

UNIVERSITATEA TEHNICĂ „GHEORGHE ASACHI” IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DOMENIUL: Calculatoare și Tehnologia Informației
SPECIALIZAREA: Tehnologia Informației

Inteligență Artificială - Proiect
Inferența prin enumerare în rețele bayesiene

Coordonator,
Asist. drd. ing. Codruț-Georgian Artene

Autori,
Dima Raul Andrei, 1411B
Melinte Alexandru-Gicu, 1411B
Nistor Paula-Alina, 1411B

IAȘI

2021-2022

Cuprins

1. Descrierea problemei considerate	1
2. Aspecte teoretice privind algoritmul	1
2.1 Probabilitatea	1
2.2 Probabilitatea condiționată	1
2.3 Teorema lui Bayes	1
2.4 Independență și independență condiționată	2
2.5 Distribuție comună de probabilitate.....	2
2.6 Rețele Bayesiene	2
2.7 Sortarea topologică	2
2.8 Inferență prin enumerare.....	2
3. Modalitatea de rezolvare.....	3
4. Cod sursă, explicații, comentarii - blocuri semnificative	5
4.1 Metoda EnumerationAsk	5
4.2 Metoda EnumerateAll.....	6
4.3 Metoda KahnSorting – Sortare topologică	7
4.4 Constructorul rețelei bayesiene.....	8
5. Rezultate obținute	9
6. Concluzii	11
7. Rolul membrilor din echipă	11
8. Bibliografie	11

1. Descrierea problemei considerate

Proiectul vine ca și suport pentru descrierea unei aplicații ce primește date stocate într-un fișier .txt despre nodurile, legăturile dintre noduri și evidențele acestora. Aceste date reprezintă structura și parametrii unei rețele bayesiene, iar pe baza acestor date se dorește a se determina, cu ajutorul algoritmului de inferență prin enumerare, rezultatele probabilistice dorite pentru orice tip de rețea.

Mai exact se dorește ca aplicația să aibă posibilitatea interogării unui anumit nod din rețeaua bayesiană pentru a extrage date din diverse situații (prin setarea evidențelor) în care acesta este implicat.

2. Aspecte teoretice privind algoritmul

2.1 Probabilitatea

Dacă un eveniment **A**, ce face parte dintr-o mulțime de evenimente Ω , se poate realiza în s probe dintr-un total de n încercări echiprobabile pe care experimentul le poate produce, atunci probabilitatea evenimentului **A** se poate defini prin formula:

$$P(A) = \frac{\text{nr.cazuri favorabile}}{\text{nr.cazuri posibile}} = \frac{s}{n} \text{ și } 0 \leq P(A) \leq 1, \forall A \in \Omega \quad (2.1)$$

2.2 Probabilitatea condiționată

Dacă **A** și **B** sunt două evenimente arbitrare, prin **probabilitatea condiționată** a lui **A** de către **B**, notată $P(A|B)$, se înțelege probabilitatea de a se realiza evenimentul **A** dacă în prealabil s-a realizat evenimentul **B**. Prin definiție:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad (2.2)$$

Putem exprima probabilitatea intersecției în două moduri și de aici deducem expresia lui $P(B|A)$ în funcție de $P(A|B)$:

$$P(A \cap B) = P(A|B) \cdot P(B), \quad (2.3)$$

$$P(A \cap B) = P(B|A) \cdot P(A), \quad (2.4)$$

2.3 Teorema lui Bayes

Teorema lui Bayes descrie probabilitatea unui eveniment date fiind condițiile ce ar putea duce la apariția evenimentului.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}, \quad P(E|I) = \frac{P(E|I) \cdot P(I)}{P(E)}, \quad (2.5)$$

I - ipoteza;

E - evidența (provenită din datele observate);

P(I) - probabilitatea a-priori a ipotezei (gradul inițial de încredere în ipoteză);

P(E) - probabilitatea evidenței;

P(E|I) - verosimilitatea datelor observate (măsura în care s-a observat evidența în condițiile îndeplinirii ipotezei);

P(I|E) - probabilitatea a-posteriori a ipotezei, dată fiind evidența.

Cu ajutorul teoremei, putem calcula astfel probabilitățile cauzelor, date fiind efectele. Este mai simplu de cunoscut când o cauză determină un efect, dar invers, când cunoaștem un efect, probabilitățile cauzelor nu pot fi cunoscute imediat.

2.4 Independență și independență condiționată

Două evenimente A și B sunt **independente** dacă probabilitatea realizării (sau nerealizării) oricărui din cele două evenimente nu se modifică în funcție de realizarea, nerealizarea sau ignorarea celuilalt. Matematic, aceste noțiuni s-ar traduce în formula următoare:

$$P(A \cap B) = P(A) \cdot P(B) \quad (2.6)$$

Independența condiționată reprezintă independența bazată pe îndeplinirea unor condiții.

$$P(B|A) = P(B) = P(B|\bar{A}) \quad (2.7)$$

2.5 Distribuție comună de probabilitate

În probabilitate, având în vedere două variabile aleatorii X și Y, definite pe același spațiu de probabilitate, **distribuția lor comună** este definită ca distribuția de probabilitate asociată vectorului (X, Y).

2.6 Rețele Bayesiene

Literatura de specialitate oferă multe definiții generale ale unei rețele Bayesiene, redate în forme destul de diferite:

- O **rețea bayesiană** reprezintă o modalitate de vizualizare grafică și de analiză a modelelor care implică incertitudinea, incertitudine gestionată într-un mod matematic riguros, eficient și simplu.
- O **rețea bayesiană** este o reprezentare a unei distribuții comune de probabilitate a unui set de variabile cu posibile legături mutuale cauzale între ele.
- O **rețea bayesiană** este o reprezentare grafică a unor mărimi (și decizii) incerte care, în mod explicit, relevă dependența cauzală între variabile ca și circulația informației în modelul procesului analizat.
- O **rețea bayesiană** se referă la o metodă sistematică și concretă pentru structurarea informației cu caracter probabilistic într-un mod coerent și cu ajutorul unor algoritmi de inferență.
- O **rețea bayesiană** este un graf orientat aciclic (engl. “directed acyclic graph”), în care evenimentele sau variabilele se reprezintă ca noduri, iar relațiile de corelație sau cauzalitate se reprezintă sub forma arcelor dintre noduri.
- O **rețea bayesiană** este un model grafic probabilistic, adică un graf cu o mulțime de noduri, care reprezintă evenimente aleatorii, conectate de arce, care reprezintă dependențe condiționate între evenimente.

2.7 Sortarea topologică

Sortarea topologică a unui graf este o ordonare liniară a nodurilor sale astfel încât, pentru fiecare arc $A \rightarrow B$, A apare înaintea lui B. Pentru o **rețea bayesiană**, **sortarea topologică** asigură faptul că nodurile părinte vor apărea înaintea nodurilor fiu. Dacă există cicluri în graf, sortarea topologică este imposibilă.

2.8 Inferență prin enumerare

Inferența este un proces logic care derivă o concluzie dintr-o premisă, adică extrage o consecință necesară, o informație specifică, dintr-o descriere de stare dată.

Inferența prin enumerare are scopul de a calcula probabilitatea unei variabile interogate (engl. “query”), date fiind variabilele observate (evidență).

3. Modalitatea de rezolvare

Pentru interogarea unui nod dintr-o rețea bayesiană, utilizatorul dispune de o interfață unde poate vizualiza nodurile, evidențele și poate selecta nodul dorit. Se va încărca un fișier .txt ce conține parametrii rețelei, adică numărul de noduri, evidența fiecărui nod în parte, relațiile dintre noduri și probabilitățile.

Node	Evidence	Query
Flu -> Parents {}	NotPresent	<input type="radio"/>
Abscess -> Parents {}	NotPresent	<input type="radio"/>
Fever -> Parents {Flu, Abscess}	NotPresent	<input type="radio"/>
Fatigue -> Parents {Fever}	NotPresent	<input type="radio"/>
Anorexia -> Parents {Fever}	NotPresent	<input type="radio"/>

Select the node

Query the node

Figura 1

În secțiunea help a aplicației se regăsește un model .txt ce conține descrierea rețelei bayesiene (Figura 2).

```

Nodes: 5
[Yes, No]
Flu: {}
0.1 0.9

[Yes, No]
Abscess: {}
0.05 0.95

[Yes, No]
Fever: { Flu, Abscess}
Yes Yes 0.8 0.2
Yes No 0.7 0.3
No Yes 0.25 0.75
No No 0.05 0.95

[Yes, No]
Fatigue: { Fever}
Yes 0.6 0.4
No 0.2 0.8

[Yes, No]
Anorexia: { Fever}
Yes 0.5 0.5
No 0.1 0.9
    
```

OK

Figura 2

Prin selectarea unui nod și/sau a unor evidențe, la apăsarea butonului *Query the node* se va apela algoritmul de inferență prin enumerare și va furniza probabilitatea nodului selectat pentru rețeaua bayesiană.

Node	Evidence	Query
Flu -> Parents []	NotPresent	<input type="radio"/>
Abscess -> Parents []	NotPresent	<input type="radio"/>
Fever -> Parents [Flu Abscess]	NotPresent	<input checked="" type="radio"/>
Fatigue -> Parents [Fever]	Yes	<input type="radio"/>
Anorexia -> Parents [Fever]	Yes	<input type="radio"/>

Query Results

P(Yes) = 0.680823915421072
P(No) = 0.319176084578928

The node {Flu} was set with evidence {NotPresent}.
The node {Abscess} was set with evidence {NotPresent}.
The node {Fever} was set with evidence {NotPresent}.
The node {Fatigue} was set with evidence {Yes}.
The node {Anorexia} was set with evidence {Yes}.
The node {Fever} was queried with evidence {NotPresent}.

Figura 3

În acest exemplu (Figura 2), se consideră că atât gripa (*Flu*) cât și abcesul (*Abscess*) pot determina febra (*Fever*). De asemenea, febra poate cauza o stare de oboseală (*Fatigue*) sau anorexie (*Anorexia*).

Fiecare variabilă are o mulțime de valori. În cazul cel mai simplu, variabilele au evidențe binare, de exemplu Da (Yes) și Nu (No). În general însă, o variabilă poate avea oricâte evidențe.

Asociate cu variabilele, o rețea bayesiană conține o serie de tabele de probabilități, precum cele din Figura 2. Pentru nodurile fără părinți se indică probabilitățile marginale ale fiecărei valori (adică fără a lua în considerare valorile celorlalte variabile). Pentru celelalte noduri, se indică probabilitățile condiționate pentru fiecare valoare, ținând cont de fiecare combinație de valori ale variabilelor părinte. În general, o variabilă binară fără părinți va avea un singur parametru independent, o variabilă cu 1 părinte va avea 2 parametri independenți iar o variabilă cu n părinți va avea 2^n parametri independenți în tabela de probabilități corespunzătoare.

Folosindu-ne de algoritmul de inferență prin enumerare, avem posibilitatea de a răspunde la orice întrebare privind nodurile din rețea setate cu diferite evidențe.

Ideea de bază la algoritmul de inferență prin enumerare este calcularea unui produs de probabilități condiționate, însă în cazul variabilelor despre care nu se cunoaște nimic (*NotPresent*), se sumează variantele corespunzătoare tuturor valorilor acestora.

Exemplu dat răspunde la întrebarea „Care este probabilitatea de a avea febra (*Fever*), dacă prezintă simptome de oboseală (*Fatigue*) și anorexie (*Anorexia*)?”, obținându-se:

$$P(\text{Yes}) \cong 0.69$$

$$P(\text{No}) \cong 0.31$$

4. Cod sursă, explicații, comentarii - blocuri semnificative

4.1 Metoda EnumerationAsk

```

/// <summary>
/// Method that calculate all probabilities for a query node
/// </summary>
/// <param name="queryVariable">Query node</param>
/// <returns>An array of probabilities</returns>
1 reference
public double[] EnumerationAsk(String queryVariable)
{
    var nodes = _bayesNetwork.NetworkGraph.KahnSorting();

    var queryNode = _bayesNetwork.NetworkGraph.GetNode(queryVariable);

    if (queryNode == null) throw new Exception("Query node - not found!");

    double[] q = new double[queryNode.GetEvidenceDomain().Count];

    foreach (var node in nodes)
    {
        var nodeEvidenceDomain = node.GetEvidenceDomain();
        if (node.NodeID == queryVariable && node.Evidence != Node.NOT_PRESENT)
        {
            int j = 0;
            foreach (var typeOfEvidence in nodeEvidenceDomain)
            {
                if (typeOfEvidence == Node.NOT_PRESENT)
                    continue;

                if (node.Evidence == typeOfEvidence)
                {
                    q[j] = 1.0;
                }
                else
                {
                    q[j] = 0.0;
                }
                ++j;
            }
            return q;
        }
    }

    int i = 0;

    foreach (var typeOfEvidence in queryNode.GetEvidenceDomain())
    {
        if (typeOfEvidence == Node.NOT_PRESENT)
            continue;

        _bayesNetwork.NetworkGraph.SetNodeEvidence(queryVariable, typeOfEvidence);
        var vars = CopyNodesList(nodes);

        _bayesNetwork.NetworkGraph.SetNodeEvidence(queryVariable, typeOfEvidence);

        q[i] = EnumerateAll(vars, queryNode.GetEvidenceDomain());

        _bayesNetwork.NetworkGraph.SetNodeEvidence(queryVariable, Node.NOT_PRESENT);
        i++;
    }

    return Normalization(q);
}

```

Figura 4

Metoda EnumerationAsk primește ca parametru de intrare nodul ce se dorește a fi interogat și furnizează probabilitățile în funcție de evidențe. Se construiesc variabilele funcției: lista de noduri sortată topologic (*nodes*), nodul din graf ce se dorește a fi interogat (*queryNode*) și un vector ce conține probabilitățile (*q*). În caz de nodul interogat nu există în graful rețelei, se va genera o excepție cu mesajul *Query node – not found!*. Se vor calcula probabilitățile pentru nodul interogat în funcție de evidențele selectate. Dacă evidențele sunt setate pe valoarea *NOT_PRESENT*, se vor genera probabilitățile marginale. În caz contrar, se vor genera probabilitățile calculate în situațiile alese. Se va returna vectorul de valori normalizat.

4.2 Metoda EnumerateAll

```

/// <summary>
/// Recursive metod that calculate probability of a node in a given situation
/// </summary>
/// <param name="vars">List of nodes</param>
/// <param name="domain">Evidences domain</param>
/// <returns>Probability in a given situation</returns>
3 references
private double EnumerateAll(List<Node> vars, List<String> domain)
{
    if (vars.Count == 0)
        return 1.0;

    var y = vars.First();
    vars.Remove(y);

    List<Node> parents = new List<Node>();

    foreach (var node in _bayesNetwork.NetworkGraph.Nodes)
    {
        if (y.IsChidOf(node.NodeID))
        {
            parents.Add(node);
        }
    }

    var evidence = "";

    if (parents.Count == 0)
    {
        evidence = "p";
    }
    else
    {
        foreach (var parent in parents)
        {
            evidence += parent.Evidence + " ";
        }

        evidence = evidence.Remove(evidence.Length - 1, 1);
    }

    if (y.Evidence != Node.NOT_PRESENT)
    {
        return _bayesNetwork.NetworkGraph.GetProbabilityOfNode(y.NodeID, y.Evidence, evidence) * EnumerateAll(vars, domain);
    }
    else
    {
        double sum = 0.0;
        foreach (var ev in domain)
        {
            if (!y.GetEvidenceDomain().Contains(ev))
                continue;
            _bayesNetwork.NetworkGraph.SetNodeEvidence(y.NodeID, ev);
            sum += _bayesNetwork.NetworkGraph.GetProbabilityOfNode(y.NodeID, ev, evidence) * EnumerateAll(CopyNodesList(vars), domain);
        };
        _bayesNetwork.NetworkGraph.SetNodeEvidence(y.NodeID, Node.NOT_PRESENT);
        return sum;
    }
}

```

Figura 5

Metoda EnumerationAll este o metodă recursivă ce primește ca parametri de intrare lista de noduri și domeniul de evidențe și va returna probabilitatea nodului în situația dată (evidențele selectate). În caz de numărul de noduri este 0, se va returna valoarea reală 1.0. Altfel, se ia fiecare nod în parte și se verifică dacă are părinți, care este numărul lor și care sunt evidențele. În caz de evidențele selectate corespund valorii *NOT_PRESENT*, se va calcula în funcție de probabilitățile marginale. În caz contrar, se calculează recursiv probabilitatea fiecărui nod.

4.3 Metoda KahnSorting – Sortare topologică

```

/// <summary>
/// Kahn Sorting - Topological sort
/// </summary>
/// <returns>List of nodes</returns>
1 reference
public List<Node> KahnSorting()
{
    List<Node> L = new List<Node>();
    List<Node> S = new List<Node>();
    int[] indegree = new int[_noNodes];

    for (int i = 0; i < _noNodes; ++i)
    {
        indegree[i] = _nodes[i].ParentsNumber;
    }

    foreach (var node in _nodes)
    {
        if (node.HasParent == false)
        {
            S.Add(node);
        }
    }

    int cnt = 0;
    while (S.Count != 0)
    {
        var tempNode = S.First();
        L.Add(tempNode);
        S.RemoveAt(0);

        var aux = _nodes.Where(node => node.IsChildOf(tempNode.NodeID) == true).ToList();
        foreach (var node in aux)
        {
            if (--indegree[_nodes.IndexOf(node)] == 0)
            {
                S.Add(node);
            }
        }
        cnt += 1;
    }

    if (cnt != _nodes.Count)
        throw new Exception("The graph has cycles!");

    return L;
}

```

Figura 6

Metoda KahnSorting este metoda echivalentă cu sortarea topologică, adică se vor sorta vârfurile unui graf orientat aciclic astfel încât, dacă există o legătură directă între nodul A și nodul B (adică un arc (A, B)), atunci nodul A va apărea înaintea nodului B în lista de noduri sortată.

```

/// <summary>
/// Default constructor
/// </summary>
/// <param name="filePath">The path of the file which contains the parameters of the bayesian network</param>
1 reference
public BayesNetwork(string filePath)
{
    List<string> lines = System.IO.File.ReadLines(filePath).ToList();

    int noNodes = Int32.Parse(lines[0].Split(':')[1].Trim());
    _networkGraph = new Graph(noNodes);
    int pos = 2;

    //number of nodes
    for (int i = 0; i < noNodes; ++i)
    {
        //start from 3rd line
        //read first line with nodes
        var evidences = lines[pos].TrimStart('[').TrimEnd(']').Replace(" ", "").Split(',');
        var temp = lines[++pos].Split(':');
        var nodeID = temp[0].Trim();
        var parents = Regex.Replace(temp[1], @"\{\} \t", "").Split(',');
        var node = new Node(nodeID);

        foreach (var e in evidences)
        {
            node.AddEvidenceToDomain(e);
        }

        if (parents.Length == 1 && string.IsNullOrEmpty(parents[0]))
        {
            _networkGraph.AddNodes(node, null);
            _networkGraph.SetProbabilities(nodeID, lines.GetRange(pos + 1, 1));
            pos += 3;
        }
        else
        {
            _networkGraph.AddNodes(node, parents);
            var noOfProb = 1;

            foreach (var parent in parents)
            {
                noOfProb *= _networkGraph.GetNode(parent).GetEvidencesNumber();
            }

            _networkGraph.SetProbabilities(nodeID, lines.GetRange(pos + 1, noOfProb));
            pos += noOfProb + 2;
        }
    }

    _networkGraph.PrintNodesProbabilities();
}

```

Figura 7

5. Rezultate obținute

Întrebare: Care este probabilitatea de a avea gripă (*Flu*) dacă ai oboseală (*Fatigue*) și anorexie (*Anorexia*)?

Rezultat: $P(\text{Yes}) \cong 0.39$
 $P(\text{No}) \cong 0.61$

Vizualizare interfață:

Node	Evidence	Query
Flu -> Parents []	NotPresent	<input checked="" type="radio"/>
Abscess -> Parents []	NotPresent	<input type="radio"/>
Fever -> Parents [Flu Abscess]	NotPresent	<input type="radio"/>
Fatigue -> Parents [Fever]	Yes	<input type="radio"/>
Anorexia -> Parents [Fever]	Yes	<input type="radio"/>

Query Results

P(Yes) = 0.396281443674809
P(No) = 0.603718556325191

The node [Flu] was set with evidence [NotPresent].
The node [Abscess] was set with evidence [NotPresent].
The node [Fever] was set with evidence [NotPresent].
The node [Fatigue] was set with evidence [Yes].
The node [Anorexia] was set with evidence [Yes].
The node [Flu] was queried with evidence [NotPresent].

Figura 8

Întrebare: Care este probabilitatea de a avea gripă (*Flu*) dacă ai oboseală (*Fatigue*) și anorexie (*Anorexia*)?

Rezultat: $P(\text{Yes}) = 0.2$
 $P(\text{No}) = 0.8$

Vizualizare interfață:

Node	Evidence	Query
Flu -> Parents []	Yes	<input type="radio"/>
Abscess -> Parents []	NotPresent	<input type="radio"/>
Fever -> Parents [Flu Abscess]	No	<input type="radio"/>
Fatigue -> Parents [Fever]	NotPresent	<input checked="" type="radio"/>
Anorexia -> Parents [Fever]	NotPresent	<input type="radio"/>

Query Results

P(Yes) = 0.2
P(No) = 0.8

The node [Flu] was set with evidence [Yes].
The node [Abscess] was set with evidence [NotPresent].
The node [Fever] was set with evidence [No].
The node [Fatigue] was set with evidence [NotPresent].
The node [Anorexia] was set with evidence [NotPresent].
The node [Fatigue] was queried with evidence [NotPresent].

Figura 9

Întrebare: Care este probabilitatea de a se întâmpla evenimentul alarmă (*Alarm*) dacă nu s-a întâmplat un eveniment de tip hoț (*Burglary*), cutremur (*Earthquake*), dar s-au întâmplat evenimentele de tip apel Ionuț (*JohnCalls*) și apel Maria (*MarryCalls*)?

Rezultat: $P(\text{Yes}) \cong 0.56$

$P(\text{No}) \cong 0.44$

Vizualizare interfață:

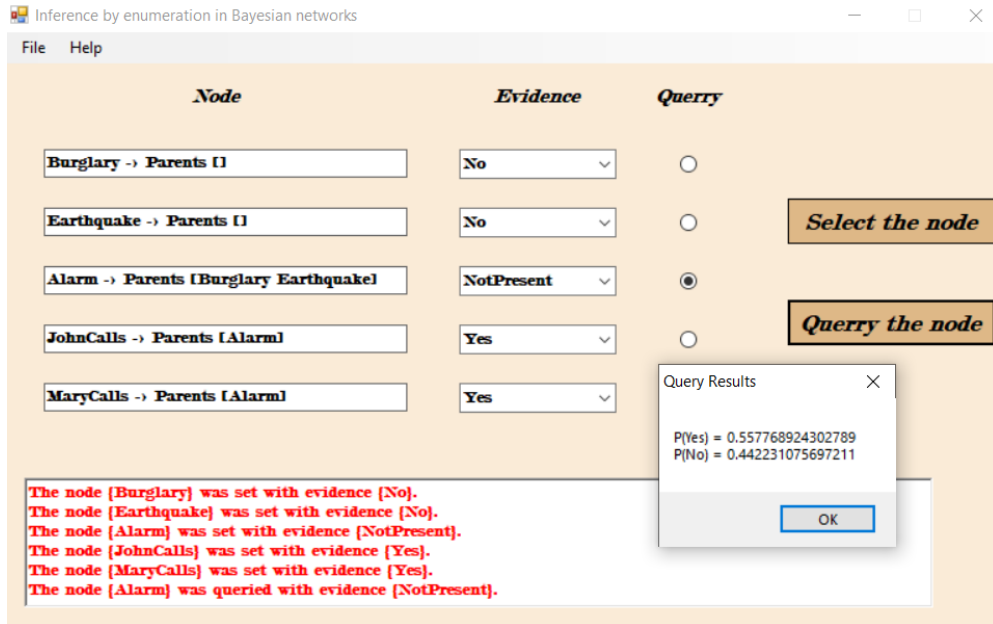


Figura 10

Întrebare: Dacă o mașină este pe segmentul S1 (*Section1*), banda B1 (*Banda1*), care este probabilitatea ca după intersecție, pe drum (*Drum*) să meargă înainte (*Înainte*)?

Rezultat: $P(\text{Banda1}) \cong 0.36$

$P(\text{Banda3}) \cong 0.10$

$P(\text{Banda2}) \cong 0.54$

$P(\text{Banda4}) = 0$

Vizualizare interfață:

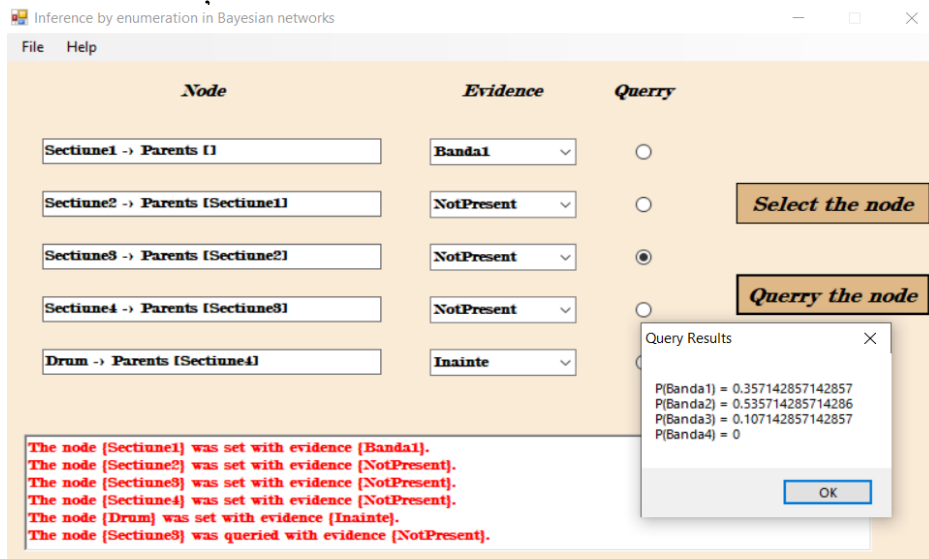


Figura 11

6. Concluzii

Aplicația respectă cerințele impuse și anume, cu ajutorul algoritmului de inferență prin enumerare se calculează interoghează un nod și se furnizează mai multe probabilități în funcție de situația aleasă și oferă posibilitatea utilizatorului de a crea propria rețea bayesiană prin intermediul unui fișier .txt.

Aplicația ar putea fi îmbunătățită prin crearea unei interfețe mai complexe, unde se pot desena automat și grafurile în funcție de parametrii rețelei din fișier.

7. Rolul membrilor din echipă

- **Dima Raul Andrei** – structurile rețelei (clasa Node, Graph, BayesNetwork), algoritmul de inferență prin enumerare.
- **Melinte Alexandru-Gicu** – interfața cu utilizatorul, documentație, comentarii cod.
- **Nistor Paula-Alina** – algoritmul de inferență prin enumerare (clasa InferenceByEnumeration), crearea domeniului de evidențe, creare fișiere de configurare a rețelei bayesiene.

8. Bibliografie

- [http://www.doctorat.tuiasi.ro/doc/SUSTINERI_TEZE/ETH/Ciobanu%20\(Aionoae\)/Ciobanu%20Aionoae%20Alexandra%20rezumat%20teza.pdf](http://www.doctorat.tuiasi.ro/doc/SUSTINERI_TEZE/ETH/Ciobanu%20(Aionoae)/Ciobanu%20Aionoae%20Alexandra%20rezumat%20teza.pdf)
- https://en.wikipedia.org/wiki/Bayesian_network
- http://florinleon.byethost24.com/Curs_IA/IA10_ReteleBayesiene.pdf?i=1
- https://koaha.org/wiki/Distribuzione_multivariata