კიბერუსაფრთხოების სისტემების აგების პრინციპების შემუშავება და კვლევა პროგრამული უზრუნველყოფის სუსტი ადგილების ანალიზისა და მოდელირებისათვის

ზურაბ ბოსიკაშვილი, ვალერიანე გელოვანი
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

განხილულია ინფორმაციული სისტემების პროგრამული უზრუნველყოფის სუსტი (დაუცველი) ადგილების მოდელირებისა და ანალიზის საკითხები. მოწყვლადობის (სისუსტეების) შეფასება განსაზღვრავს, აიდენტიფიცირებს და კლასიფიცირებას უწევს უსაფრთხოების ხვრელებს (დაუცველ ადგილებს) კომპიუტერში, ქსელსა თუ საკომუნიკაციო ინფრასტრუქტურაში. მოწყვლადობის ანალიზს შეუძლია აგრეთვე შემოთავაზებული საპასუხო ზომების ეფექტურობის პროგნოზირება და ფაქტობრივი ეფექტურობს შეფასება მათი შესრულების შემდეგ. ნაშრომში შემოთავაზებულია პროგრამული უზრუნველყოფის მოწყვლადი ადგილების ანალიზის ალგორითმი გადაწყვეტილების მიღების ხისებრი გრაფის საფუძველზე, მისი შესაძლო შედეგების შეფასებით, შემთხვევითი მოვლენების, რესურსების ხარჯის და სარგებლიანობის თვალსაზრისით.

საკვანძო სიტყვები: კიბერუსაფრთხოება. პროგრამული უზრუნველყოფა. მოდელირება. ანალიზი. ინფორმაციული ტექნოლოგია.

1. შესავალი

ინფორმაციული და კომპიუტერული ტექნოლოგიების განვითარების თანამედროვე ეტაპზე, ენერგეტიკულ და გარემოს დაცვის პრობლემებთან ერთად, ერთერთი მნიშვნელოვანი პრობლემა ინფორმაციული უსაფრთხოებაა. ეს განპირობებულია კომპიუტერული და ქსელური ტექნოლოგიების შეჭრით ადამიანის საქმიანობის ყველა სფეროში, აგრეთვე დაგროვილი და გენერირებული ციფრული ინფორმაციის მოცულობის ექსპონენციალური ზრდით. უფრო მეტიც, ჩვენს სამყაროს შეიძლება ვუწოდოთ „გაფართოებადი ციფრული სამყარო“. ადამიანის საქმიანობა სულ უფრო დამოკიდებული ხდება ციფრულ გარემოზე და ამ სფეროში ნებისმიერმა შეფერხებამ შეიძლება მნიშვნელოვანი ზარალი მიაყენოს მას, როგორც მატერიალურად, ასევე ფიზიკურად და/ან მორალურად. ციფრული ინფორმაციის მოცულობის ზრდასთან ერთად იზრდება ინფორმაციის დატაცების, არასანქცირებული წვდომის, გამიზნული შეცვლის რისკები და მათი დანაშუალებრივი მიზნებით გამოყენების ცდუნებები. ეს პრობლემები კიდევ უფრო აქტუალური ხდება გლობალური ქსელების და ტექნოლოგიების არსებობის პირობებში. საფრთხეები მოსალოდნელია როგორც ქვეყნის შიგნიდან, ასევე გარედან [1].

ინფორმაციული სისტემების უსაფრთხოების უზრუნველყოფა, პირველ რიგში, მოიცავს მოწყვლდობის (სისუსტეების) გამოვლენის და აღმოფხვრის ღონისძიებებს, შეტევების აღმოჩენის და აღკვეთის სამუშაოებს, რომელიც ციკლური პროცესია (ნახ.1) [2].

ნახ. 1. ციკლური პროცესი

ინფორმაციული სისტემების სისუსტეებზე და საფრთხეებზე ანალიზი კვალიფიციურ სპეციალისტებს და დიდ რესურსებს მოითხოვს (შესაბამისად საკმაოდ ძვირი ჯდება). ამასთან სასურველია აიგოს მოდელი და ავტომატიზებული სისტემა, რომელიც ასეთ საქმიანობას გააადვილებდა. იმისთვის, რომ განხორციელდეს ინფორმაციული სისტემების უსაფრთხოების უზრუნველყოფის პროცესის სრულყოფა, საჭიროა მისი ადექვატური და ეფექტური მოდელის შექმნა. წარმოდგენილი პროცესი ძნელად ფორმალიზებადია, რადგანაც მასში გასათვალისწინებელია როგორც აპარატურულ-პროგრამული ასპექტები, ასევე ადამიანური ფაქტორები. შესაბამისად მოდელი უნდა ასახავდეს როგორც ფორმალურ, ასევე არაფორმალურ ასპექტებს. ამ მიმართულებით წარმოდგენილ პროექტში შემოთავაზებულია იმ გამოცდილების გამოყენება, რაც დაგროვებულია ხელოვნური ინტელექტის სფეროში [3].

ხელოვნურ ინტელექტში კარგად არის ცნობილი რთული ამოცანების ამოხსნის რედუქციული მეთოდი, როდესაც საწყისი ამოცანა დაიყვანება სხვა ცნობილ ამოცანათა მიმდევრობაზე, რომელთა ამოხსნა შესაძლებელია, ხოლო შემდეგ ცალკეულ ამონახსნთა სიმრავლეზე ხორციელდება მთლიანი ამოცანის ამოხსნა [4].

წარმოდგენილ პროექტში შემოთავაზებული მნიშვნელოვანი სიახლეა ინფორმაციული სისტემის უსაფრთხოების მოდელის რედუქცია ტესტური კონტროლის ზოგად მოდელში. ტექნიკური სისტემების ტესტური კონტროლი გულისხმობს სისტემის პროექტირების, აგების ან ფუნქციონირების პროცესში დაშვებული შეცდომების და უზუსტობების გამოვლენას ტესტური ზემოქმედების შერჩევის საშუალებით, ე.ი. სისტემის გამართული მუშაობის კონტროლს. ტექნიკური სისტემების დიაგნოსტირების შემთხვევაში სისტემაში არსებული ხარვეზების გამოვლენა ხორციელდება სისტემის შემავალი ზემოქმედებების შერჩევით. შესაბამისად ტესტური კონტროლის ამოცანა შეიძლება გაიგივებული იქნეს სისუსტეების ანალიზის ამოცანასთან, ხოლო შემოჭრათა აღმოჩენის და ანალიზის ამოცანა - ტექნიკური დიაგნოსტირების ამოცანასთან [5]. შესამაბისად ტესტური კონტროლის მეთოდოლოგია , მეთოდები და ალგორითმები შეიძლება გამოყენებული იქნას ინფორმაციული სისტემების უსაფრთხოების უზრუნველყოფის ამოცანებში. კერძოდ აიგოს ცალკეული კომპონენტების

და მთლიანად სისტემის მოდელები ბულის ფუნქციების ბაზაზე, რაც უსაფრთხოების რიცვხითი მოდელირების საშუალებას იძლევა.

სიახლეს წარმოადგენს ინფორმაციული სისტემის უსაფრთხოების უზრუნველყოფის ერთიანი პროცესის წარმოდგენა მოდელების იერარქიის სახით (ნახ.2).

ნახ.2

2. მეთოდები და ალგორითმები

კლასიფიკაცია აერთიანებს კონკრეტული ტიპის ალგორითმებს. მისი ერთ-ერთი კლასიკური ალგორითმია გადაწყვეტილებათა ხეები (Decision Tree) [1]. ჩვენ განვიხილავთ ისეთ გადაწყვეტილებათ ხეს, სადაც მონაცემები წარმოდგენილია ორობითი სახით. ალგორითმისათვის წინასწარაა ცნობილი/მოცემული:

- მატრიცა (რომელიც აგებულია გარკვეული ვექტორ-სვეტების საფუძველზე);
- საწყისი ენტროპია (გამოთვლილი სახით);

ალგორითმი მუშაობს შემდეგი პრინციპით:

- თითოეული ვექტორ-სვეტისათვის გამოვთვალოთ საშუალო ენტროპია AE_i;
- გამოთვლილი საშუალო ენტროპიებიდან ამოვარჩიოთ მინიმალური, რის შედეგადაც მოხდება წინა ეტაპზე მიღებული მატრიცის გაყოფა მარცხენა და მარჯვენა ქვემატრიცებად, ხოლო არჩეული ვექტორსვეტი კი იქნება ხის მიმდინარე კვანძი;
- გავიმეოროთ ეს პროცესი მანამ, სანამ არ მოხდება მთელი ხის აგება;
- ახლა განვსაზღვროთ თუ როგორ ხდება საშუალო ენტროპიის დათვლა თითოეული ვექტორ-სვეტისათვის:

1) f_i ვექტორ-სვეტისათვის დავთვალოთ დადებითი და უარყოფითი ელემენტების რაოდენობა მარცხენა და მარჯვენა კვანძისათვის (თითოეული f_i წარმოადგენს ხის კვანძს, რომელსაც აქვს მარცხენა [დადებითი კვანძი]) და მარჯვენა [უარყოფითი კვანძი] კვანძები). მარცხენა კვანძისთვის დადებითი ელემენტების რაოდენობა ტოლია იმ 0-ების რაოდენობის, რომლის შესაბამისი Y არის 0. მარცხენა კვანძისთვის უარყოფითი ელემენტების რაოდენობა ტოლია იმ 0-ების რაოდენობის, რომლის შესაბამისი Y არის 1. მარჯვენა კვანძისთვის დადებითი ელემენტების რაოდენობა ტოლია იმ 1-ის ტოლი მნიშვნელობების რაოდენობის, რომლისთვისაც შესაბამისი Y არის 0. მარჯვენა კვანძისთვის უარყოფითი ელემენტების რაოდენობა ტოლია იმ 1-ის ტოლი მნიშვნელობების რაოდენობის, რომლისთვისაც შესაბამისი Y არის 1;

1) ენტროპია მოცემულია შემდეგი ფორმულით (რადგან საუბარია ორობით სისტემაზე, აუცილებელია ორივე ენტროპიის გამოთვლა):

$$I_i(p_0,n_0) = -\frac{p_0}{p_{0+}n_0} log_2(\frac{p_0}{p_{0+}n_0}) - (1-\frac{p_0}{p_{0+}n_0}) log_2(1-\frac{p_0}{p_{0+}n_0})$$

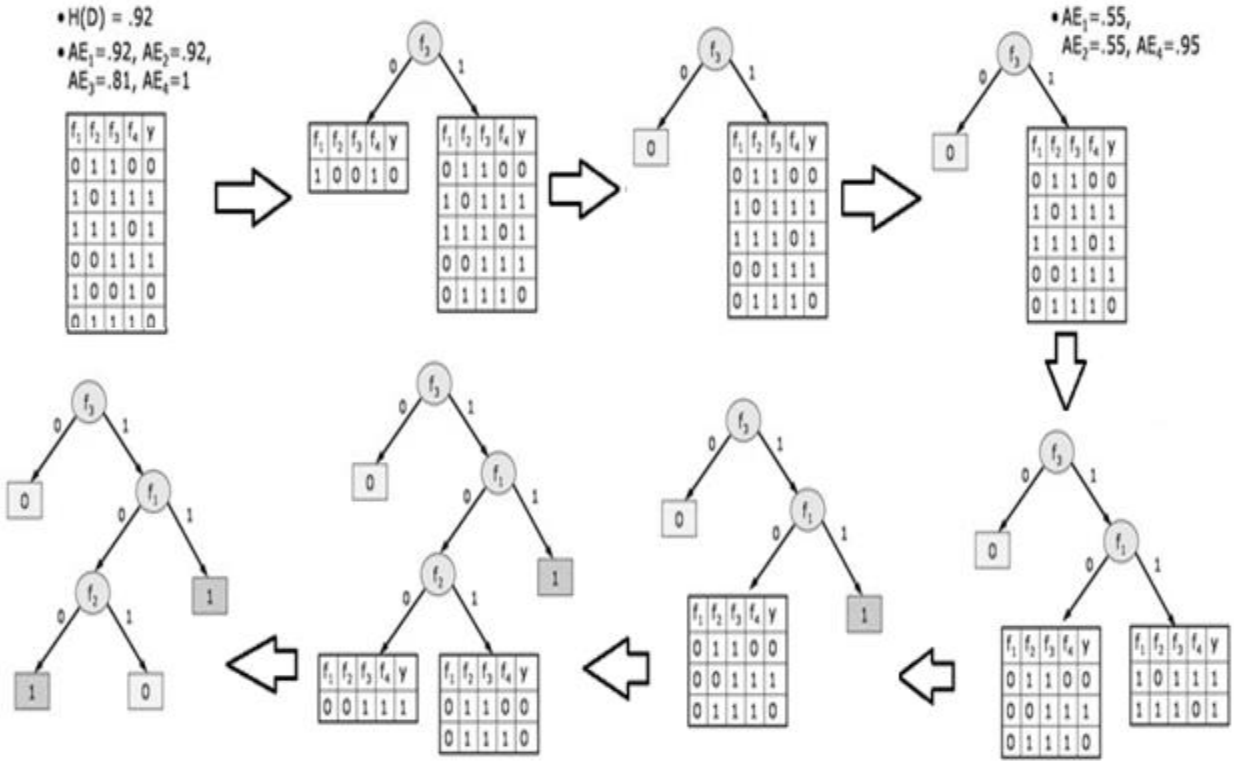
და

$$I_i(p_1,n_1) = -\frac{p_1}{p_{1+}n_1} log_2(\frac{p_1}{p_{1+}n_1}) - (1-\frac{p_1}{p_{1+}n_1}) log_2(1-\frac{p_1}{p_{1+}n_1});$$

2) ბოლოს უნდა გამოვთვალოთ AE_i შემდეგნაირად:

$$AE_i = \frac{p_0+n_0}{p+n} I_i(P_0,n_0) + \frac{p_1+n_1}{p+n} I_i(P_1,n_1);$$

ამ 3 ეტაპის შემდეგ ვარჩევთ ისეთ AE_i რომელიც არის მინიმალური, ვფოფთ მატრიცას ორ ნაწილად (მარცხენა და მარჯვენა ქვემატრიცებად. გაყოფა ხდება არჩეული ვექტორსვეტის საფუძველზე) და მთელ პროცესს ვიმეორებთ მანამ, სანამ არ მოხდება მთელი ხის აგება(ნახ.3).

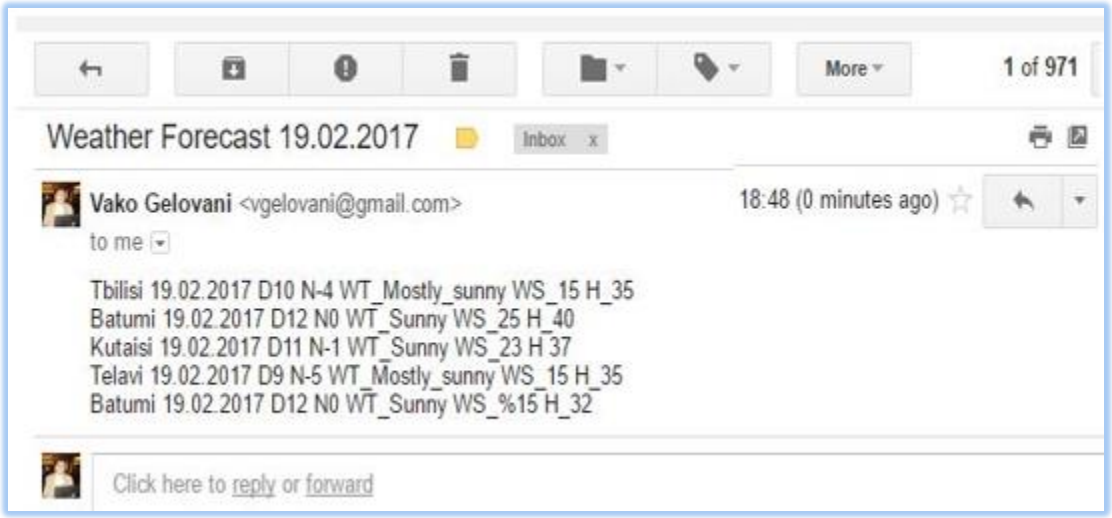


ნახ.3. გადაწყვეტილებეთა ხე, კონკრეტული მაგალითი

3. პრაქტიკული მაგალითი

მოვიყვანოთ მაგალითი: ჩვენ გვაქვს პროგრამილი უზრუნველყოფა და გვსურს ამ პროცესისთვის ერთიანი მოდელირების ფორმალიზმის შემუშავება. ჩვენ მიერ მოცემული პროგრამა შედგება რამდენიმე კომპონენტისგან. ესენია: ამინდის პროგნოზი, სავალუტო კურსი, კინო-აფიშა, ანეგდოტები და სხვ.

განვიხილოთ მისი ერთ-ერთი კომპონენტის ამინდის პროგნოზი. პროგრამის ამ კომპონენტს შეუძლია მეტეო-სადგურიდან ელექტრონული ფოსტის მეშვეობით გამოგზავნილი ინფორმაცია წაიკითხოს, დაამუშავოს და შემდეგ გადამუშავებული ინფორმაცია ჩაწეროს მონაცემთა ბაზაში. ელექტრონული ფოსტით მოდის ღია ტექსტი, სადაც არის ინფორმაცია ქალაქის შესახებ, თუ რომელ ქალაქის ამინდის პროგნოზზეა საუბარი, თარიღი- თუ რომელი დღის ამინდის პროგნოზია გამოგზავნილი, ტემპერატურა დღისით, ტემპერატურა ღამით, რა ტიპის ამიდნია მოსალოდნელი: წვიმიანი, თოვლიანი, სეტყვა, ღრუბლიანი, მზიანი და ა.შ, ქარის მგრძნობლობის სიჩქარე, ტენიანობა. მოვიყვანოთ მაგალითი მე-4 ნახაზზე.



ნახ. 4. ამინდის პროგნოზის მაგალითის ფრაგმენტი

აქ ასახულია ელექტრონული ფოსტით გამოგზავნილი ინფორმაცია ტექსტური სახით, ყოველგვარი ბმულის გარეშე. ჩვენმა პროგრამამ აღნიშნული ინფორმაცია წაიკითხა, გადაამუშავა და ჩაწერა მონაცემთა ბაზაში,

მონაცემთა ბაზაში გვაქვს ცხრილი, რომელიც შეიცავს შემდეგ ველებს (კომპონენტებს):

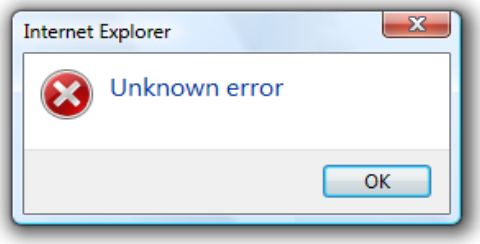
City, Date,Day_Temp, Night_Temp, Weather_type, Wind speed, Humidity, Check flag

როგორც ეს მე-5 ნახაზზეა მოცემული:

City	Date	Day_Temp	Night_Temp	Weather_type	Wind speed	Humidity	Check flag
Tbilisi	19.02.2017	10	-4	Mostly_sunny	15	35	1
Batumi	19.02.2017	12	0	Sunny	25	40	1
Kutaisi	19.02.2017	11	-1	Sunny	23	37	1
Telavi	19.02.2017	9	-5	Mostly_sunny	15%	32	0

ნახ. 5. მონაცემთა ბაზის ცხრილის ფრაგმენტი

მოცემულ მაგალითში ბოლო ველში Check flag-ში იწერება 2 ტიპის მნიშვნელობა: 1, როცა გამოგზავნილი ინფორმაცია სწორად იქნა გაგებული და ჩაწერილი და 0 როცა გამოგზავნილი ინფორმაცია შეიცავს შეცდომას. ჩვენმა პროგრამამ გამოიტანა შეცდომის ფანჯარა და შეწყვიტა ფუნქციონირება, როგორც ეს მე-6 ნახაზზეა ნაჩვენები.



ნახ. 6.

ელემენტის უსაფრთხოების ინციდენტი ვუწოდოთ მისი სასურველი მდგომარეობიდან არასასურველ მდგომარეობაში გადასვლას, რომელსაც იწვევს ელემენტზე მოქმედი სიდიდეები.

ელემენტის სისუსტე ვუწოდოთ ელემენტის ხარვეზს, ნაკლოვანებას, რომელმაც შეიძლება გამოიწვიოს არასწორი ფუნქციონირება (ინციდენტი).

ჩვენ მიერ გარჩეული ელემენტის შემავალი სიდიდეებია:

City, Date, Day_Temp, Night_Temp, Weather_type, Wind speed, Humidity,

რომელიც შეგვიძლია ასეც აღვნიშნოთ:

{a,b,c,d,e,f,g},

ხოლო გამომავალი სიდიდეა: Check flag. გამოსავალი სიდიდე აღვნიშნოთ {h}-ით.

ინციდენტი გვექნება იმ შემთხვევაში თუ მოცემული

{a,b,c,d,f,g,h} Tbilisi 19.02.2017 D10 N-4 WT_Mostly_sunny **WS_15** H_35 1.

სიდიდეების მაგივრად გვექნება

Tbilisi 19.02.2017 D10 N-4 WT_Mostly_sunny **WS_%15** H_35 1.

სიდიდეები i≡0 წერტილში დივერსიის ან ხელშეშლის გამო და Check flag სიგნალის მნიშვნელობა არ იცვლება.

თუ მოცემული {a,b,c,d,f,g} სიდიდეებისთვის შეიძლება ინციდენტის გამოწვევა და i წერტილში სიგნალის ცვლილებას ვერ ავსახავთ გამოსავალზე, მაშინ შეიძლება ითქვას, რომ {a,b,c,d, ,f ,g } ზემოქმედებებისთვის

{ Tbilisi 19.02.2017 D10 N-4 WT_Mostly_sunny **WS_15** H_35, Tbilisi 19.02.2017 D10 N-4 WT_Mostly_sunny **WS_%15** H_35 }

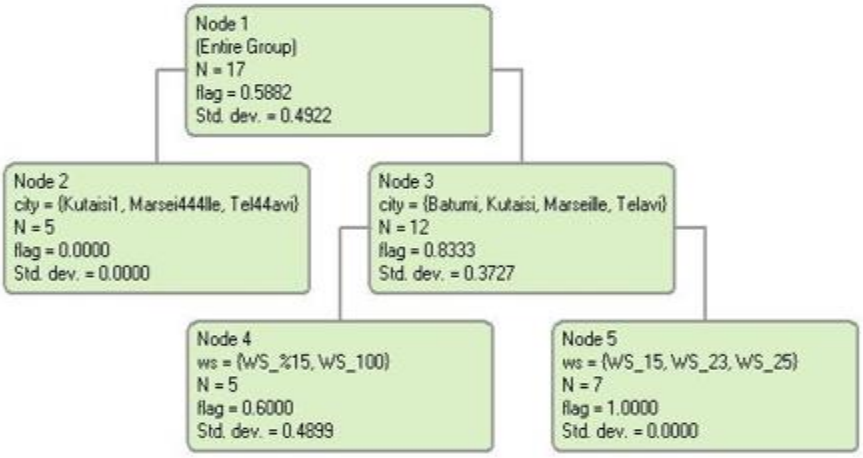
i წერტილი სისუსტეს წარმოადგენს , რადგანაც ამ ზემოქმედებების ფარგლებში ვერ მოხერხდება ამ წერტილში დივერსიის აღმოჩენა (გამოსავალზე ყოველთვის 1 გვექნება).

ჩვენი ძირითადი ამოცანაა ამ სისუსტეების აღმოჩენა. მანქანური დასწავლის მეთოდით შეიძლება ავაგოთ კომპონენტების ლოგიკური ფუნქციები და შემდეგ ვეძებოთ სისუსტეები სტანდარტული ალგორითმებით (ID3 ან სხვებით) [8,9].

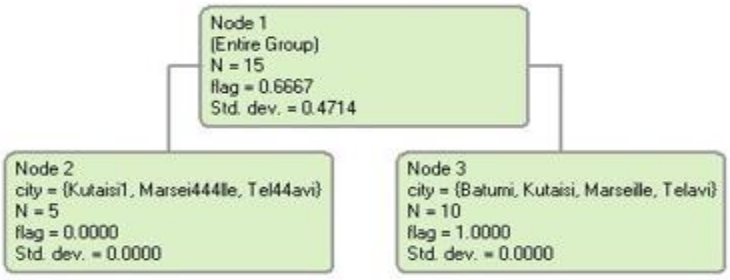
განხილული მაგალითის მიხედვით, დავწერეთ ჯავა პროგრამა, რომელიც მიღებულ ინფორმაციას ათავსებს ბაზაში და flag ველში წერს 0 ან 1-ს, იმის მიხედვით სწორად ჩაწერა ინფორმაცია თუ არა.

შემდეგ განვიხილეთ მანქანური დასწავლის სხვადასხვა ალგორითმი, მათ შორის ID3 და ბაზაში არსებულ მონაცემებზე დაყრდნობით ავაგე გადაწყვეტილებათა ხეები პროგრამული უზრუნველყოფის საშუალებით.

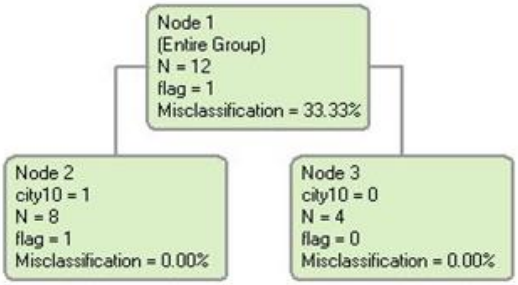
7-:10 ნახაზებზე ნაჩვენებია აგებული Dessision tree-ების გრაფიკულ გამოსახულებები. შემდეგ ბაზაში არსებული ინფორმაცია დავშალეთ კომპონენტებად, ავიღეთ მხოლოდ ერთი სვეტი და მისი check_flag. ინფორმაცია გადავიყვანეთ ორობით კოდში, თითოეული სიტყვა შევავსეთ სფეისებით, რომ ორობითში ყველა სიტყვას ტოლი სიგრძე ჰქონდეს, თითო სიმბოლო მივიჩნიეთ ერთ კომპონენტად და კვლავ ავაგეთ გადაწყვეტილებათა ხეები.



ნახ. 7.



ნახ. 8.



ნახ. 9.

```
city1,city2,city3,city4,city5,city6,city7,city8,city9,
          city10,city11,city12,city13,city14,city15,city16,flag
0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,1,1
0,1,1,1,0,0,0,0,0,1,1,1,0,0,1,0,1
0,1,1,0,1,0,1,1,0,1,1,1,0,1,0,1,1
0,1,1,0,1,0,1,1,0,1,1,0,1,0,0,1,1
0,1,1,0,1,0,1,1,0,0,1,1,0,0,0,1,0
0,1,1,0,1,0,1,1,0,0,1,1,0,0,1,0,0
0,1,1,1,0,1,0,0,0,1,1,0,0,0,1,0,1
0,1,1,1,0,1,0,0,0,1,1,0,0,1,0,1,1
0,1,1,1,0,1,0,0,0,0,1,1,0,0,1,0,0
0,1,1,1,0,1,0,0,0,1,1,0,0,1,0,1,1
0,1,1,1,0,1,0,0,0,1,1,1,0,1,0,0,1
0,1,1,1,0,1,0,0,0,0,1,1,1,0,0,1,0
0,0,1,1,1,0,0,1,0,1,1,1,0,1,0,0,0
```

ნახ. 10.

4. დასკვნა

ინფორმაციული და კომპიუტერული ტექნოლოგიები შეიჭრა ადამიანთა საქმიანობის თითქმის ყველა სფეროში და გლობალური ხასიათი მიიღო . ამიტომ ასეთი სისტემების უსაფრთხოების გაზრდასთან დაკავშირებული საკითხები ნებისმიერის სახელმწიფოსთვის სტრატეგიულ საკითხს წარმოადგენს. უსაფრთხოების სისუსტეები შეიძლება წარმოიშვას სისტემების პროექტირების და შექმნის, ადმინისტრირების და ასევე მათი გამოყენებისას.

შესაბამისად, სისტემის პოტენციური მომხმარებლებია:

- ინფორმაციული სისტემების მწარმოებლები;
- ინფორმაციული სისტემების გამსაღებლები;
- ინფორმაციული სისტემების სისტემური ადმინისტრატორები (მომსახურეები);
- ინფორმაციული სისტემების მოხმარებელები.

მომხმარებელთა ნებისმიერ ამ დონეზე სისტემა შეიძლება იყოს გამოყენებული, როგორც სისუსტეების ანალიზისთვის, ასევე მომსახურე პერსონალის და მომხმარებლების სასწავლო მიზნებისთვის.

ჩვენ მიერ წარმოდეგენილი პროექტი ინოვაციურია ამ სისტემებთან შედარებით, იმ კუთხით, რომ შემოთავაზებულ სისტემაში ჩადებულია ინტელექტუალური, მრავალდონიანი (სხვადასხვა დონის მოდელების), ინტეგრირებული, თვითსწავლებადი და ტესტირების სისტემების აგების პრინციპები ერთთიან კომპლექსში.

ლიტერატურა - References – Литература:

1. Bosikashvili Z. (2014). Extansion of the Architecture of Software Systems with Artificial Intelligence Elements, Modern Computer Applications in Science and Education. Editor Constantin Buzatu. Cambbridge, MA, USA, pp. 75-84

2. Bosikashvili Z., Kapanadze D., Zhvania T. (2010). About Unified Model of Safety of Information Systems. Reacent Advances in Computational Intelligance Proceedings of the 4yh WSEAS International Conference on Computational Intelligence (CI'10). Universitataea Politehnica, Bucharest, Romania, April 20-22, pp. 35-38

3. Bosikashvili Z. (2010). The blocking meta-heuristics for combinatorial problems solving. The ACM Digital Library, World Scientific and Engineering Academy and Society (WSEAS) Stevens Point, Wisconsin, USA

4. Bosikashvili Z., Lominadze T. (2009). Factorization of combinatorial problems with blocking meta-heuristics. The ACM Digital Library, World Scientific and Engineering Academy and Society (WSEAS) Stevens Point, Wisconsin, USA

5. Bosikashvili Z., Kapanadze D., Zhvania T. (2007). About formalization of the problem of the test control, Transactions Automated Control Systems #1(2), GTU, Tbilisi

6. Artificial Intelligence. Copyright © 2004 by Massachusetts Institute of Technology

7. Rokach L., Maimon O. (2014). Data mining with decision trees Theory and Applications. 2nd ed. World Scientific Publishing Co. Pte. Ltd. Series in "Machine Perception and Artificial Intelligence*

8. Han J., Kamber M., Pei J. (2013). Data Mining: Concepts and Techniques. 3rd ed. Univ. of Illinois at Urbana-Champaign & Simon Fraser University

9. Karibskiy V.V. (1967). Etc., Technical diagnostics of objects of the control, -M.: Energya.

DEVELOPING AND RESEARCHING THE PRINCIPLES OF BUILDING CYBER SECURITY SYSTEMS FOR SOFTWARE VULNERABILITY ANALYSIS AND MODELING

Bosikashvili Zurab, Gelovani Valeriane
Georgian Technical University

Summary

The problems of modeling and analysis of vulnerabilities of information systems software are considered. Vulnerability analysis, also known as vulnerability assessment, is a process that defines, identifies, and classifies the security holes (vulnerabilities) in a computer, network, or communications infrastructure. In addition, vulnerability analysis can forecast the effectiveness of proposed countermeasures and evaluate their actual effectiveness after they are put into use. The algorithm of the analysis of software vulnerabilities on the basis of a tree graph as a decision support tool and their possible consequences, including random events, resource costs and utility, is proposed.

РАЗРАБОТКА И ИССЛЕДОВАНИЕ ПРИНЦИПОВ ПОСТРОЕНИЯ СИСТЕМ КИБЕРБЕЗОПАСНОСТИ ДЛЯ АНАЛИЗА И МОДЕЛИРОВАНИЯ УЯЗВИМОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Босикашвили З., Геловани В.
Грузинский Технический Университет

Резюме

Рассматриваются вопросы моделирования и анализа уязвимостей программного обеспечения информационных систем. Оценка уязвимости представляет собой процесс, который определяет, идентифицирует и классифицирует дыры безопасности (уязвимости) в компьютерной, сетевой или коммуникационной инфраструктуре. Кроме того, анализ уязвимости может прогнозировать эффективность предлагаемых контрмер и оценивать их фактическую эффективность после их использования. В работе предлагается алгоритм анализа уязвимостей программного обеспечения на основе древовидного графа, как инструмента поддержки принятия решений, и его возможные последствия, включая случайные события, затраты ресурсов и полезность.