებში.**

მრავლაგენტიანი ხელოვნური ინტელექტის ერთერთი ძირითადი მექანიკა მდგომარეობს კოლექტიურ გადაწყვეტილებებში. აგენტები ურთიერთქმედებენ ერთმანეთთან და ახდენენ საკითხის საერთო შემუშავებას. ასეთი მიდგომის გამოყენება ყოველთვის არ არის მიზანშეწონილი, რატომაუნდა მოცანის ტიპზეა დამოკიდებული. ვიდეოთამაშებში უანრი და სცენარი მკაცრად განსაზღვრავს ტექნიკურ მახასიათებლებს. მაგალითად, ჭიანჭველების ამოცანა იდეალური ვარიანტია, რომ განვახორციელოთ კოლექტიური გადაწყვეტილებების სისტემა.

#### **6.1 მულტიაგენტური სისტემის ზოგადი დახასიათება**

ზოგ გაერთიანებას (დომინ) ჭირდება MAS(MultyAgent System). უფრო დეტალურად, თუამ გაერთიანებაში არიან სხვადასხვა ორგანიზაციები, სხვადასხვა ტიპისა დამიზნების მქონე ხალხი, საჭიროა მრავლაგენტიანი სისტემა მათიურთიერთობისათვის. იმ შემთხვევაშიც კი, თუ თითოეულ ორგანიზაციას სურს საკუთარი შინაგანი უშიშროების მოდელირება ერთი საერთო სისტემით, ორგანიზაციები არ მისცემენ ამ სისტემის შექმნის უფლებას მხოლოდ ერთ ადამიანს.

სხვადასხვა ორგანიზაციებს ჭირდებათ საკუთარი სისტემა, რომელიც ყველაზე უკეთ აკმაყოფილებს მათ მოთხოვნებს.

მაგალითისათვის წარმოვიდგინოთ კომპანია X, რომელიც აწარმოებს საბურავებს, მაგრამ ხელშეკრულებით თანამშრომლობს თხილის პროდუქციაზე კომპანია თან. ამ პროდუქციის პროცესის ერთი სისტემაში მოსაყვანად, უნდა მოხდეს X-სა და -ის შიდა საქმეების მოდელირება. თუმცა არცერთ კომპანიას არ აწყობს საკუთარი ინფორმაციის ან კონტროლის სხვა კომპანიისათვის გასაჯაროება ან გადაცემა. ორი კომპანიის შემთხვევაში შესაძლოა მიიღწიოს შეთანხმებას, მაგრამ მრავალი კომპანიის შემთხვევაში MAS მნიშვნელოვანია.

სხვა მაგალითი შეიძლება იყოს საავადმყოფოს განრიგის შედგენა. ეს გარემო მოითხოვს სხვადასხვა აგენტებს რომ წარმოადგინოს სხვადასხვადამიანების ინტერესები საავადმყოფოში (სხვადასხვა გამოცდილებისა დაპროფესიის ექიმები, რომლებსაც გააჩნიათ განსხვავებული განრიგი, ექთნები, ასევე ოპერატორები და ა.შ). რადგანაც ამ ადამიანებს გააჩნიათ სხვადასხვა განრიგი, სხვადასხვა კრიტერიუმების გათვალისწინებით, მათი წარმოდგენა საჭიროა სხვადასხვა აგენტებით.

იმ გარემოებებშიც, სადაც არ გამოიყენება დისტრიბუციულისისტემები, ზოგჯერ საჭიროა MAS. მრავალი აგენტის არსებობამ შესაძლებელია ოპტიმიზაცია გაუკეთოს სისტემის წარმადობას, პარალელური გამოთვლის მეთოდის წარმოებით. MAS ის პარალელიზმი შესაძლოა დაეხმაროს დროით შეზღუდული ან სივრცით შეზღუდული მოთხოვნების დაკმაყოფილებას.

გარდა იმისა, რომ დავალებების სხვადასხვა აგენტებზე განაწილებით მიიღწევა პარალელიზმი, სიმძლავრე არის MAS ისუპირატესობა, რომელსაც ყავს ზედმეტი აგენტები. თუ კონტროლი და შესაძლებლობები თანაბრადაა გადანაწილებული სხვადასხვა აგენტებზე, სისტემის მუშაობა ბევრად სტაბილური და ეფექტურია. თუ ერთი აგენტი (პროცესორი) მართავს ყველაფერს, ერთიუმნიშვნელო შეცდომა ხშირ შემთხვევაში განაპირობებს სისტემის ავარიულად

დასრულებას. თუმცა არ არის აუცილებელი მრავალაგენტიანი სისტემა განაწილებული იყოს სხვადასხვა პროცესორებზე, სისტემის ავარიული მარცხის თავის ასარიდებლად აგენტები უნდა იყვნენ გადანაწილებული სხვადასხვა მანქანებზე.

მრავალაგენტიანი სისტემების კიდევ ერთი უპირატესობაა მასშტაბურობა (სკალაბილიტე). ისინი არსებითად მოდულურები არიან და კარგ მოდელირებულ სისტემაში ადვილია ახალი აგენტების დამატება რათა სისტემა გაფართოვდეს ახალი შესაძლებლობებით. ასევე ეს უპირატესობამოიცავს არსებული მოდულების შესაძლებლობებისა და კონტროლის ცვლილების სიმარტივესა და მოქნილობას. პროგრამისტის თვალსაზრისით, მოდულურობას მიყვავართ მარტივ პროგრამირებასთან. ერთი რთული ამოცანის ცენტრალიზებულ აგენტზე დავალების ნაცვლად, მას შეუძლია განსაზღვროს ქვე-ამოცანები და დაუწესოს კონტროლი სხვადასხვა აგენტებს ამ დავალებების შესასრულებლად.

მრავალაგენტიანი სისტემა, ინტელექტთან ერთად ასევე შეიძლება გამოყენებულ იქნას სოციალური და ცხოვრებისეული მეცნიერების ფუნდამენტური პრობლემის გადასაჭრელად. ინტელექტი ღრმად დაკავშირებულია ურთიერთქმედებასთან. ითვლება რომ ყველაზე საუკეთესო გზის დასაწყისი ინტელექტუალური მანქანების შესაქმნელად არის სოციალური მანქანების შექმნა. ეს თეორია ეფუძნება სოციალურ- ბიოლოგიურ თეორიას რომელიც მიიჩნევს რომ ხელოვნური ინტელექტი სწორედ სოციალური ურთიერთობის საფუძველზე შეიქმნა.

გარდა იმისა, რომ MAS თავსებადია ზოგადად ყველა ყველა ზემოთ ხსენებულ პრინციპთან, არსებობს რამდენიმე არგუმენტი მულტი-რობოტულ სისტემებში. დავალებებში, რომლებიც ითხოვენ რობოტების განლაგებას კონკრეტულ ადგილებში, უპირატეობა აქვთ რობოტების ჯგუფს ერთ რობოტთან შედარებით, რადგან შესაძლებელია მათი გადანაწილება სხვადასხვა გეოგრაფიულ ადგილებში. საბოლოო ჯამში, როგორც ვიმსჯელებთ, მრავალ-რობოტიანი სისტემა, რომელიც შედგება შედარებით იაფფასიანი, შეზღუდული შესაძლებლობების, მაგრამ კონკრეტული დავალების შესრულების უნარის მქონე რობოტებისაგან,

მეტი უპირატესობით სარგებლობს, ვიდრე ერთ-რობოტიანი სისტემა, მძლავრი და ყოვლისშემძლე რობოტით.

ქვემოთ ჩამოთვლილია მიზეზები, თუ რატომ უნდა გამოვიყენოთ

MAS:

მრავალაგენტური სისტემების გამოყენების მიზეზები

- რაღაც გარემოებები ითხოვენ მას
- პარალელიზმი
- სიმძლავრე
- მასშტაბურობა
- მარტივი პროგრამირება
- ინტელექტუალურობის შესასწავლად
- გეოგრაფიული გადანაწილება
- ფასის ეფექტურობა

## 6.2 მულტიაგენტური სისტემის საჭიროება

### ჭიანჭველების ამოცანაში.

ჭიანჭველების ამოცანაში აგენტთა ქცევები დაფუძნებულია მათ როლებზე. თვითოეული როლი განსაზღვრავს ჭიანჭველის ცოდნის ბაზის სპეციალიზაციას. მათი ცოდნის გადაცემა ერთმანეთის მიმართ მჭიდროდ არის დაკავშირებული ამოცანის ამოხსნის სისწრაფესთან. როგრც ვნახეთ, ამის მთავარი განმპირობებელია ცოდნის პირდაპირი გზით გადაცემა აგენტებისათვის.

რა მოხდება თუ ჭიანჭველები იმოქმედებენ გუნდურად? რას ნიშნავს ჭიანჭველების გუნდად გაერთიანება? რა განაპირობებს რამდენიმე ჭიანჭველის კოლექტივში შესვლას? გაცილებით უკეთეს შედეგს მივიღებთ, თუ მოვახდენთ ჭიანჭველათა გუნდურად ფორმირებას.

ჭიანჭველების ამოცანაში გუნდების ფორმირება ხდება დინამიურად. ნებისმიერ დროს, რეალურ დროში, რომელიმე კლასის ჭიანჭველამ შეიძლება დაიმატოს სხვა, იგივე კლასის ჭიანჭველა და მათი მოქმედება მოხდეს კოლექტიურად. ასევე, შეიძლება ვინმე გათავისუფლდეს გუნდიდან. თვითოეული კლასის ჭიანჭველათა გაერთიანებისთვის და გაცალკევებისათვის არსებობს გარკვეული

პირობები, ასევე ამ პროექტის დადებითი და უარყოფითი მხარეები. გუნდში გაერთიანება უნდა მოხდეს ოპტიმალური პარამეტრის მხრივ, ოპტიმალური პარამეტრი ბევრი რამ შეიძლება იყოს, მაგალითად, მზვერავისთვის სწრაფად სიარული, ან თვალთახედვის რადიუსი, გადამზიდისათვის ჯამური ფოთლების აწევის ცონა, ან მოძრაობის სისწრაფე. განვიხილოთ აგენტთა ჯგუფურად გაერთიანების მაგალითები და კოლექტიური გადაწყვეტილებები.

### **6.3 კოლექტიური გადაწყვეტილების მაკონტროლებელი მექანიზმი- შავი დაფის არქიტექტურა.**

ჯგუფური გადაწყვეტილებების მექანიზმი დაფუძნებულია ჯგუფურ არგუმენტებზე. არსებობს არგუმენტები, რომელთაგან ხდება "უფრო სწორის" არჩევა და მიიღება კოლექტიური გადაწყვეტილება. ბუნებრივია, ჩნდება კითხვა, რას ნიშნავს სწორი არგუმენტი? ვინ წყვეტავს რომელი არგუმენტი არის გამომავალი მონაცემი?

შავი დაფის არქიტექტურა არის რთული ამოცანების გადაჭრის სტრატეგია სხვადასხვა ცოდნის წყაროების გამოყენებით[5,16,17]. შავი დაფა არის გლობალური წვდომადი მონაცემთა ბაზა რომელიც გამოიყენება შუალედურად, ამოცანის გადაჭრის ნაწილობრივ შედეგებზე. შავი დაფის არქიტექტურა შეგვიძლია წარმოვიდგინოთ როგორც ჯგუფი სპეციალისტების რომლებიც სხედან შავი დაფის წინ. ისინი მუშაობენ ერთობლივად იმისთვის რათა მოახდინონ ამოცანის გადაჭრა და ამისთვის ისინი იყენებენ შავ დაფას როგორც სამუშაო ადგილს ამოცანის გადაწყვეტისთვის.

პრობლემის გადაჭრა იწყება მისივე გამოცხადებით და ხდება თავდაპირველი მონაცემების დაწერა შავ დაფაზე. სპეციალისტები უყურებენ შავ დაფას და ეძებენ შესაძლებლობას რომ წვლილი შეიტანონ ამოცანის განვითარებაში. როდესაც სპეციალისტი იპოვის შესაძლებლობას ის აკეთებს ჩანაწერს შავ დაფაზე და იმედოვნებს რომ ისინი გამოიყენებენ მის წვლილს რათა იპოვონ ამოცანის გადაჭრის გზა. ეს პროცესი გრძელდება მანამ სანამ არ მოხდება ამოცანის გადაწყვეტა. შავი დაფის არქიტექტურა შედგება სამი კომპონენტისგან.

შავი დაფის არქიტექტურა შედგება 3 კომპონენტისგან:



- ცოდნის წყაროები
- შავი დაფა
- მაკონტროლებელი კომპონენტი

ცოდნის წყაროები არიან დამოუკიდებელი მოდულები რომლებიც შეიცავენ საჭირო ცოდნას ამოცანის გადაწყვეტისთვის. მათ არ სჭირდებათ იცოდნენ არსებობა სხვა ცოდნის წყაროების, მაგრამ მათ უნდა იცოდნენ თუ რა მდგომარეობაშია ამოცანის გადაჭრის პროცესი და ინფორმაცია რომელიც წარმოდგენილია შავ დაფაზე.

#### **6.4 შავი დაფის არქიტექტურა სტრატეგიულ ვიდეოთამაშებში**

მთავარი მიზანი რეალურ დროში სტრატეგიული ვიდეოთამაშისა არის დამოკიდებულება ჯგუფებს შორის და არა ინდივიდუალური დამოკიდებულება, ამიტომაც საჭიროა ცენტრალიზებული სისტემის კონტროლი. ეს ამცირებს ამოცანის მთლიან სირთულეს რადგან აძლევს ინდივიდუალებს იყენენ მარტივები, ხოლო სირთულე ექცევა ცენტრალიზებული სისტემის ფარგლებში.

აგენტის სხვა ასპექტით აგენტების კორდინაცია არის მათიკომუნიკაცია. შავი დაფის სისტემა არის მიდგომა რომელიც საშუალებას იძლევა მოხდეს აგენტების კომუნიკაცია ცენტრალიზებულად[18]. ზუსტად შავი დაფის არქიტექტურაც წარმოადგენს ისეთ არქიტექტურას, სადაც მუშაობა ხდება ცენტრალიზებულად და კომუნიკაცია სტრუქტურული მესიჯებით[17].

აგენტი აგენერირებს ინფორმაციას, რომელიც მოიცავ ცოდნას მისი შიდა მდგომარეობის და შემოგარენის შესახებ, ეს მესიჯი კი ისახება შავ დაფაზე. იგი წარმოადგენს სისტემის საწყის მდგომარეობას. ეს ყველაფერი საშუალებას აძლევს სისტემას იყოს მარტივიმოქნილი, სირთულეების გარეშე.

შავი დაფის არქიტექტურა შექმნილია იმისთვის რათა გაუმკლავდეს რთულ ამოცანებს, ცუდ სტრუქტურებს. ეს ყველაფერი კი დაფუძნებულია საძიებო პროგრამულ ცოდნაზე და საშუალება აქვს გააერთიანოს სხვადასხვა ცოდნის წყაროები. მთავარი პრობლემაა შავი დაფის არქიტექტურაში მოხდეს სტრატეგიის შენარჩუნება და რეალურ დროში უსაფრთხოების უზრუნველყოფა.

ჭიანჭველების ამოცანაში შეგვიძლია შევთანხმდეთ, რომ ნებისმიერ ჭიანჭველას შეუძლია გასწიოს ლიდერობა ჯგუფში. შესაბამისად მასვე შეუძლია ჩამოაყალიბოს გუნდი. თვითონ კი იქნება ლიდერი. ამ შემთხვევაში გუნდის შავი დაფის მაკონტროლებელი მექანიზმიც თავადვე შეიძლება იყოს. გუნდის ფორმირება შეიძლება ბევრნაირად შეიძლება მოხდეს(პარამეტრზე დამოკიდებული, მარგი ქმედების კოეფიციენტი და ა.შ.)

#### **6.4 მზვერავთა გუნდის დინამიური ფორმირება დაკოლექტიური გადაწყვეტილებები.**

ჭიანჭველების ამოცანაში ნებისმიერ მზვერავს აქვს ფოთლის ამოცნობის  $R$  რადიუსი. მოცემული რადიუსის მქონე წრეწირის შიგნით იგი აღმოაჩენს ვარგის ფოთოლს და მონიშნავს მას. ყოველი მზვერავი მოძრაობს გარკვეული  $V$  სიჩქარით. დავუშვათ, ამოცანაში შემდეგი პირობაა - ნებისმიერი მზვერავი ჭიანჭველისთვის რადიუსის დამოკიდებულება უკუპროპორციულია სიჩქარისა. ჭიანჭველა რაც უფრო ჩქარა მოძრაობს, მით ნაკლებია მისი მოქმედების რადიუსი, ხოლო რაც უფრო ნელა მოძრაობს, პირიქით, მას შეუძლია უფრო ფართო ტერიტორია გამოიკვლიოს. მზვერავის მარგი ქმედების კოეფიციენტი გამოვთვალოთ შემდეგი ფორმულით:

$$Z = RV$$

მზვერავების გუნდურად გაერთიანება მნიშვნელოვანია ამოცანის ამოსხნის წარმადობისთვის. თითოეული მზვერავის გუნდში აყვანის შემდეგ გუნდი იწყებს ახალი სიჩქარით მოძრაობას. ყველა ჭიანჭველას საშუალოსიჩქარეა

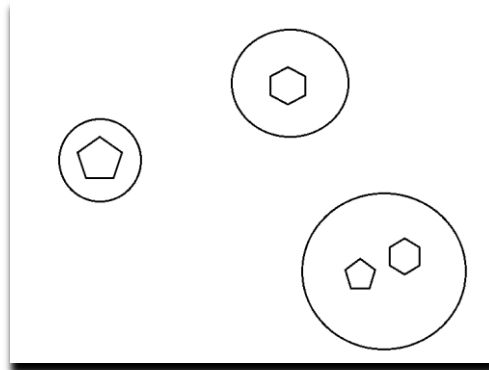
$$V_{max} = \sum_{i=1}^n (V_i / n)$$

ხოლო ჯამური დასახვერი ფართობის რადიუსი, თითოეული ჭიანჭველის მიერ გამოკვლეული ფართობის რადიუსების ჯამის საშუალო არითმეტიკულია:

$$R_{max} = \sum_{i=1}^n (R_i / n)$$

ერთის მხრივ, მზვერავებმა უნდა გაითვალისწინოს ამოცანის შესრულების სისწრაფე და გუნდში ჰყავდეს სწრაფი გადაადგილების მქონე ჭიანჭველები. მეორეს მხრივ, ფართე რადიუსი, რადგან უფრო

მეტი ხილვადობა ჰქონდეს ჯგუფს. ამ შემთხვევაში, საჭიროა ოპტიმალური შუალედის პოვნა, ჭიანჭველები არც ძალიან უნდა აჩქარდეს და არც უნდა შენელებს. შესაბამისად, მათ უნდა ჰქონდეს მაქსიმალურად დიდი ხილვადობის რადიუსი და უნდა მოძრაობდეს მაქსიმალურად სწრაფად (სურ 39).



სურ. 39 მზვერავთა გუნდური ფორმირება და საერთო რადიუსი

თვითოეულ აგენტთა ჯგუფზე შეგვიძლია განვიხილოთ კლასიკური შავი დაფის არქიტექტურა. აგენტთა არგუმენტები იწერება შავ დაფაზე და მაკონტროლირებელი მექანიზმი წყვეტს საბოლოო გამომავალ გადაწყვეტილებას.

ყოველი ახალი ჭიანჭველას შეხვედრის შემთხვევაში, ირთვება გუნდის არგუმენტაციის მექანიზმი, თვითოეული ჭიანჭველა ამბობს სათავისო არგუმენტს მისი მარგი ქმედების კოეფიციენტიდან გამომდინარე, ანუ რომელი პარამეტრით არის უპირატესი. შავ დაფაზე დაწერილი არგუმენტების მიხედვით, მაკონტროლებელი მექანიზმი წყვეტს გუნდიდან წამოსული არგუმენტების მიხედვით აიყვანოს თუ არა ახალი წევრი, თუ ჩაანაცვლოს არსებული. ახალი წევრის აყვანამ დიდი რადიუსით, შეიძლება ძალიან შეანელოს მოქმედება, ხოლო სწრაფად მოძრაობა ჭიანჭველამ კი ხილვადობის რადიუსი შეამციროს.

აღწერით გაძლიერებითი დასწავლის ალგორითმი [12], სადაც თვითოეულ ჯერზე გადაწყვეტილების მიღება დაფუძნებულია ჯილდოზე. ჯილდო პირობითად, არის მიზნის მიღწევის შედეგად მიღებული სარგებელი. ჩვენს კონკრეტულ შემთხვევაშიც გურუები, ზუსტად ასეთი მეთოდით მოქმედებს.

თუ ჩვენ გვაქვს არსებული მდგომარეობა  $x$  და ხელმისაწვდომი მოქმედება  $a$ , მაშინ  $Q$  დასწავლითი ალგორითმი მოიცემა ბოლცმანის ტოლობით[19]:

$$p(a_i|x) = \frac{e^{Q(x,a_i)/T}}{\sum_{k \in \text{actions}} e^{Q(x,a_k)/T}}$$

შედაც  $T$  არის ტემპერატურის პარამეტრი და ის განსაზღვრავს გადაწყვეტილებათა მიღების შემთხვევითობას. გურუ იღებს გადაწყვეტილებას, რის შედეგადაც იღებს  $r$ - ჯილდოს. შემდეგ კი გადადის ახალ  $y$  მდგომარეობაზე. ყოველ ჯერზე აგენტი ანახლებს  $Q(x, a)$  - რეკურსიულად, უმატებს პოზიტიურობის წონას  $\beta$  და აკლებს სამომავლო რესურსებს:

$$Q(x, a) \leftarrow Q(x, a) + \beta(r + \gamma V(y) - Q(x, a))$$

$\gamma (0 \leq \gamma \leq 1)$  არის კლებადობის პარამეტრი და  $V(x)$  მიიღება ასე:

$$V(x) = \max_{b \in \text{actions}} Q(x, b)$$

გაძლიერებითი დასწავლა თვითოეული აგენტისთვის ამომავალი არგუმენტია, რათა უპირატესი იყოს მისი აზრი შავ დაფაზე. როგორც ვთქვით, შავი დაფის მაკონტროლირებელი მექანიზმისთვის ყველაზე მთავარი ფუნქციაა ოპტიმალური არგუმენტის პოვნა. კონკრეტულ მაგალითში საუკეთესო მეთოდი იქნებოდა ხარბი ალგორითმების გამოყენება.

ხარბი ალგორითმების[7] (greedy algorithms ) გამოყენებისას, ყოველ ბიჯზე კეთდება ლოკალურად ოპტიმალური არჩევანი იმ იმედით, რომ საბოლოო შედეგიც ოპტიმალური იქნება. ეს მიდგომა ყოველთვის არ ამართლებს, მაგრამ ზოგიერთი ამოცანისათვის მართლაც ძალიან ეფექტურია. რაც მთავარია, არსებობს სტანდარტული პროცედურები იმის გასარკვევად, კორექტულად იმუშავებს თუ არა მოცემული ამოცანისთვის ხარბი ალგორითმი.

ჭიანჭველების ამოცანაში ხარბი ალგორითმის შემავალი პარამეტრი ორნაირი შეიძლება იყოს, ეს პარამეტრები განსაზღვრავს გამომავალ ოპტიმალურ ვარიანტს. მაგალითად, მზვერავის შემთხვევაში შეიძლება იყოს რადიუსი, ან სიჩქარე. ჩვენ წინასწარ უნდა განვსაზღვროთ უპირატესი პარამეტრი, გვინდა, რომ ჭიანჭველამ სწრაფად იაროს,

თუმცა მინიმალური რადიუსი ჰქონდეს? თუ პირიქით, იაროს ნელა და დიდი ხედვის რადიუსით.

ყოველი შემომავალი მონაცემი უნდა შედარდეს არსებულებს და იქიდან აირჩევა მაქსიმალური, განვიხილოთ ფუნქცია *GREEDY\_ACTIVITY\_SELECTOR(s,f)* ყოველი შემომავალი ორი მასივისთვის[20]:

```
n = length[s];
A = {1};
j = 1;
for(i = 2;i<=n;i++)
    if(si>=fi)
        A+={i}
        j = 1;
return A;
```

მზვერავთა შემთხვევაში, შავი დაფისთვის ასარჩევია გადაწყვეტილება, როდესაც გუნდში ხდება ახალი მზვერავის აყვანა, აიყვანოს თუ არა ჭიანჭველა გარკვეული რადიუსით, თუ გარკვეული სისწრაფით. ამ ფსევდოკოდის მიხედვით მოცემული კონკრეტული პარამეტრი, სიჩქარე, ან რადიუსი იქნება ოპტიმალური არჩევანი.

### 6.5 ფოთლის მჭრელთა კოლოპერაცია და ჯგუფური გადაწყვეტილებები

მზვერავთა ჯგუფური გადაწყვეტილებების მსგავსად შეგვიძლია განვიხილოთ ფოთლის მჭრელთა დინამიკური დაჯგუფებებიც. მათი კოლექტივიზაცია და მოკიდებულია მარგი ქმედების კოეფიციენტზე, რომელიც გამომდინარეობს მათსავე დამახასიათებელი ქცევიდან - შესაბამისი სიმბოლოს ფოთლის დანაწევრებისგან.

ჭიანჭველათა ჯგუფი, რომელთა შემადგენლობაში შედის *a, v, g, n...* ფოთლის ამომცნობი ჭიანჭველები. თითოეული ჭიანჭველათა ჯგუფისათვის არსებობს ჯგუფური მარგი ქმედების კოეფიციენტთა ჰემ-ცხრილი(ცხრ.4), სადაც წერია შესაბამისი

სიმბოლოს ამომცნობი ჭიანჭველის არაწარმადობის ან წარმადობის კოეფიციენტი. თავიდან ეს კოეფიციენტები არის 0-ის ტოლი.

ყოველი ახალი სიმბოლოს ამომცნობი ჭიანჭველის დამატების შემთხვევაში ამ ჰემ-ცხრილს შეემატება შესაბამისი სიმბოლოს ერთი ჩანაწერი, მნიშვნელობით 0. განვიხილოთ შემთხვევები თუ როგორ იცვლება ჰემ-ცხრილი სიმულაციის ყოველ შემდეგ ნაბიჯზე.

$a$	0
$v$	0
$g$	0
$n$	0
...	0

ცხრილი 4 ფოთლის მჭრელთა საწყისი პარამეტრების ცხრილი

ნებისმიერი ფოთლის დამუშავების შემდეგ, ჯგუფში გადაიხედება შესრულებული სამუშაო და ამ სამუშაოზე მომუშავე ჭიანჭველები გადაირჩევა. ვთქვათ, თუ ჭიანჭველა  $a$ -სთვის შესაბამისი სიმბოლო ვერ მოიძებნა და ის, ასე ვთქვათ, უსაქმოდ დარჩა, ჰემ-ცხრილის გრაფაში მისი სიმბოლოს შესაბამის მნიშვნელობა შემცირდება 1-ით. ესე იგი, არსებული ჭიანჭველის წარმადობა შემცირდა. თუ დარჩა ისეთი ფოთლის ნაწილი, რომლის სიმბოლოს ამომცნობი არ მოიძებნება გუნდში, მაშინ გუნდის ჰემ-ცხრილში ჯერ დაიძებნება შესაბამისი სიმბოლოს ჩანაწერი. პოვნის შემთხვევაში მისი მნიშვნელობა გაიზრდება 1-ით, ხოლო არ-პოვნის შემთხვევაში, გაკეთდება ახალი ჩანაწერი ამ სიმბოლოთი და მნიშვნელობით 0. ეს იმას ნიშნავს, რომ შესაბამისი სიმბოლოს ამომცნობი ჭიანჭველის საჭიროების კოეფიციენტი გაიზარდა, ან საჭირო ჭიანჭველის გრაფა დაემატა. (ცხრილი 5)

$a$	1
$v$	0
$g$	2

$n$	-3
...	0
$t$	0

ცხრილი 5 ფოთლის მჭრელის ჰემ-ცხრილის გარკვეული ცვლილებები გარკვეულ ბიჯზე

პირობითად, არსებობს გარკვეული ზღვარი ჰემ-ცხრილის მნიშვნელობებში, როდესაც საჭირო ხდება გურუ-ჭიანჭველის მიერ ახალი წევრის დამატება ან ძველი წევრის გაშვება. თუ მარგი ქმედების კოეფიციენტი მნიშვნელობით გარკვეულ ზღვარს ( $-\delta$ ) ჩამოსცდება (ცხრ.2), გურუ იღებს გადაწყვეტილებას, რომ შესაბამისი სიმბოლოს ამომცნობი ჭიანჭველა საჭირო აღარ არის ამ გუნდში, და იგი მას ათავისუფლებს. ამ ჭიანჭველის შესაბამისი სიმბოლოიანი ჩანაწერიც იშლება ჰემ-ცხრილიდან. თუმცა გათავისუფლებული ჭიანჭველა შეიძლება საჭირო იყოს სხვა გუნდისთვის, რომლის გურუმაც მისი სიმბოლოს მქონე ჭიანჭველის დამატების გადაწყვეტილება მიიღო, რაც შემდეგნაირად განისაზღვრება - თუ სიმულაციის გარკვეული ბიჯების შემდეგ რომელიმე სიმბოლოს კოეფიციენტი გასცდა გარკვეულ ზღვარს ( $\delta$ ), ე.ი. საჭიროა ამ სიმბოლოს ამომცნობი ჭიანჭველის დამატება გუნდში, რადგან მისი სამუშაო უხვად არის შესაბამის კოლონაში.

- ყოველი შემდეგი ჭიანჭველის დამატების ფსევდო კოდი

```
foreach(each i; i < HashTable.length; i++)
  {if(i.value >  $\delta$ ) AddNextAntInColony(i.key)}
```

- არასაჭირო ჭიანჭველის გათავისუფლება კოლონიიდან

```
foreach(each i; i < HashTable.length; i++)
  {if(i.value <  $-\delta$ ) RemoveAntFromColony(i.key)}
  {DecrementHashTable(i.key)}
```

## თავი 7. ჭიანჭველების ამოცანის პროგრამული არქიტექტურა

ჭიანჭველების ამოცანა არის თამაში, შესაბამისად, საუკეთესო ვარიანტი იქნებოდა სათამაშო ძრავის გამოყენება მისი იმპლემენტაციისათვის. დღესდღეობით ერთერთი პოპულარული სათამაშო ძრავი არის Unity 3D . აღვწეროთ ჭიანჭველათა ამოცანის არქიტექტურა, როგორც თამაშის მიმდინარეობა და დავწეროთ ზოგიერთი ფსევდოკოდი.

### 7.1 ჭიანჭველის, როგორც პროგრამული აგენტის სტრუქტურა

ნებისმიერ ჭიანჭველა აგენტს აქვს საერთო ფუნქცია და ცვლადი, რომელშიც გარკვეულ საერთო მონაცემებს ვინახავთ, მაგალითად, სიჩქარე, მოძრაობის რადიუსი, კლასის ტიპი, როლის ტიპი და ა.შ.

```
Class Ant : Object
```

```
{
```

```
Role objectRole;
```

```
Float speed;
```

```
Float actionRadius;
```

```
List <Ant> TeamMembers;
```

```
Void Move();
```

```
Void Action();
```

```
Void MakeDecision();
```

```
Void GenerateArgument();
```

```
Void GetArgument();
```

```
Void DeclineArgument();
```

```
Void AddARgumentInKnowledgeDb();
```

```
Void RemoveArgumentInKnowledgeDb();
```

```
Void MarkKnowledge(float x,float y);
```

```
Void GetMarkedKnowLedge(float x, float y);
```

```
Void AddTeamMember(Ant newAnt);
```



```
Void RemoveTeamMember(Ant oldAnt);  
}
```

საბაზისო კლასი ჭიანჭველა, რომელსაც საერთო ფუნქციები გააჩნია.ეს კლასი საერთოა, ნებისმიერი კონკრეტული შინაარსის ჭიანჭველის კლასი ამ კლასის შვილი უნდა იყოს.

**Role objectRole** - ამ ველში არის შენახული აგენტის საწყისი, ტიპი, მზვერავი იქნება, ფოთლის მჭრელი თუ გადამზიდი;

**Float speed** - მოძრაობის სიჩქარე;

**Float actionRadius** - ეს შეიძლება იყოს კონკრეტული ქცევის არეალი, ან ახალი აგენტის ცნობის რადიუსი.

**List <Ant> TeamMembers;** - გუნდის წევრები.

**Void Move()** - აგენტის მოძრაობა;

**Void Action()**- კონკრეტული მოქმედება, ეს ზოგადი ფუნქციაა და მოითხოვს შესაბამის შვილობილ ობიექტში შესაბამის იმპლემენტაციას;

**Void MakeDecision()**- გადაწყვეტილების მიღების ზოგადი ფუნქცია, მოითხოვს შესაბამის მემკვიდრეში შესაბამის იმპლემენტაციას;

**Void GenerateArgument()** - არგუმენტის გენერაცია, თვითოეულ კლასისთვის დამახასიათებელია შესაბამისი არგუმენტების ტიპები;

**Void GetArgument()**- არგუმენტის აღება ცოდნის ბაზიდან;

**Void DeclineArgument()**- შემომავალი, სხვა აგენტის არგუმენტის უარყოფა;

**Void AddKnowledgeInDb()** - ცოდნის დამატება ცოდნის ბაზაში;

**Void RemoveKnowledgeFromDb()** - ცოდნის წაშლა ცოდნის ბაზიდან.

**Void MarkKnowledge(float x, float y);** - ცოდნის მონიშვნა სიბრტყის კონკრეტულ ადგილას.

**Void GetMarkedKnowLedge(float x, float y);** - ცოდნის აღება სიბრტყის კონკრეტული ადგილიდან.

**Void AddTeamMember(Ant newAnt);** - ახალი გუნდის წევრის აყვანა გაერთიანებაში.

**Void RemoveTeamMember(Ant oldAnt);** - ძველი გუნდის წევრის გაშვება გაერთიანებიდან.

## 7.2 მზვერავი ჭიანჭველის ქცვის ძირითადი ფსევდოკოდი

განვიხილოთ მზვერავი ჭიანჭველის ქცვის ძირითადი ფსევდოკოდი. მისი მიზანია იპოვოს გარკვეული ობიექტები სიბრტყეზე და შეამოწმოს არის თუ არა ფოთოლი. კოლექტივში გაერთიანების მხრივ

```
Class Seeker : Ant
{
    Float globalSeekRadius;
    Float globalSpeed;
    // ovveride all neccesary functions
    Void Init()
    {
        globalSeekRadius = radius;
```

```

    globalSpeed = speed;
}
Void FindLeaves(GameWorld world)
{
    While(true)
    {
        WorldObject foundWorldObject = GetObjectInRadius(); // get some
objects of world
        If(foundWorldObject.hasMarkedKnowledge()) // if leave is already
marked
        {
            Continue; // continue search
        }
        Else
        {
            Knowledge knowledge = this.GetKnowledgeFromDatabase(); // get
knowledge
            If(knowledge.Match(foundObject)) // check if this is leaf
            {
                MarkKnowledgeOnLeaf(foundObject.position.x,foundObject.position.y)
                Continue;
            }
            Else // this object wasnt leaf, so continue search
            {
                Continue;
            }
        }
    }
}

```

```

    }
}
Void AddTeamMember(Ant newAnt)
{
    Argument currentAddArgument = getCurrentAddArgument();
    Foreach(Ant teamMember in TeamMembers)
    {
        If(currentAddArgument.addByRadius)
        {
            If(newAnt.actionRadius > teamMember.actionRadius)
            {
                RemoveTeamMember(teamMember);
                TeamMembers.Add(newAnt);
                globalSeekRadius = CalculateNewGlobalsSeekRadius();
                break;
            }
        }
        If(currentAddArgument.addBySpeed)
        {
            If(newAnt.speed > teamMember.speed)
            {
                RemoveTeamMember(teamMember);
                TeamMembers.Add(newAnt);
                globalSpeed = CalculateNewGlobalsSpeed();
                Break;
            }
        }
    }
}
}

```

```
}
```

## 7.2 ფოთლის მჭრელის მოქმედების ძირითადი ფსევდოკოდი

ფოთლის მჭრელის ძირითადი მოქმედება მდგომარეობს ფოთლის ნაწილებად დაჭრაში. მანდ უნდა გაითვალისწინოს წინა აგენტების მიერ დატოვებული

```
Class Rogue : Ant
{
  Bool hasAnyPartStabbed = false;
  Char symbol;
  int foundSymbolCount;
  Hashtable <char,int> TeamSymbolsTable;
  int symbolsCountToAddNewMember;
  int symbolsCountToRemoveNewMember;

  //
  // ovveride all necessary methods
  //
  Void FindLeaf(World world)
  {
    While(true)
    {
      WorldObject foundWorldObject = GetObjectInRadius(); // get some
```

```

objects of world
    If(foundWorldObject.IsLeaf() == false) // if object isn't leaf continue
search
    {
        Continue; // continue search
    }
    Else
    {
        If(foundWorldObject.SymbolNoMoreExists(symbol)==!SymbolNoMore)
            Continue;
        Else
        {
            If(foundWorldObject.ContainsSymbol(symbol))
                {
                    StabSymbolOnLeaf(symbol);
                    // mark symbol as found
                    MarkKnowledge(foundWorldObject.pos.x,
foundWorldObject.pos.y,symbol,true);
TeamSymbolsTable[symbol][symbolsCountToRemoveNewMember ++];
hasAnyPartStabbed = true;
Break;
                }
            Else
            {
                // mark symbol as not found
                MarkKnowledge(foundWorldObject.pos.x,
                    foundWorldObject.pos.y,symbol,false);
                TeamSymbolsTable[symbol][ symbolsCountToAddNewMember ++];
                continue;
            }
        }
    }

```

```

        }
    }
}

Void AddTeamMember(Ant newAnt)
{
    Foreach(Ant ant in TeamMembers)
    {
        If(newAnt.symbol == ant.TeamSymbolsTable[symbol][max->
symbolsCountToAddNewMember])
        {
            TeamMembers.Add(ant);
        }
    }
}
//
Void RemoveTeamMember()
{
    Foreach(Ant ant in TeamMembers)
    {
        If(newAnt.symbol == ant.TeamSymbolsTable[symbol][max->
symbolsCountRemoveMember])
        {
            TeamMembers.Remove(ant);
        }
    }
}
}

```

```
}
```

### 7.3 ჭიანჭველა-გადამზიდის მოქმედების ძირითადი ფსევდოკოდი

```
Class Transporter : Ant  
{  
  Float TotalWeightToLift;  
  // override all necessary methods  
  //  
  Void FindLeaf(World world)  
  {  
    While(true)
```



```

    {
        WorldObject foundWorldObject = world.GetObjectInRadius(); // get
some objects of world
        If(foundWorldObject.hasAnyPartStabbed == false) // if object hasn't
any stab continue
        {
            Continue; // continue search
        }
        Else
        {
            LeafPart[]optimalWeights =
GREEDY_SELECT(foundWorldObject.allStabbedParts,
TotalWeightToLift);
            TakeToHome(optimalWeights);
            MarkKnowledge(foundWorldObject.position.x,
foundWorldObject.position.y,
optimalWeights);
        }
    }

    break;
}
}

Void AddNewTeamMember(Ant newMember)
{
    TeamMembers =
GREEDY_SELECT_WEIGHT(newMember.TotalWeightToLift,
TotalMembers)
    If(TeamMembers.length > TeamMaxLength)
    {

```

```
RemoveMember(ant=>TeamMembers With minimal weight);
```

```
}
```

```
}
```

```
}
```

## დასკვნა

თემაში განვიხილეთ მრავალაგენტიან ვიდეოთამაშებში აგენტთა კოლექტიური გადაწყვეტილებები. მიმოვიხილეთ თამაშების ჟანრები. ვნახეთ, რომ ჟანრების მიხედვით ხელოვნური ინტელექტის ამოცანები ერთმანეთისგან განსხვავდება. თუმცა პრინციპი ერთია, აგენტის მოქმედება დინამიურ ან სტატიკურ გარემოში. აგენტთან ერთად კი, რაც მთავარია, მოთამაშის მოქმედება. მოთამაშე მთავარი ასპექტია ვიდეოთამაშში, ვის გარშემოც ტრიალებს მთელი ხელოვნური ინტელექტუალური სისტემა და ინტერაქცია.

განვიხილეთ აგენტთა სისტემები, ერთაგენტიანი, მულტიაგენტური. ასევე ინტელექტუალურ აგენტთა თეორია, აღვწერეთ რას ნიშნავს ერთაგენტიანი და მულტიაგენტური სისტემა, რითი განსხვავდება ერთმანეთისგან, რა მსგავსებებია და რა საჭიროების შემთხვევაში გამოიყენება ერთი, ან მეორე.

კოლექტიურ გადაწყვეტილებების გარჩევამ დაგვანახა ცნობილი პრობლემა ვიდეოთამაშებში და ზოგადად, ხელოვნურ ინტელექტში, ცოდნის გადაცემის პრობლემა. ვნახეთ რა მდგომარეობაა დღესდღეობით თანამედროვე ვიდეოთამაშებში ამ მხრივ, წარმოვადგინეთ ჩვენი ვერსია ცოდნის გადაცემისა, ცოდნის მარკირების ამოცანა. ვნახეთ, რომ აგენტთა როლების[14] გადანაწილებით და ერთდროული მოქმედებით გაცილებით სწრაფად და ადვილად იხსნება რთული სისტემები. ეს ნათლად დაგვანახა ჭიანჭველების ამოცანამ, რომელსაც თამაშის სახე მივეცით.

ჭიანჭველების ამოცანაში შემოვიტანეთ სამი კლასი, ამ სამი კლასის დაფუძნებით მოცემული ამოცანა დაიყო ქვეამოცანებათ, ქვეამოცანების შესრულება დამოკიდებული გახდადეთ თვითოეული კლასის ჭიანჭველაზე, რომლებიც ერთმანეთს ცოდნის მარკირებით ეკონტაქტებიან.

თემაში განვიხილეთ ასევე ვიდეოთამაშების არგუმენტაციის მექანიზმი, რომელიც მთავარი ამოსავალი წერტილია კოლექტიური გადაწყვეტილებებისა. ჩვენი არგუმენტაციის მექანიზმის იმპლემენტაცია ჭიანჭველების ამოცანაშიც წარმოვადგინეთ და ვაჩვენებთ, თუ როგორ

შეიძლება მოახდინოს გარკვეული კლასის ჭიანჭველამ გუნდის ფორმირება, რაც მთავარია, რეალურ დროში მიმდინარეობს.

ჩვენს ნაშრომში აღწერილი თემა გამოსადეგია ნებისმიერი მსგავსი ამოცანისთვის, სადაც საჭიროა მულტიაგენტური ტექნოლოგია. ვიდეოთამაშების სხვა დეველოპერებს შეუძლიათ პირდაპირ გამოიყენონ ეს ნაშრომი, რათა განახორციელონ ცოდნის მარკირების ამოცანა. ასევე შეუძლიათ აგენტთა გუნდის ფორმირება ჯილოდოს პრინციპზე დაფუძნებით.

ეს ამოცანა გამოდგება არამართო ვიდეოთამაშებში არამედ სხვა სფეროშიც. საჭიროა განვახორციელოთ ამ ამოცანის უნიფიცირება, როგორც უნიკალური ფრეიმვორქი, რომელიც ერგება ნებისმიერი სფეროს, სადაც მრავალაგენტიანი, მულტიპროცესულ, ხელოვნურ აგენტთა მოქმედებაა საჭირო.

## გამოყენებული ლიტერატურა

1. გიორგი აბელაშვილი, ზურაბ ბოსიკაშვილი, "ვიდეოთამაშებში მულტიაგენტური დასწავლის მეთოდების შესახებ" - მართვის ავტომატიზებული სისტემები, N2 (20), თბილისი, 2015 წელი, გვერდი 52. სტატიის ლინკი: [http://www.gtu.ge/Journals/mas/Referat/ASU-2015\(2\\_20\)/52\\_57.pdf](http://www.gtu.ge/Journals/mas/Referat/ASU-2015(2_20)/52_57.pdf)
2. გიორგი აბელაშვილი, ზურაბ ბოსიკაშვილი, "ხელოვნური ინტელექტის არგუმენტაციის მექანიზმის გამოყენება ვიდეოთამაშებში" - მართვის ავტომატიზებული სისტემები, N2 (20), თბილისი, 2015 წელი, გვერდი 58. სტატიის ლინკი: [http://www.gtu.ge/Journals/mas/Referat/ASU-2015\(2\\_20\)/58\\_65.pdf](http://www.gtu.ge/Journals/mas/Referat/ASU-2015(2_20)/58_65.pdf)
3. გიორგი აბელაშვილი, "ვიდეოთამაშებში აგენტთა გაერთიანებისა და კოლექტიური გადაწყვეტილებების შესახებ", მართვის ავტომატიზირებული სისტემები, N2 (21), თბილისი, 2016 წელი, გვერდი 138.
4. გიორგი არჩვაძე - სწავლების და ცოდნის მოპოვების მეთოდების დამუშავება და კვლევა მრავალკომპონენტურ მესხიერებთან სისტემებისთვის <http://www.nplg.gov.ge/dlibrary/collect/0002/000858/Dis.Archv adze%20G.pdf>
5. [http://en.wikipedia.org/wiki/Blackboard\\_system](http://en.wikipedia.org/wiki/Blackboard_system) , Wikipedia, the free encyclopedia.
6. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 2003
7. Yoav Shoham, Leyton-Brown K. (2009). Multiagent systems, Algorithmic, Game-Theoretic and Logical Foundations. Cambridge University Press , Cambridge, England
8. Hunt J. (2002). Blackboard Architectures. JayDee Technology Ltd 27.
9. L. Amgoud. A formal framework for handling conflicting desires. In Proc. 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'2003), pages 552–563, Aalborg, Denmark, 2003.
10. L. Amgoud, Y. Dimopolous, and P. Moraitis. A unified and general framework for argumentation-based negotiation. In Proc. 6th International Joint Conference on Autonomous Agents and Multi-Agents Systems (AAMAS'2007), Honolulu, Hawai'i, 2007.
11. L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-00), pages 31–38, 2000.
12. Wang P. (2013). NON-AXIOMATIC LOGIC A Model of Intelligent Reasoning. Copyright © 2013 by World Scientific Publishing Co. Pte. Ltd, 2013
13. Smith R. (2012). Aristotle's logic. in *The Stanford Encyclopedia of Philosophy*, ed. Zalta, E. N. (Spring 2012 edn.).
14. Merritt D. (2004). Best Practices for Rule-Based Application Development. <http://msdn.microsoft.com/en-us/library/aa480020.aspx>
15. Giorgi Archvadze, Zurab Bosikashvili, **MATHEMATIC MODEL AND ARGUMENTATION THEORY WITH MULTI AGENT BASED ARCHITECTURE**

- FOR PATTERN RECOGNITION**, Georgian Technical University [http://www.gtu.ge/Journals/mas/Referat/ASU-2015\(1\\_19\)/187\\_196.pdf](http://www.gtu.ge/Journals/mas/Referat/ASU-2015(1_19)/187_196.pdf)
16. Cavazza M., Mead S.J., Strachan A.I., Whittaker A., “A Blackboard System for Interpreting Agent Messages”, Proceedings GameOn 2000: International Conference on Intelligent Games & Simulation, UK, 2001.
  17. van Liere R., Harkes J., de Leeuw W., “A Distributed Blackboard Architecture For Interactive Data Visualization”, Proceedings of the conference on Visualization '98, 1998, pp. 225 – 231.
  18. Metzner C., Cortez L., Chacin D., “Using A Blackboard Architecture In A Web Application”, The Journal of Issues in Informing Science and Information Technology, Volume 2, 2005, pp. 743-756.
  19. [https://en.wikipedia.org/wiki/Boltzmann\\_equation](https://en.wikipedia.org/wiki/Boltzmann_equation) Wikipedia, The Free Encyclopedia
  20. [https://en.wikipedia.org/wiki/Greedy\\_algorithm](https://en.wikipedia.org/wiki/Greedy_algorithm) Wikipedia, The Free Encyclopedia
  21. Ponsen, M. J. V. and Spronck, P. H. M. (2004). Improving adaptive game AI with evolutionary learning. In Mehdi, Q. H., Gough, N. E., and Al-Dabass, D., editors, Proceedings of Computer Games: Artificial Intelligence, Design and Education