FACTA UNIVERSITATIS (NIŠ)
Ser. Math. Inform. 28, No 1(2013), 27–42

PROBABILITY GRAPHS

Igor Ševo

Abstract. Here, I propose the idea of using graphs (graph theory) for modeling, analysis and solving problems from probability theory. Here, algorithms are proposed which allow finding the most probable sequences of events in a given set of events. Also, an idea of modeling probability problems with graphs is proposed.

Keywords: Probability, graph, simplification, transition matrix, weight, Floyd-Warshall algorithm.

1. Introduction

In a model of a system of random events, there exist mutual relations between individual events [1]. Each two events from a given set of events have some relation, whether it is causality or inclusion. For different problems, different systems are modeled differently, but generally they all have some similar properties.

The idea proposed here is that each such system can be modeled using a graph. It is known that for a given system of events there exists a transition matrix which describes transition probabilities between given events [1, 2]. By combining automata theory [3] and probability theory, graph systems can be deduced which can model systems of random events.

If events are represented via nodes (vertices) of a graph, then edges of a graph would represent relations between individual events. From the transition matrix of a system of events, a graph of the system can be modeled such that the edges of that graph have weights corresponding to the transition probabilities between related events. To model an event system more accurately, the definition of the graph can be modified so that it contains more properties of the given random event system. Moreover, as the topic is random events systems, search algorithms need to be modified according to properties of random event systems. In this case the complete path (probability) between two given events would be obtained by

Received August 08, 2012.; Accepted January 23, 2013.
2010 *Mathematics Subject Classification.* Primary 60A99; Secondary 60J99, 05C85

multiplication of weights of edges that lead from one node to another (as opposed to summing the edge weights).

By applying automata theory and graph theory I deduce some rules for manipulation and analysis of graphs that describe random events systems. As the definition of a graph is modified here, graphs used here are named probability graphs.

2. Mathematical model

Definition 2.1. For n events $A_i \subset \Omega, i = \overline{1,n}$, event system is an ordered pair $S = (M_s, X)$, where X is a set such that $X = \{A_1, A_2 \ldots A_n\}$, and M_s is a causality matrix (transition matrix) defined as:

$$M_s = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ P_{21} & P_{22} & & P_{2n} \\ \vdots & & \ddots & \vdots \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{bmatrix},$$

where $P_{ij} = P(A_i A_j)$ is the occurrence probability of the event A_i immediately after the occurrence of the event A_j.

Definition 2.2. Event system S is closed if:

$$\sum_{i=1}^{n} P_{ij} = 1$$

for $j = \overline{1,n}$ i $P_{ij} \in M_s \in S$.

Definition 2.3. For each event system S there exists a directed, weighted graph G_s which is called the probability graph of the event system S, which completely describes the system S and which is defined in the following way:

$G_s = (V, E)$, where V is a set of graph vertices, where each vertex corresponds to a single event in S, $|V| = |X|, X \in S$; and E is a set of graph edges, where each edge is defined as a tuple (x, y, p), so that $p = P(A_y A_x)$, that is $p = P(A_y A_x) = P_{yx} \in M_s \in S$, where $p > 0$, meaning there are no relations between vertices for which the corresponding events have a zero transition probability.

Weight of edge (x, y, p) is defined as $\delta(x, y) = p = P(A_y A_x)$, that is, weight of edge (x, y, p) is equal to the value of the third member of the given tuple, so it is equal to the occurrence probability of y-th event with occurrence of x-th event.

Definition 2.4. Let $S(i,j) \in G_s$ be a walk in a probability graph G_s. The weight of the walk $S(i,j)$ is written as $\omega(S(i,j))$ and computed as:

$$\omega(S(i,j)) = \prod_{(p,q,\cdot)\in S(i,j)} \delta(p,q)$$

Theorem 2.1. *Let G_s be a probability graph which describes event system S. The occurrence probability of the event A_j starting from event A_i, for $A_j, A_i \in X \in S$, is equal to the sum of weights of all walks from i-th to j-th vertex in a graph, that is:*

$$P(A_jA_i) = \sum_{S(i,j)\in G_s} \omega(S(i,j))$$

Proof. Every vertex from graph G_s corresponds to an event in S, and every edge corresponds to the transitional probability between two events corresponding to the vertices it connects. For two events A_x and A_y, for which holds $P(A_yA_x) = 0$ exists no edge in graph G_s which connects A_x with A_y, that is the edge $(x, y, \cdot)$ does not exist.

Therefore, every sequence of events $A_iA_{k_1}A_{k_2}\dots A_{k_r}A_j$ with probability

$$P(A_j|A_{k_r}|A_{k_{r-1}}|\cdots|A_{k_2}|A_{k_1}|A_i) > 0$$

exists as a walk in the graph G_s . If $P(A_j|A_{k_r}|A_{k_{r-1}}|\cdots|A_{k_2}|A_{k_1}|A_i) = 0$ holds for each sequence, then there exists at least one probability $P(A_nA_m) = 0$ in the given sequence, for which in graph G_s a corresponding edge does not exist, hence for impossible sequences of events a corresponding walk in the graph does not exist.

For sequence of events $A_iA_{k_1}A_{k_2}\dots A_{k_r}A_j$ there exists a walk with the weight equal to the probability of the given sequence, that is if:

$$P(A_j|A_{k_r}|A_{k_{r-1}}|\cdots|A_{k_2}|A_{k_1}|A_i)=P(A_jA_{k_r})\cdot P(A_{k_r}A_{k_{r-1}})\cdots P(A_{k_2}A_{k_1})\cdot P(A_{k_1}A_i)$$

definition $G_s$yields:

$$P(A_j|A_{k_r}|A_{k_{r-1}}|\cdots|A_{k_2}|A_{k_1}|A_i) = p_{jk_r}\cdot p_{k_rk_{r-1}}\cdots p_{k_2k_1}\cdot p_{k_1i}$$

$$= \omega((j,k_r,\cdot),(k_r,k_{r-1},\cdot)\dots(k_2,k_1,\cdot),(k_1,i,\cdot)),$$

where $(j,k_r,\cdot),(k_r,k_{r-1},\cdot)\dots(k_2,k_1,\cdot),(k_1,i,\cdot)$ is a walk in the graph G_s, written via edges of the graph.

As one walk corresponds to only one sequence of events and as the occurrence probability of event A_j starting from event A_i is equal to the sum of probabilities of all sequences of events which in the given system begin with A_i and end with A_j, it follows that the sought probability is equal to the sum of weights of corresponding walks in the graph G_s, and therefore:

$$P(A_jA_i) = \sum_{S(i,j)\in G_s} \omega(S(i,j))$$

which proves the assertion. □

Theorem 2.2. *Let graph G_s be the probability graph which describes event system S. Probability of event sequence$A_{k_1}A_{k_2}A_{k_3}\dots A_{k_n}$, for $A_{k_1}, A_{k_2}, A_{k_3}\dots A_{k_n} \in X \in S$, is equal to the ration of weight of walk $k_1k_2k_3\dots k_n$ and sum of weights of all walks from vertex k_1 to vertex k_n in graph G_s.*

Proof. Theorem directly follows from geometric definition of probability [4, 2]. As the probability of occurrence of some event is equal to the ratio of value of favorable outcomes to the value of all possible outcomes, the probability of some sequence of events is equal to the ratio of value of that sequence of events and value of all sequences of events with the same starting and ending event. In graph G_s value of a sequence is the weight of the walk, and hence the probability is:

$$P(A_{k_1}A_{k_2}A_{k_3}\dots A_{k_n}) = \frac{\omega(k_1k_2k_3\dots k_n)}{\sum\limits_{\mathrm{S(i,j)\in G_s}} \omega(S(i,j))}$$

□

Corollary 2.1. *In case of three vertices, the given formula reduces to Bayes' theorem.*

Theorem 2.3. *Let G_s be the probability graph which describes event system S. Occurrence probability of event A_j starting from event A_i, for $A_j, A_i \in X \in S$, so that between events A_i and A_j events $A_{k_1}, A_{k_2}, A_{k_3}\dots A_{k_n} \in X \in S$ occurred, is equal to the ratio of the sum of weights of all walks in graph G_s which start with vertex i, end with vertex j and contain vertices $k_1, k_2, k_3\dots k_n$ and the sum of weights of all walks in G_s which start with vertex i and end with vertex j.*

Proof. Theorem directly follows from the geometric definition of probability. Occurrence probability of each event is equal to the ratio of value of favorable outcomes to the value of all possible outcomes. The outcome is favorable if events $A_{k_1}, A_{k_2}, A_{k_3}\dots A_{k_n}$ occurred, that is if $k_1, k_2, k_3\dots k_n$ are part of a walk from i to j. The value of favorable events is, therefore, the sum of weights of all walks from i to j which contain the given vertices. Analogously, the value of all events is the sum of weights of all walks from i to j, and therefore the assertion is proven, since the probability of occurrence of event is equal to the ratio of value of favorable to all possible outcomes, that is:

$$P(A_{k_1}A_{k_2}A_{k_3}\dots A_{k_n}) = \frac{\sum\limits_{\mathrm{S(i,j)\in G_s} \wedge \{k_1,k_2\dots k_n\}\in \mathrm{S(i,j)}} \omega(S(i,j))}{\sum\limits_{\mathrm{S(i,j)\in G_s}} \omega(S(i,j))}$$

□

Lemma 2.1. *If probability graph G_s describes a closed event system S, then the sum of all exiting edges for each vertex in G_s is equal to 1.*

Proof. As $\sum_{i=1}^{n} P_{ij} = 1$, for each $j \in 1, 2, 3 \ldots n$ and $P_{ij} = P(A_i A_j)$ then $\sum_{i=1}^{n} \delta(j, i) = 1$, and as the number of vertices in a graph is n all edges with the starting vertex j are in the sum. The sum of exiting weights for each vertex j in graph G_s is equal to 1, and as $j \in 1, 2, 3 \ldots n$, the sum of exiting weights for any vertex of graph G_s is equal to 1. □

Theorem 2.4. *Let G_s be the probability graph which describes the event system S. If G_s is acyclic then the following algorithm finds the most probable sequence of events for all pairs of vertices:*

- *Edges of zero weight are added to graph $G_s$to from a complete graph.*
- *A matrix $W = [w_{ij}]$ is formed containing elements which are the weights of edges in graph G_s, that is, it is the weight matrix of graph G_s.*
- *A predecessor matrix $T = [t_{ij}]$ is formed with the same number of rows and columns as W and initialized:*
$$t_{ij} = \begin{cases} 0, i = j \\ i, i \neq j \end{cases}$$
- *A relaxation is performed on each vertex k in graph G_s over all pairs of vertices i and j from G_s in the following way:*

  If the condition $w_{ij} < w_{ik} \cdot w_{kj}$ is satisfied then:
$$t_{ij} := t_{kj}$$
$$w_{ij} := w_{ik} \cdot w_{kj}$$
- *The resulting matrix W is the probability matrix of the most probable sequences of events (w_{ij} is the probability of the most probable sequence of events starting from i-th and ending with j-th event).*
- *Tuple $D_{ij} = (d_1, d_2, d_3 \ldots d_n)$ is the most probable sequence of events from i-th to j-th event and is obtained using the recursive algorithm:*
$$f(i, j, x) = \begin{cases} d_x := i; i = j \\ d_x := j, f(i, t_{ij}, x + 1); i \neq j \end{cases}$$

with starting parameter $x = 1$, that is, the most probable sequence of events is obtained starting with $f(i, j, 1)$.

Proof. Proof of the theorem (algorithm) reduces to proving analogousness with Floyd-Warshall algorithm [5]. The formed matrix W is analogous to the shortest distance matrix. As graph G_s is a complete graph, there exists no pair (i, j) such that edge $(i, j, p) \in E$, so the initialization of the matrix W is identical to the initialization in Floyd-Warshall algorithm. As there exists no pair (i, j) such that $w_{ij} = \infty$

according to Floyd-Warshall algorithm, the initialization of the predecessor matrix is analogous to the initialization in the Floyd-Warshall algorithm.

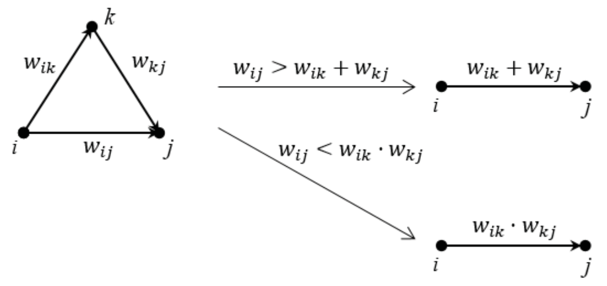

FIG. 2.1: First relaxation is Floyd-Warshall, the second is our algorithm.

In this algorithm (Figure 2.1), relaxation is done when the probability of the sequence of events A_iA_j is less than probability of sequence $A_iA_jA_k$. In order for the given algorithm to be completely analogous with the Floyd-Warshall algorithm, it is necessary to prove the analogy between the relaxation principles of the given algorithm and the Floyd-Warshall algorithm.

As graph G_s is an acyclic graph and as the weight of the path between i-th and j-th vertex is limited by 0 and 1, it is possible to search for a path of maximum weight (the product of numbers in interval [0, 1] is in that interval).

From above it follows that the relaxation condition is correct, and the principle of relaxation is valid, and hence the principle of relaxation is analogous with the principle of relaxation in the Floyd-Warshall algorithm [5], which proves the theorem about the most probable sequences of events in acyclic probability graphs. □

Theorem 2.5. *Let G_s be the probability graph describing event system S. Then the following algorithm finds the most probable sequence of events in the system, with the n-th event as starting event and m-th event as ending event:*

- *A graph G_I is formed, identical to graph G_s, but with a sequence of events corresponding to the vertices of the edge added to the edge, that is for edge (x, y, p) there exists a sequence A_xA_y*
- *Graph G_I is reduced to a graph with two vertices, n and m, such that the most probable sequence of events is added to that event. The reduction is done in the following way:*

  For each vertex k in graph G_I we find all pairs of vertices i and j over which the reduction is done by the pattern in Figure 2.2.

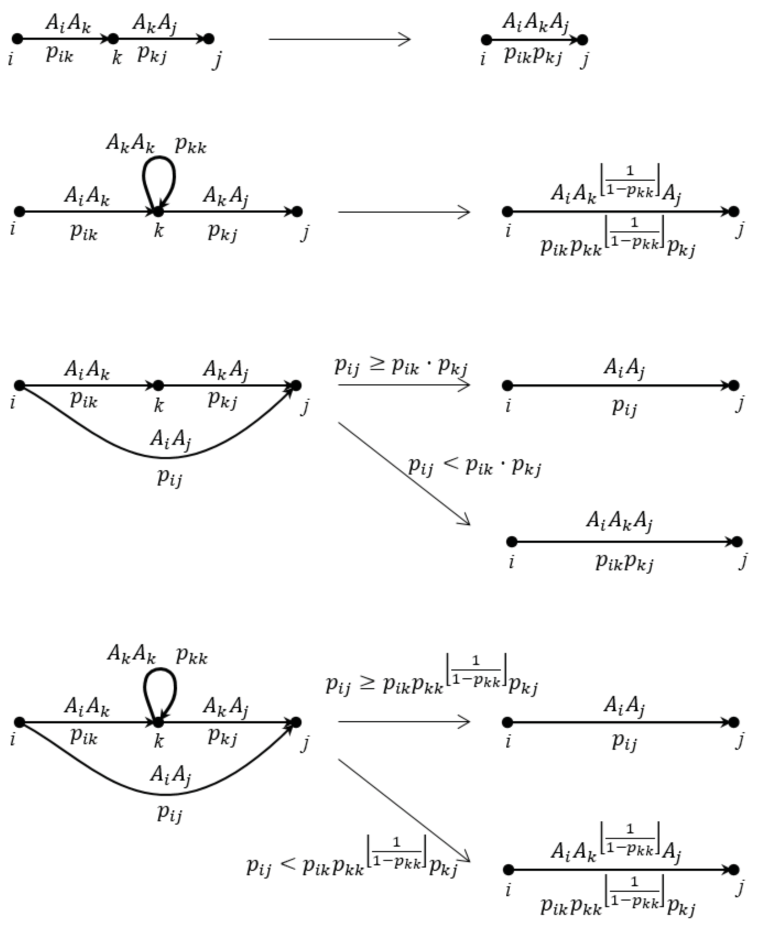
A_iA_k
p_{ik}
A_kA_j
p_{kj}
i
k
j
$A_iA_kA_j$
$p_{ik}p_{kj}$
A_kA_k
p_{kk}
$A_iA_k^{\left\lfloor\frac{1}{1-p_{kk}}\right\rfloor}A_j$
$p_{ik}p_{kk}^{\left\lfloor\frac{1}{1-p_{kk}}\right\rfloor}p_{kj}$
A_iA_j
p_{ij}
$p_{ij} \geq p_{ik} \cdot p_{kj}$
$p_{ij} < p_{ik} \cdot p_{kj}$
$p_{ij} \geq p_{ik}p_{kk}^{\left\lfloor\frac{1}{1-p_{kk}}\right\rfloor}p_{kj}$
$p_{ij} < p_{ik}p_{kk}^{\left\lfloor\frac{1}{1-p_{kk}}\right\rfloor}p_{kj}$

FIG. 2.2: Vertex relaxations

Where A_i, A_j and $A_k$are events or sequences of events. Reduction is done over vertices n and m, since these vertices are the starting and ending vertex.

The order of vertex deletion is given by the pseudo code:

```
for all k in $G_I$ where $k \neq n, k \neq m$ do
    for all i in $G_I$ where $i \neq k, i \neq n, i \neq m$ do
        for all j in $G_I$where $j \neq i, j \neq k, j \neq n, j \neq m$ do
            Reduction(i, k, j);
        end for
    end for
end for
```

- *The resulting graph G_I contains two vertices and one edge. The weight of the edge is the probability of the most probable sequence of events starting with A_i and ending with A_j, and the sequence of events added to the edge is the most probable sequence of events starting with event A_i and ending with event A_j.*

Proof. The proof of the theorem (algorithm) reduces to proving the analogousness with the algorithm for converting pushdown automata into regular expressions. The formed graph G_I is equivalent to the pushdown automaton with one starting and one ending state (vertices n and m from graph G_S respectively).

For PDA-RE algorithm [3] reductions are given in Figure 2.3.

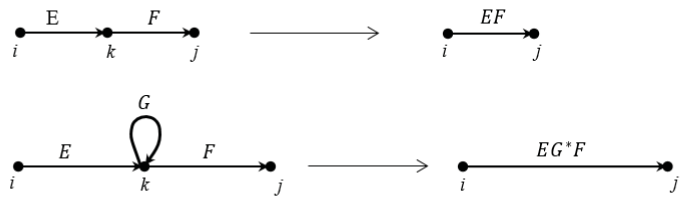

FIG. 2.3: PDA-RE algorithm reductions.

The first is analogous with the given algorithm. In order for the given algorithm to be analogous with the PDA-RE algorithm, it is necessary to solve the problem of repeating the event A_k.

Evidently, event A_k must occur a finite number of times. The most probable number of repetitions of event A_k with probability p_{kk} can be obtained in the following way:

Let X be a random variable which counts the number of repetitions of event A_k with the repetition stopping probability $p = 1 - p_{kk}$, then:

$$X : \begin{pmatrix} 1 & 2 & 3 & \cdots & n \\ p & (1-p)p & (1-p)^2 p & & (1-p)^{n-1}p \end{pmatrix}$$

The most probable number of repetitions of event A_k is the expected value of random value X.

$$\begin{aligned}
E(X) &= \sum_{n=1}^{+\infty} n(1-p)^{n-1}p = p\sum_{n=1}^{+\infty} n(1-p)^{n-1} \\
&= p\left[\sum_{n=1}^{+\infty}(1-p)^{n-1} + \sum_{n=2}^{+\infty}(1-p)^{n-1} + \sum_{n=3}^{+\infty}(1-p)^{n-1} + \cdots\right] = \\
&= p\left[\frac{1}{p} + (1-p)\frac{1}{p} + (1-p)^2\frac{1}{p} + \cdots\right] \\
&= 1 + (1-p) + (1-p)^2 + \cdots \\
&= \lim_{n\to+\infty}\frac{1-(1-p)^n}{1-(1-p)} = \frac{1}{p} = \frac{1}{1-p_{kk}}
\end{aligned}$$

Therefore, the second step of the reduction is analogous to the PDA-RE algorithm.

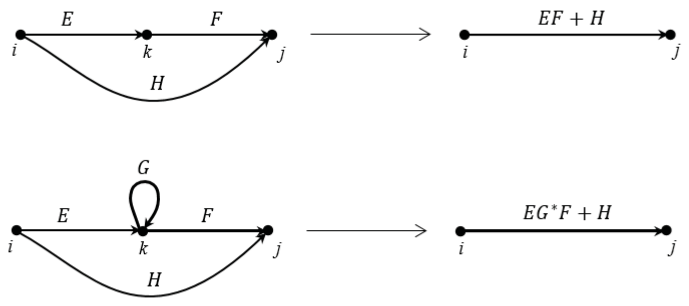

Fig. 2.4: PDA-RE algorithm reductions

As " +" is the " or" operation, in the given algorithm it is necessary to determine whether EF or H is more probable, that is whether $A_iA_kA_j$ or A_iA_j is more probable, which can be determined from the truthfulness of the relation $p_{ij} < p_{ik} \cdot p_{kj}$, so that only one of the given two sequences remains, which is analogous with the PDA-RE algorithm. As the first three steps of reduction are analogous to the PDA-RE algorithm, this step is analogous to the PDA-RE algorithm, since the application of this step reduces to the application of the previous two. These reductions are given in Figure 2.4.

By showing the analogousness of the given algorithm with PDA-RE algorithm, its correctness is proved, thus the proof is done. □

3. Discussion

The application of the given algorithms is not limited only to the systems which are closed. If we observe the subsystem of a larger system for which all parameters are not known, we can conduct analysis and modeling on the basis of the known parameters. The closure property guarantees that the obtained solution will be correct in the given system (that is, the solution will be correct for the given system/mathematical model). If we observe some isolated part of the system, then the closure property is not relevant, and the solution of such system (its analysis) will yield the result according to the probability theory. In such a case, it is possible that the sum of entering paths in some vertex is not 1, which is still correct considering the probability theory, since the sum excludes the cases not encapsulated by the observed subsystem of a larger system. Analogously, the same is true for the sum of weights of exiting edges from a vertex in a chosen subsystem.

A modified Floyd-Warshall algorithm (theorem 2.4) will, in case of an incomplete system (a system without the closure property) yield a correct result for the given derived model (model represented by the given subsystem), in case that the probability graph is acyclic.

In the case of a cyclic graph, the application of theorem 2.4 is limited only to the specific cases. Generally, if a probability graph contains a cycle then the solution will be correct only in case the number of repetitions in the cycle, for each walk containing the cycle, is equal to zero or one (that is, the probability of multiple repetition of a sequence of events from the cycle is less than the probability of its one or no repetition). In the case of a set of successive events (event system for which the probability graph does not contain a cycle) the algorithm will yield a correct result with the execution time of $O(n^3)$ which is equivalent to the execution time of Floyd-Warshall algorithm.

Unlike theorem 2.4, theorem 2.5 is applicable to cyclic graphs. The execution time (time complexity) of this algorithm should theoretically be $O(n^4)$, with the assumption that the reduction time is $O(n)$. As for the mathematical correctness, the order of reduction is irrelevant, since all vertices are reduced the same way, and hence the obtained result will be the same, regardless of the order of reduction. From the aspect of practical implementation of this algorithm, one of the possible implementations is representing the graph as a set of vertices and edges (according to the mathematical definition), where vertices (or, possibly, edges) are indexed for immediate access. This way the algorithm execution time can be reduced. In the index structure, vertices can be sorted by their internal number or number of edges that connect them to other vertices. Depending on the implementation of the reduction itself, execution time of the algorithm can vary. If the implementation is as above, then the reduction could be executed in $O(1)$, since the indexing would enable passing the values by reference, and hence the execution time of the algorithm would be $O(n^3)$.

In addition to mentioned theorems (algorithms), some of algorithms applicable to standard graphs are applicable to probability graphs. For example, it is possible to use breadth and depth search on probability graphs in order to find relations between events. By using the breadth-first search one can obtain the least number of events over which two events from the observed system are connected. Also, one can obtain all possible sequences of events from a system using breadth-first and depth-first search. Breadth-first and depth-first search algorithms do not require modifications in terms of functionality in order to be applied to probability graphs. Algorithms for finding the minimum spanning tree and flow maximization would require more significant modifications in order to be applicable to probability graphs. Depending on the problem modeled using a probability graph, the application of specific algorithms on such graph can change.

Probability graphs can be applied to some problems in which there are an infinite number of events or graph vertices. Problem modeled with a graph with infinite vertices could be analyzed or solved using graph properties (using the properties of probability graphs). For analysis of probability graphs with infinite number of vertices the closure property is useful (if the modeled system or some part of it has that property) which guarantees that the sum of weights of exiting edges for each vertex will be equal to 1.

3.1. Graphs, Markov Chains and Automata

In graph theory, a structure containing mutually connected nodes (vertices) is defined as a graph. Graph is a general structure, which has many specific variants (e.g. directed and undirected graph; a tree is a directed graph such that for each node except for the root node exists only one other node connected to it). From the principle of polymorphism it can be derived that a Markov chain is, in fact, a variant of graph. Same can apply for a finite automaton.

Given two variants of a graph, principles of generalization and association allow some methods of analysis from one of the variants to be applied to the other and to the general graph structure. Generalization also implies that methods applicable to all graphs are applicable to specific variants of the same. Hence it can be derived that, if both Markov chains and finite automata are graphs, analysis methods applied to general graphs can be applied to both Markov chains and finite automata, and, more importantly, that there exist analysis methods applicable to finite automata which can be applied to Markov chains.

As for the visual representation of Markov chains, it may not be of much significance whether they are represented as graphs or state diagrams. However, it might be a good practice to represent Markov chains according to the context they are used in. As graph representation is suitable for traversing nodes, this approach might be more suitable in traversing events in a Markov chain.

Furthermore, visual representation can vary depending on the context of the problem. For example, for a problem with an infinite number of states some meta-graph structure might be more suitable. The idea is to use a specific principle for a specific problem, that is, to apply the most adequate variant of a graph to a problem. Methods from specific graph variants could also be applied to other graph variants. Graph structures are also suitable and used for computer analysis of large datasets in certain mathematical problems.

4. Examples

Example 4.1. A fighter and bomber are in an aerial combat. First, the fighter fires at the bomber with hit probability of 0.8. If he misses, then bomber returns fire with hit probability of 0.3.

1. If the bomber misses, then fighter fires again with hit probability of 0.5.
2. If the bomber misses, then the cycle continues until either bomber or fighter is hit.

What is the probability of bomber going down and what is the probability of the fighter going down?

Solution:

1. Let A be the starting event, C the event of fighter missing the first shot, and D the event of bomber missing the first shot (that is, the state where the fighter shoots the second time). Events L and B are events of fighter going down and bomber going down, respectively (Figure 4.1).

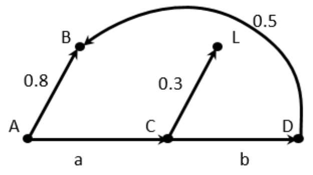

FIG. 4.1: Probability graph

From the closure property for vertices (events A and C) weights of edges a and b can be obtained, which are thus: $a = 0.2$ and $b = 0.7$. As the probability of some event is equal to the sum of weights of all walks leading from the starting event to that event, it is $P(B) = 0.8 + a \cdot b \cdot 0.5 = 0.8 + 0.2 \cdot 0.7 \cdot 0.5 = 0.87$ and $P(L) = a \cdot 0.3 = 0.2 \cdot 0.3 = 0.06$. The reason for the closure property not being applied to vertex D is that the probability of failure of hitting either plane is not required and hence it is not necessary to model that part of the problem.

2. The given situation can be modeled with the probability graph given in Figure 4.2.

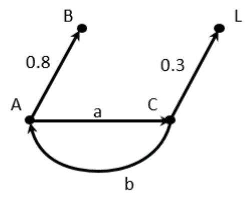

FIG. 4.2: Probability graph

In this case it is necessary to sum the weights of all walks from vertex A to the corresponding vertex (B or L). Weights of edges a and b are the same as in the first case due to the closure of the system. From theorem 1, probabilities of events B and L are: $P(B) = 0.8 \cdot \sum_{i=0}^{+\infty} (a \cdot b)^i$ and $P(L) = a \cdot 0.3 \cdot \sum_{i=0}^{+\infty} (b \cdot a)^i$, that is

$$P(B) = 0.8 \cdot \frac{1}{1 - a \cdot b} = 0.93$$

and

$$P(L) = a \cdot 0.3 \cdot \frac{1}{1 - b \cdot a} = 0.07.$$

As only two outcomes are possible it is $P(B) + P(L) = 1$.

Example 4.2. Meteorological events are represented with $A - G$ (A is rain, B weak rain, C fog, D cloudy, E mildly cloudy, F somewhat cloudy and G sunny). Their mutual dependence is given with a graph in Figure 4.3. Find the most probable sequence of meteorological events and its probability so that the starting event is rain and the ending event is sunny weather.

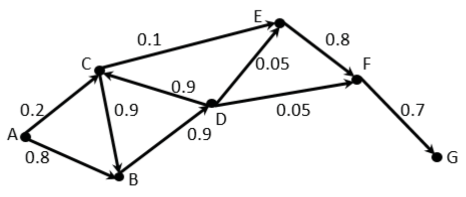

FIG. 4.3: Dependency graph

Solution: Reduction is done according to the theorem 5.

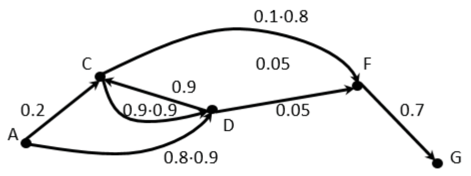

FIG. 4.4: Reduction step 1

After eliminating vertices B and E from the graph, vertex C is eliminated, while the added sequences of events are maintained for each edge (Figure 4.4).

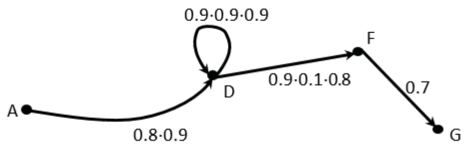

FIG. 4.5: Reduction step 2

Elimination of vertex D will cause repetition of event D three times, according to the formula in theorem 5, and the probability added to the edge will be $(0.9\cdot 0.9\cdot 0.9)^3$ (Figure 4.5).

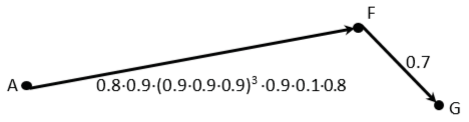

FIG. 4.6: Reduction step 3

Further reduction will reduce the graph to two vertices and one edge, so that the values added to the edge will be the solution, that is $P_{\max}(AG) = 0.0141 = 1.41\%$, with the sequence of events ABDCBDCBDCBDCEFG (Figure 4.6).

Example 4.3. Let $X_1, X_2, \ldots$ be the sequence of independent random variables such that

$$X_k : \begin{pmatrix} -1 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix},\ k = 1, 2, \ldots$$

and

$$X = \sum_{k=1}^{+\infty} \frac{X_k}{2^k}$$

Determine the distribution of the variable X.

Solution: If a graph which describes all values that random variable X can have is modeled, we get a tree with one starting vertex (node) and infinite number of leafs as given in Figure 4.7. On each level of the graph, probability of occurrence for each event of that level is equal, since the number of different walks from root to an individual vertex on a given level is equal to 1 (or, possibly, 2 on level N_∞), and weights of these walks are also equal. It follows that the distribution is uniform on some interval. The boundaries of that interval are the values of vertices of the graph in case that every X_k has value -1 (or 1, for the upper boundary of the interval), and hence the sought interval is:

$$\left(-\sum_{k=1}^{+\infty} \frac{1}{2^k}, \sum_{k=1}^{+\infty} \frac{1}{2^k}\right) = (-1, 1)$$

Therefore, variable X has uniform distribution on the interval $(-1, 1)$.

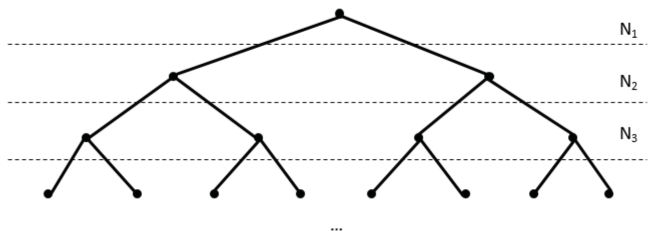

Fig. 4.7: Infinite probability graph

5. Conclusion

From the above analysis, it is evident that graph theory can be efficiently utilized as a probabilistic and/or combinatorial method for analyzing event systems.

An algorithmic approach to analyzing event sequences and systems was demonstrated. Evidently, the algorithms and data structures used in graph theory are applicable to discrete event systems. Several adaptations of classic graph algorithms were demonstrated. It is likely that other graph algorithms can be adapted for use in analysis of these systems.

Few common probabilistic problems were solved using the probability graphs approach.

Probability graphs are another utility for solving complex probabilistic problems and computer analysis of large event systems, as demonstrated. Since graph theory is a proven tool in computer science, probability graphs offer the diversity of this tools' application in the probabilistic analysis.

REFERENCES

1. D. A. Levin, Y. Peres and E. L. Wilmer: *Markov Chains and Mixing Times*. American Mathematical Society, 2009.
2. N. Elazović: *Theory of probability (in Croatian)*. Zagreb, 1995.
3. J. E. Hopcroft and J. D. Ullman: *Introduction to Automata Theory, Languages, and Computation (2nd Edition)*. Addison-Wesley, 2001.
4. W. Feller: *An Introduction to Probability Theory and Its Applications, Vol. 1*. Wiley, 1968.
5. M. Tomašević: *Algorithms and Data Structures (in Serbian)*. Akademska misao, 2008.
6. M. V. Ćelić and B. Sukara-Ćelić: *Linear algebra (in Serbian)*. Banja Luka: Glas Srpski Grafika, 2010.
7. D. Cvetković and S. Simić: *Discreete mathematics – mathematics for computer sciences (in Serbian)*. Naučna knjiga, 1990/1997.
8. P. Hotomski and D. Malbaški: *Mathematical logic and the principles of programming (in Serbian)*. TF Zrenjanin, 2003.
9. E. Parzen: *Stochastic Processes*. Holden-Day, 1962.
10. M. Sipser: *Introduction to the Theory of Computation*. PWS Publishing Company, 1997.

Igor Ševo
Faculty of Electrical Engineering
Department of Informatics and Computer Science
78000 Banja Luka, Republika Srpska, Bosnia and Herzegovina
igor@igorsevo.com