UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

Evaluation eines Algorithmus zur Absicherung von Datenbanken mit Propabilistischen

Evaluation of an Algorithm for Securing Databases from Propabilsitic Inference

Bachelorarbeit

verfasst am
Institut für Informationssysteme

im Rahmen des Studiengangs
IT-Sicherheit
der Universität zu Lübeck

vorgelegt von
Kevin Schmelzer

ausgegeben und betreut von
Prof. Dr. Ralf Möller

mit Unterstützung von
Simon Schiff

Lübeck, den 18. August 2020

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Kevin Schmelzer

Zusammenfassung

Datenbanken mit sensiblen Daten sollten einen hohen Sicherheitsstandard erfüllen. Trotzdem ist das einrichten von Sicherheitsvorkehrungen komplex und erfordert spezielle Fachkräfte, die sich mit den Systemen auskennen müssen. Diese Problematik haben viele Forschungseinrichtungen und auch Unternehmen, weil es viel zu Beachten gibt. Eins davon ist die Inferenzkontrolle für aggregierte Anfragen, wodurch ein Angreifer durch statistische Auswertungen und klug gewählte Anfragen an sensible Daten kommen kann.
Wir werden hierbei den Prototypen Angerona evaluieren, der Informationslücken verhindert die von probabilistischen Abhängigkeiten resultieren. Es wird gezeigt, wie man Angerona mit echten Patientendaten initalisiert und welche Vor-und Nachteile dieser Algorithmus aufweist. Zum Schluss generieren wir mit Synthea Patienten und testen die Performance mit verschiedenen Eingaben.

Abstract

Englische Abstract

Inhaltsverzeichnis

1

Einleitung

Der Schutz von sensiblen Daten ist für viele Unternehmen und andere Einrichtungen wichtig, die persönliche Daten, wie Herkunft, Religion, Alter usw., erfassen, verarbeiten und speichern. Die Relevanz des Themas Privatsphäre und Digitalisierung ist spätestens seit der Wirksamkeit der Europäischen Datenschutz-Grundverordnung (DSGVO) [10] im Mai 2018 bemerkbar, da die dadurch entstandene Herausforderungen auf Unternehmensseite Sie vor rechtliche und insbesondere auch technische Probleme stellt, weshalb DSGVO-konform software-Lösungen entwickelt wurden um die Arbeit zu erleichtern [12].
Ein besonderer Fokus wird auf die Privatsphäre in Krankenhäusern gelegt, da in Krankenhäusern das Thema Privatsphäre schon immer ein sehr sensibles Thema gewesen ist. Dabei gab es schon vor der Einführung der DSGVO eine andere Regelung zum Schutz der Privatsphäre speziell im medizinischen Bereich und zwar die Health Insurance Portability and Accountability (HIPAA) aus dem Jahr 1996 aus den USA. Hierbei wird für jeden der Gesundheitsdaten verarbeitet oder speichert, durch Anordnungen und Regelungen zur Verarbeitung von Daten vorgeschrieben, die Privatsphäre der Patienten durch z.B. Zugangskontrolle, Anonymisierung, richtige Datenspeicherung usw. zu schützen [1]. Dabei kam es damals schon zum Konflikt zwischen dem Schutz der Privatsphäre und zur Verwendung oder Veröffentlichung von Gesundheitsinformation um wichtige soziale Ziele, wie zum Beispiel Forschungszwecke, zu erfüllen [17]. Der bisherige Ansatz sensible Daten zu schützen war es den Zugang in Teilen einfach zu verbieten und die Daten zu anonymisieren [15]. Dieser Ansatz ist zwar für die Sicherheit optimal, jedoch leidet darunter die Usability, da der Zugang zu den Daten für beispielsweise Forschungszwecke schwierig ist bzw. bei zu starker Anonymisierung unbrauchbar werden.
Die Gewährleistung der Vertraulichkeit von sensiblen Daten erfordert einen Schutz vor direktem und indirektem Zugriff auf eine Datenbank. Der direkte Zugriff beschreibt dabei den Zugang zu Daten aus einer Datenbank mithilfe von einfachen Anfragen an die Datenbank. Beim indirekten Zugriff hingegen, wird versucht durch statistische Auswertungen von externen Informationen und klug gewählte Anfragen an die Datenbank, an sensible Daten zu kommen. Daher kam es zu einigen neuen Ansätzen, die auch den indirekten Zugang schützen sollen und eine davon ist die Database Inference Control (DBIC). Damit DBIC auch effektiv den indirekten Zugang verhindert, muss dieser eine große Menge von probabilistischen Abhängigkeiten abdecken können, um viele verschiedene

Angreifermodelle darzustellen und es muss eine angemessene Laufzeit aufweisen, um diese auch auf reale und große Datenbanken anwenden zu können. Die Laufzeit wird dabei in *Online* und *Offline* Zeiten unterteilt, wobei die *Online* Zeit das Intervall zwischen dem Start der Anfrage und der Antwort ist und die *Offline* Zeit vom Start des Systems bis das System bereit für eine Eingabe ist.
Der DBIC Mechanismus der in diesem Paper evaluiert wird ist Angerona, wovon erstmals nur ein Prototyp existiert. Der unterschied zu bisherige DBIC Mechanismen zu Angerona ist, dass die bisherigen nur präzise Datenabhängigkeiten oder nur eine begrenzte Anzahl von probabilistischen Abhängigkeiten erlaubt haben und somit nicht für Reale Datenbanken tauglich waren. Angerona hingegen soll auch bei komplexen probabilistischen Abhängigkeiten eine angemessene Laufzeit haben und somit in der Praxis nutzbar sein.
Um das Datenbankmodell mit den probabilistischen Abhängigkeiten nach Angerona zu übertragen, werden zuerst alle Abhängigkeiten in ein Bayes Netz modelliert. Anschließend wird aus dem Bayes Netz ein Angreifermodell erstellt, in dem das Vorwissen von jedem Benutzer gespeichert wird. Mithilfe von Sicherheitsregeln wird dann ein Schwellwert definiert, der den Zugang zum Datenbanksystem nur zulässt, wenn das Vorwissen des Angreifers unter dem Schwellwert liegt.
gehört der letzte Satz überhaupt hier rein??]

1.1 Beiträge dieser Arbeit

In dieser Arbeit wird der Prototyp von Angerona, einem Prototypen für ein DBIC Mechanismus von Marco Guarnieri auf echten, pseudonomisierten Patientendatenbanken evaluiert. Dabei wird im ersten Schritt die korrekte Einrichtung von Angerona vorgestellt und anschließend mit realistischen Beispielen bewerten. Zum Schluss wird die Online und Offline Laufzeit bei der Nutzung verschiedener Patientendatenbanken gemessen.

1.2 Verwandte Arbeiten

Es existieren bereits einige Ansätze, die auch versucht haben DBIC-Mechanismen zu implementieren. Einige Ansätze davon sind :
Ansätze suchen

.

1.3 Aufbau dieser Arbeit

In dieser Arbeit wird Angerona vorgestellt gefolgt von einigen Beispielen, die den Zweck von Angerona näher bringen. Anschließend folgen Initialisierungen und Auswertungen auf echte, pseudonymisierte Datenbanken wie MIMIC III, eICU und Synthea. Zum Schluss werden die Online und Offline Laufzeiten auf diesen Datenbanken analysiert und ausgewertet.

2

Grundlagen von Angerona

2.1 Grundlegendes

Systemmodell

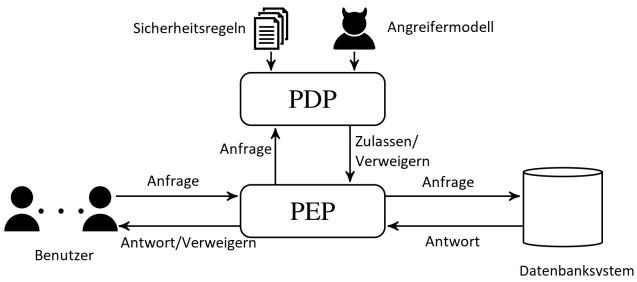


Abbildung 2.1: Systemmodell

Die Abbildung 2.1 zeigt das von Angerona genutzte Systemmodell. Dabei interagiert der Beutzer mit dem Inferenzkontrollsystem, welches aus den zwei Komponenten Policy Decision Point(PDP) und dem Policy Enforcement Point (PEP) besteht. Das Inferenzkontrollsystem entscheidet Anhand von vordefinierten Sicherheitsregeln und einem Angreifermodell, ob die Anfrage des Benutzers an das Datenbanksystem übergeben wird. Dabei wird davon ausgegangen, dass alle Anfragen und Antworten aus dem Systemmodell über sichere Kommunikationskanäle laufen. Außerdem gilt als Voraussetzung, dass jeder Benutzer das Datenbankschema und die Sicherheitsregeln kennt.

Datenbanksytem Das Datenbanksystem lagert alle Daten und gibt diese ausschließlich dem Inferenzkontrollsystem heraus. Dadurch kann ein Benutzer keine direkte Anfrage an das Datenbanksystem stellen.
Benutzer Jeder Benutzer hat ein eigenes Konto um Informationen mithilfe von SELECT Anfragen an das Inferenzkontrollsystem zu erhalten. Dabei hat jeder Benutzer nur Leserechte und kann somit die Datenbank nicht verändern. Jede Anfrage wird vom Inferenzkontrollsystem geprüft und wird nur ausgeführt, wenn diese von den Sicherheitsregeln

autorisiert wird.
Sicherheitsregeln Die Sicherheitsregeln bestehen aus einer Menge von Regeln, die definieren welche Informationen geheim gehalten werden sollen. Diese Regeln definieren die Erwartungen jedes Benutzers über den Inhalt der Datenbank als Wahrscheinlichkeitsverteilung. Die Regeln werden formalisiert durch ein Kommando in der Form `SECRET` q `FOR` u `THRESHOLD 1`, wobei q die Anfrage, u den Benutzer und l den Grenzwert darstellt, bei der die Anfrage genehmigt werden darf. Dabei gilt $0 \leq l \leq 1$. Eine Sicherheitsregel „Der Benutzer u ist nicht autorisiert die Antwort von der Anfrage q zu erfahren" wird ausgedrückt mit `SECRET` q `FOR` u `THRESHOLD 1`. Außerdem kann die Regel „Für alle Benutzer $u \notin \{u1, \ldots, u_n\}$, muss die Erwartung von u bei der Anfrage q kleiner als l sein" mit dem Kommando `SECRET` q `FOR USERS NOT IN` $\{u_1, \ldots, u_n\}$ `THRESHOLD 1`
Angreifer Ein potentieller Angreifer ist jeder Benutzer des Datenbanksystem mit einem Benutzerkonto. Das Ziel eines Angreifers ist es die Sicherheitsregeln zu verletzen indem er mindestens auf ein `SECRET` q schließen kann mit einer Wahrscheinlichkeit unter dem dazugehörigen `THRESHOLD` l.
Der Angreifer interagiert mit dem Inferenzkontrollsystem, indem er Anfragen wie ein Benutzer stellt. Somit kann der Angreifer neue Informationen aus den Antworten erhalten und Daten die in Beziehungen stehen feststellen. Ein Beispiel für eine Beziehung zwischen Daten ist zum Beispiel, dass wenn ein Patient raucht, dass die Wahrscheinlichkeit für Krebs für Ihn erhöht ist.
Datenbankzustand Der Datenbankzustand beschreibt die Belegung der Datenelemente in der Datenbank.
Angreifermodell Das Angreifermodell repräsentiert das Vorwissen des Benutzers über den aktuellen Datenbankzustand und wie dieses sich verändert, wenn der Benutzer mit der Datenbank interagiert. Diese Erwartung kann das Wissen des Angreifers über die Beziehung zwischen den Datenelementen oder Vorwissen widerspiegeln.
Inferenzkontrollsystem Das Inferenzkontrollsystem schützt die Vertraulichkeit der Daten in der Datenbank. Es besteht aus dem PEP und dem PDP und wird mit den Sicherheitsregeln P und dem Angreifermodell ATK konfiguriert. Für jeden Benutzer behält das Inferenzkontrollsystem die Erwartung gemäß dem Angreifermodell im Überblick.
Das System fängt alle Anfragen c vom Benutzer u ab und entscheidet dann, ob u autorisiert ist c auszuführen. Wenn c die Sicherheitsregeln erfüllt, dann wird c an das Datenbanksystem weitergeleitet, dass dann c ausführt und die Antwort an u zurückgibt. Andernfalls wird eine `security exception` ausgelöst und c wird verweigert [2].

Bayes-Netze

Für jede Implementierung, die in Angerona vorgenommen wird, sollte ein dazugehöriges Bayes-Netz modelliert werden, damit die Abhängigkeiten übersichtlich dargestellt werden. Denn ohne Bayes-Netz wird es fast unmöglich bei komplexeren Abhängigkeiten überhaupt ein Angreifermodell zu deklarieren.
Ein Bayes-Netz ist ein direkter, azyklischer Graph in dem jeder Knoten die gemeinsamen Wahrscheinlichkeitsverteilungen jeder Zufallsvariablen enthält. Dabei ist jeder Knoten selbst eine Zufallsvariable mit einer Wahrscheinlichkeitsverteilung $P(V_i \mid Parent(V_i))$. Die Elternbeziehung eines Knoten gibt dabei an, dass $Parent(V_i)$ einen direkten Einfluss

auf V_i hat und wird mit einem gerichteten Pfeil dargestellt[18]. Ein Beispiel für solch ein Bayes-Netz ist in Abbildung 2.2 zu sehen.

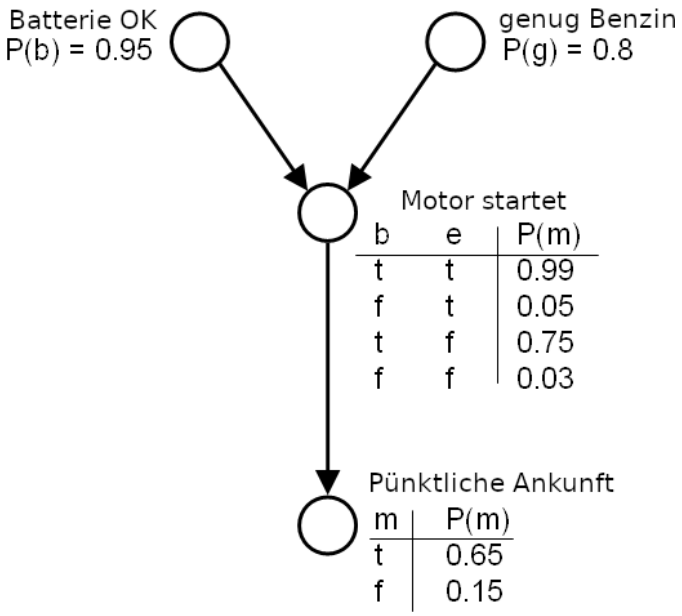


Abbildung 2.2: In diesem Bayes-Netz ist der Start des Motors davon abhängig, ob der Motor OK ist und ob genug Benzin vorhanden ist. Die Abhängigkeit kann formal beschrieben werden durch $P(m \mid b, e)$. Die Wahrscheinlichkeit, dass der Motor startet ist dabei 0.99, wenn genug Benzin und die Batterie in Ordnung ist. Die Wahrscheinlichkeit, dass der Motor startet, obwohl nicht genug Benzin vorhanden ist und die Batterie nicht in Ordnung ist beträgt 0.03.

[11]

Zitat richtig in Bild einfügen. 11 gehört zum Bild , Nach bildern ist nächste Zeile eingerückt

Eine zusätzliche Vorrassetzung für die Bayes-Netze in Angerona ist, dass diese Polytree sein müssen, da Angerona nur azyklische Modellierungen akzeptiert.

Definition 2.1 (Polytree). Ein Polytree ist ein direkter, azyklischer Graph, der azyklisch bleibt, selbst wenn alle direkten Kanten durch indirekte Kanten ersetzt werden.

Problog

Problog ist eine probabilistische Erweiterung von Prolog (PROgramming in LOGic), wobei Prolog eine logische Programmiersprache ist, die aus verschiedenen Klauseln k_i und aus logischen Verknüpfungen ein Ergebnis berechnet. Problog erweitert jedes k_i mit einer Wahrscheinlichkeit p_i, wodurch Problog Wahrscheinlichkeiten ausgibt, mit welcher die k_i eintreten, wohingegen Prolog nur ein true oder false nach logischen Berechnungen liefern kann.

Programmtext 2.1: Beispiel Problog Programm

1 0.1::*erdbeben*.
2 0.9::*alarm_bei_erdbeben*.
3 *alarm* :– *erdbeben*, *alarm_bei_erdbeben*.

Ein Problog Programm $T = p_1 :: k_1, \ldots, p_n :: k_n$ gibt somit eine probabilistische Verteilung über die einzelnen Klauseln und diese können anschließend zu einer beliebigen Logischen Verknüpfungen aus $K = k_1, \ldots, k_n$ verknüpft werden. In Beispiel 1 sieht man, dass der Fakt *erdbeben* mit Wahrscheinlichkeit $P(erdbeben) = 0.1$ wahr ist und 0.9 falsch ist. Andersrum für den Fakt alarm_bei_erdbeben ist die Wahrscheinlichkeit 0.9, dass der Alarm auslöst und eine Gegenwahrscheinlichkeit von 0.1, dass dieser nicht auslöst. Diese Aussagen aus Zeile 1 und 2 nennt man auch probabilistischer Fakt.
In Beispiel 1 ist somit für die Klausel $k_{alarm} = 0.9 \cdot 0.1 = 0.09$. [16][6]
Um das Angreifermodell zu initialisieren benötigen wir nur das $\wedge$ als logische Verknüpfung , dass in Problog mit einem „,“ dargestellt wird. Normalerweise wird eine Negation mit „\+“ gekennzeichnet, in Angerona jedoch mit „*NOT*“.

Noch ein Konstrukt, dass von Problog geliefert wird, ist die *annotated disjunction*, die es möglich macht Klauseln zu definieren, die mehr als nur zwei Werte annehmen können. Ein Beispiel dafür wäre ein Alarm, der die Werte $P(A_1) = 0.2$ (an), $P(A_2) = 0.7$ (aus) oder $P(A_3) = 0.1$ (defekt) annehmen kann. Dies würde man mithilfe der annotated disjunction folgendermaßen formulieren :

2/10::*alarm*(*X*, 1); 7/10::*alarm*(*X*, 2); 1/10::*alarm*(*X*, 3).

Angerona

Angerona ist ein DBIC (Database Inference Control) Mechanismus der Datenbanken gegen probabilistische Inferenzen absichert. Hierbei wird nur ein Prototyp von Angerona betrachtet, da es noch keine vollendete Version gibt. Der Prototyp unterstützt hierbei nur boolesche Anfragen. Es wurde zwar eine Lösung für nicht-boolesche Werte im Paper vorgestellt, jedoch wurde diese nicht im Prototypen implementiert.
Zusätzlich führt Angerona eine **Historie** $h = (u, q, a, d)$, die für jeden Benutzer die alle bereits getätigten Anfragen speichert, die an das Inferenzkontrollsystem gestellt wurden. Jeder Eintrag in der Historie speichert den Benutzer u, die Anfrage q, die Antwort aus dem Datenbanksystem a und die Entscheidung d, ob die Anfrage genehmigt wurde.
Als input benötigt Angerona ein Datenbankzustand s, eine Historie h, die Anfrage q vom Benutzer u, die Systemkonfiguration c und das Angreifermodell ATK. Die **Systemkonfiguration** definieren dabei das Datenbankschema und die Integritätsbedingungen.
Angerona prüft dabei für jede Anfrage q, ob diese eine vorher definierte Sicherheitsregel $s \in S$ verletzt. Dafür wird zuerst in der Historie h nachgesehen, ob s bereits vorher verletzt wurde. Wenn dies nicht der Fall ist, dann wird geprüft, ob nach der Antwort von s eine andere Sicherheitsregel $s' \in S$ verletzt wird. Dies wird sichergestellt, indem das Vorwissen vom Benutzer u von s' und s unter dem definierten Schwellwert liegen muss, nachdem die Anfrage s ausgeführt wurde. Damit verhindert Angerona, dass nach der Ausführung von q nicht weitere s' verletzt werden .

Geprüft wird dies, indem Angerona zuerst prüft, ob es überhaupt möglich ist, dass die Anfrage q in einem beliebigen Datenbankzustand zutreffen könnte. Wenn dies der Fall ist, dann wird die Historie h erweitert mit der Anfrage q und dem s und prüft dann anschließend ob mit der erweiterten Historie einen Datenbankzustand gibt, indem q nicht zugelassen wird [2].

Wie genau soll Angerona beschrieben werden? Vlt. noch Algorithmus hinzufügen, aber der aus dem Paper ist gefühlt zu komplex

2.2 Praktisches Setup

Der Prototyp von Angerona lässt sich auf der Seite von Marco Guarnieri [2] herunterladen. Angerona lässt sich in zwei verschiedene Modis starten :

1. **Experiment**: Hiermit können vorgefertigte Beispiele aus dem Paper „Securing Databases from Probabilistic Inference“[2] reproduziert werden. Hierbei werden keine größeren Initialiserungsschritte benötigt und das Programm übernimmt die Generierung vom beliefProgram und die dazugehörige Datenbank. Anschließend werden zufällig generierte Anfragen an die Datenbank gestellt und von Angerona geprüft. Als output erhält man die Zeitmessung für die Ausführung der Anfrage, die Ausführungszeit von Angerona und die gesamte Zeit in einer CSV Datei.
2. **Manual**: Erlaubt es mit einer Datenbank zu interagieren, die durch Angerona geschützt ist. Dieser Modus erfordert drei Dateien als input und zwar das BeliefProgram, initStatements und das template. Diese drei Dateien werden im folgenden weiter erläutert.

Im folgenden wird ausschließlich der Manual mode verwendet.

BeliefProgram

Das beliefProgram.pbl beschreibt das Angreifermodell. Hier werden die Erwartungen des Angreifers mithilfe von Problog beschrieben.
In die ersten Zeilen werden dabei alle Knoten $\{v_0, \dots, v_n\} \in V$ aus dem Bayes-Netz aufgeschrieben, die keine Abhängigkeit haben, in der Form,

---

$np_0.name(X) : -p\_np_0.name(X).$
...
$np_n.name(X) : -p\_np_n.name(X).$

---

wobei $\{np_i \in V \mid Parent(np_i) = \emptyset\}$ und das Attribut name gibt den vorher definierten Namen für den Knoten aus. Anschließend werden die Wahrscheinlichkeit für die Klausel $p\_np_0.name$ als probabilistischen Fakt für jede Möglichkeit $p \in \mathbb{N}$ in der Datenbank hinzugefügt in der Form :

$P(np_0) :: p\_np_0.name(p_0).$
...
$P(np_n) :: p\_np_n.name(p_0).$
...

$P(np_0) :: p\_np_0.name(p_n).$
...
$P(np_n) :: p\_np_n.name(p_n).$

Für die Knoten mit Abhängigkeiten, also für alle Knoten V für die gilt $H = \{v \in V \mid Parent(v) \geq 1\}$ werden die Klauseln für jede möglich Welt von den Abhängigkeiten eingefügt und verknüpft mit einer Konjunktion, die in Problog mit einem "," dargestellt wird. Die Negation wird nicht wie üblich in Problog mit einem \+, sondern mit einem „NOT“ dargestellt. Anschließend wird jeder Ausdruckt mit einer selbst definierten Variable versehen und mit diesem konjugiert, die am Ende durch einen probabilistischen Fakt die dazugehörige Wahrscheinlichkeit für diese Welt zugeordnet wird.

Ist das Verständlich ?

Programmtext 2.2: Beispiel mit 2 Variablen als Abhängigkei

```
1  H(X) :- v_0.name(X), v_1.name(X), variable11(X).H(X) :- v_0.name(X) , NOT v_1.name(X),
       variable10(X).
2  H(X) :- NOTv_0.name(X), v_1.name(X), variable01(X).H(X) :- NOT v_0.name(X) , NOT
       v_1.name(X), variable00(X).
3
4  P(h_0) :: variable11(p_0)
5  P(h_1) :: variable10(p_0)
6  P(h_2) :: variable01(p_0)
7  P(h_n) :: variable00(p_0)
8  ...
9  P(h_0) :: variable11(p_n)
10 P(h_1) :: variable10(p_n)
11 P(h_2) :: variable01(p_n)
12 P(h_n) :: variable00(p_n)
```

Eine mehrwertige Variable kann anstatt zwei Wahrheitswerten true und false auch mehr Werte annehmen. Um diese im BeliefProgram auszudrücken, wird der mehrwertigen Variable in eine *annotated disjunctiont* eingeteilt. Jedoch wird dann das „;“ entfernt und für jeden Wert den die Variable annehmen kann eine neue Klausel erstellt mit einer neuen Variable die im folgenden $sw_1, \ldots sw_n$ genannt wird und die dazugehörige Wahrscheinlichkeit mit $p_1, \ldots, p_n$ wird zum Schluss für jeder dieser neu erstellen Variable ein probabilistischen Fakt hinzugefügt mit der dazugehörigen Wahrscheinlichkeit mit der der Wert zutrifft. Der Name der mehrwertigen Variable wird im folgenden als A definieret und die Tupel, die durch die *annotated disjunction* übergeben werden als $t_1, \ldots, t_n$.

Programmtext 2.3: Vorgehensweise bei *annotated disjunction*

```
1  A(t_1) :- sw_1(t_1)
2  ...
3  A(t_n) :- NOT sw_1(t_1), ... , NOT sw_{n-1}(t_{n-1}), sw_n(t_n)
4
```

5 $p_1 :: sw_1()$
6 ...
7 $p_n :: sw_n()$

InitStatements

Die InitStatements.txt definiert das Datenbankschema und füllt die Datenbank mit Daten. Außerdem werden hier die Sicherheitsregeln definiert, wie hoch das Vorwissen sein darf, bei dem ein Benutzer auf die Information der Datenbank zugreifen darf.
Die InitStatements werden in vier Schritte initalisiert:

1. Die **Tabellen** werden mit dem Kommando *AS admin : CREATE TABLE tablename(parameter)* initialisiert.
2. Die **Benutzer** werden mit dem Kommando *AS admin : ADD USER benutzer* initialisiert.
3. Die **Sicherheitsregeln** werden mit dem Kommando *AS admin : SECRET anfrage('id') FOR benutzer THRESHOLD grenze* initalisiert.
4. Das **Füllen** der Datenbank mit den benötigten Werten wird mit dem Kommando *AS admin : INSERT IN tabelle ['id']*

Template

Die template.cpt ist die Vorlage für Angerona. In dieser werden die Tabellen aus den InitStatements und die definierten Fakten aus dem BeliefProgram initalisiert.

2.3 Beispiel

Im Beispiel wird eine Patientendatenbank eines Krankenhauses betrachtet, die das Rauchverhalten und die Elternbeziehung von Patienten speichert und ob diese Krebs haben. Die Datenbank hat dabei die Tabellen *Patienten, Raucher, Krebs, MutterHatKrebs* und *VaterhatKrebs*. In der Patientendatenbanken sind die Patienten Alice mit der *id* 1, Bob mit der *id* 2 und Carl mit der *id* 3 die Patienten , wobei Alice und Bob die Eltern von Carl sind. Alice raucht nicht, aber Bob und Carl rauchen. Alle drei Patienten haben Krebs.
Als Benutzer für das System existiert nur Mallory mit dem Benutzernamen *mallory*. Außerdem wird vom folgendem Probabilistisches Modell ausgegangen:

1. Jeder Patient entwickelt mit einer Wahrscheinlichkeit von 5% Krebs
2. Für jedes Elternteil das Krebs hat, steigt die Wahrscheinlichkeit vom Kind Krebs zu bekommen um 15%
3. Wenn ein Patient raucht, dann steigt die Wahrscheinlichkeit für Krebs um 25%

Zu Beginn wird das probabilistische Modell in ein Bayes-Netz übertragen. Dabei wird davon ausgegangen, dass die Wahrscheinlichkeit, dass ein Patient raucht 25% [7], jeweils ein Elternteil Krebs hat 10% entspricht. Ein Patient ist in diesem Fall zu 100% ein Patient, da eine Patientendatenbank betrachtet wird [2].

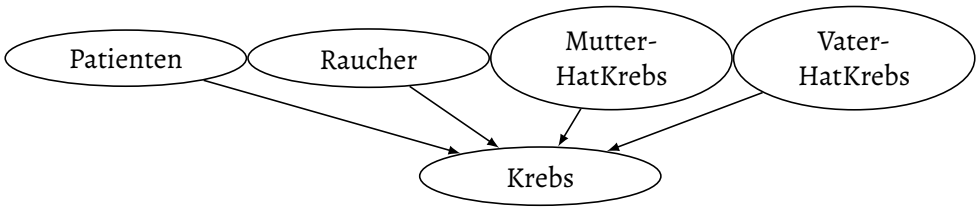
Abbildung 2.3: Bayes-Netz

Patienten $P(P)$
1.00

Raucher $P(R)$
0.25

Mutter hat Krebs $P(M)$
0.1

Vater hat Krebs $P(V)$
0.1



Krebs				
P	R	M	V	$P(L)$
T	T	T	T	0.6
T	T	T	F	0.45
T	T	F	T	0.45
T	T	F	F	0.3
T	F	T	T	0.35
T	F	T	F	0.2
T	F	F	T	0.2
T	F	F	F	0.05

Die Fälle in denen P *false* sind bewusst nicht in der Tabelle für Krebs gelistet, da diese Fälle nicht eintreffen können. Die Warhscheinlichkeiten dafür, dass ein Patient Krebs hat berechnen sich aus $P(L) = \sum_{i=0}^{N} p_i|X(p)|X(p_i) = true$

Summe richtig aufschreiben

.

Mithilfe des Bayes-Netz wird das BeliefProgram definiert, indem jeder Knoten der keine Abhängigkeiten besitzt definiert wird. Im Beispiel sind das die Knoten Patient, Raucher, MutterHatKrebs und VaterHatKrebs.

Programmtext 2.4: Beispiel für Knoten ohne Abhängigkeiten

```
1  patient(X) :– p_patient(X).
2  raucher(X) :– p_raucher(X).
3  mutterHatKrebs(X) :– p_mutterHatKrebs(X).
4  vaterHatKrebs(X) :– p_vaterHatKrebs(X).
5
6  1/1 :: p_patient(1).
7  25/100 :: p_raucher(1).
8  1/10 :: p_mutterHatKrebs(1).
9  1/10 :: p_vaterHatKrebs(1).
10 \dots
11 1/1 :: p_patient(3).
12 25/100 :: p_raucher(3).
13 1/10 :: p_mutterHatKrebs(3).
14 1/10 :: p_vaterHatKrebs(3).
```

Anschließend können die Knoten mit Abhängigkeiten definiert werden. Im Beispiel ist das nur der Knoten Krebs. Dieser wird folgendermaßen definiert:

Programmtext 2.5: Beispiel für Knoten mit Abhängigkeiten

```
1  Krebs(X) :– patient(X), raucher(X), mutterHatKrebs(X), vaterHatKrebs(X), p_p_r_m_v(X).
2  Krebs(X) :– patient(X), raucher(X), mutterHatKrebs(X), NOT vaterHatKrebs(X),
       p_p_r_m_nv(X).
3  ...
4  Krebs(X) :– patient(X), NOT raucher(X), NOT mutterHatKrebs(X), NOT vaterHatKrebs(X),
       p_np_nr_nm_nv(X).
5
6  6/10 :: p_p_r_m_v(1).
7  45/100 :: p_p_r_m_nv(1).
8  ...
9  5/100 :: p_np_nr_nm_nv(1).
10
11 ...
12
13 6/10 :: p_p_r_m_v(3).
```

14 45/100 :: *p_p_r_m_nv*(3).
15 ...
16 5/100 :: *p_np_nr_nm_nv*(3).

Damit wäre das Angreifermodell mit dem *BeliefProgram* vollständig initialisiert.

Die InitStatements werden einfach nach den vorher angegeben vier Schritten initialisiert.

1. Für jeden Knoten wird eine **Tabelle** angelegt:

   AS admin : *CREATE TABLE patient*(*id*)
   AS admin : *CREATE TABLE raucher*(*id*)
   AS admin : *CREATE TABLE mutterHatKrebs*(*id*)
   AS admin : *CREATE TABLE vaterHatKrebs*(*id*)
   AS admin : *CREATE TABLE krebs*(*id*)

   ---

2. Der **Benutzer** Mallory wird folgendermaßen angelegt:

   AS admin : *ADD USER mallory*

   ---

3. Die **Sicherheitsregeln** können beliebig gewählt werden. Im folgenden darf Mallory auf die Daten nur zugreifen, wenn sie mit einer Wahrscheinlichkeit von unter 50% weiß, dass ein beliebiger Patient Krebs hat.

   AS admin : *SECRET cancer*('1') *FOR mallory THRESHOLD* 1/2
   AS admin : *SECRET cancer*('2') *FOR mallory THRESHOLD* 1/2
   AS admin : *SECRET cancer*('3') *FOR mallory THRESHOLD* 1/2

   \item \textbf{Gefüllt} wird die Datenbank mit den oben gegeben Werten dann folgendermaßen:

   AS admin : *INSERT in patient*('1')
   AS admin : *INSERT in patient*('2')
   AS admin : *INSERT in patient*('3')
   AS admin : *INSERT in raucher*('2')
   AS admin : *INSERT in raucher*('3')
   AS admin : *INSERT in mutterHatKrebs*('1')
   AS admin : *INSERT in vaterHatKrebs*('1')
   AS admin : *INSERT in krebs*('1')
   AS admin : *INSERT in krebs*('2')
   AS admin : *INSERT in krebs*('3')

Im tempalte werden alle verwendeten Variablen aus dem BeliefProgram folgendermaßen initalisiert:

---

patient: []
raucher: []

mutterHatKrebs: []
vaterHatKrebs: []
krebs: []
p_p_r_m_v: []
p_p_r_m_nv: []
…
p_np_nr_nm_nv: []

3

Auswertung

In diesem Kapitel wird der Prototyp von Angerona auf drei verschiedene Patientendatenbanken angewendet. Betrachtet werden die Datenbanken MIMIC III, eICU und der Patientengenerator Synthea.

3.1 MIMIC III

MIMIC III (Medical Information Mart for Intesive Care) III [19] ist eine für Forschungszwecke frei zugängliche Datenbank, die pseudonymisiert Patientendaten und dazugehörige klinische Daten, die in Intensivstationen in einem Spezialkrankenhaus eingewiesen wurden, speichert. Die Daten stammen dabei aus dem Beth Israel Deaconess Medical Center in Boston, Massachusetts und wurden in dem Zeitraum vom Juni 2001 bis Oktober 2012 erfasst. Die Datenbank enthält 58976 Krankenhauseinweisungen für 38645 Erwachsene und 7875 Neugeborene. Gespeichert werden Daten wie Vitalparameter, Medikamente, Labormessungen, Beobachtungen und Notizen, die vom Personal dokumentiert wurden, Flüssigkeitsbilanzen, Verfahrenscodes, Diagnosecodes, Aufenthaltsdauer, Überlebensdaten und mehr.

Im folgenden wird für jeden Patienten in der MIMIC III Datenbank die Information, dass dieser Krebs hat, abgesichert. Die Wahrscheinlichkeitswerte des Vorwissens vom Angreifermodell wurden aus der MIMIC III Datenbank ausgelesen und somit hat jeder Angreifer ein perfektes Vorwissen.

reicht wenn man schreibt perfekt?

.

Dabei wird jede Krankenhauseinweisungen als ein Patienten betrachtet, wodurch Patienten auch mehrmals gelistet sein können, wenn diese mehrmals eingewiesen wurden. Risikofaktoren und somit Abhängiekiten für Krebs sind in dem Fall Alter, Geschlecht und Rauchverhalten des Patienten.[14, 8] Um diese Daten aus der MIMIC III Datenbank zu erhalten werden die folgenden Tabellen benötigt :

1. Die Tabelle *ADMISSIONS* enthält Informationen über die Einweisung ins Krankenhaus. Dabei ist jeder Krankenhausaufenthalt einer eindeutigen *HADM_ID* zugeord-

net. Dies ist immer unsere Ausgangstabelle, da alle Krankenhausaufenthalte betrachtet werden und die Tabelle enthält Informationen zu demographischen Daten, Ein- und Ausweisungszeiten und erste Einweisungsinformationen.
2. Die Tabelle *PATIENTS* enthält Informationen über jeden Patienten. Dabei ist jeder Patient einer eindeutigen *SUBJECT_ID* zugeordnet. Die Tabelle enthält Informationen über das Geschlecht, Geburtsdatum und Todesdatum, falls vorhanden.
3. Die Tabelle *DIAGNOSES_ICD* enthält zu jeder *HADM_ID* oder *SUBJECT_ID* die dazugehörige Diagnose als icd9-code(International Classification of Diseases). icd9-codes sind standardisierter Codes, die verwendet werdeb um Krankheiten , Verletzungen oder sonstige Diagnosen International einheitlich zu speichern[9].
4. Die Tabelle *NOTEVENTS* enthält alle Notizen zu den Patienten.

Dabei wurde das Alter in die Intervalle ≤ 40 mit 13265 , $< 40 bis \leq 60$ mit 14382 und > 60 mit 31329 eingeteilt, wobei die Wahrscheinlichkeit für Krebs mit dem Alter steigt. Dies lässt sich berechnen, indem die Differenz zwischen des Geburtsdatum aus der Tabelle *PATIENTS* und dem Einweisungsdatum aus der Tabelle *ADMISSIONS* berechnet wird. Das Geschlecht jedes Patienten kann aus der Tabelle *PATIENTS* unter dem Attribut *gender* ausgelesen werden und man erhält 32950 Männer und 26026 Frauen, wobei Männer eine höhere Wahrscheinlichkeit haben Krebs zu bekommen als Frauen. Das Rauchverhalten wurde aus der Tabelle *NOTEVENTS* ausgelesen, indem nach den Schlagwörtern „smoke“ und „cigarette“ gesucht wurde. Somit erhält man 25237 Raucher, wobei Raucher ein erhöhtes Risiko haben Krebs zu bekommen.
Um die Krankenhausaufenthalte zu bekommen, für die Patienten bei den Krebs diagnostiziert wurde, werden die icd9-codes für Krebs [3] mit den aus der *DIAGNOSES_ICD* verglichen und bei Gleichheit hinzugefügt. Dafür wurde ein Skript geschrieben, dass alle icd9-codes aus einer Liste extrahiert und in an das icd9-code Format von der MIMIC III Datenbank umwandelt und anschließend diese in eine extra angefertigte Tabelle kopiert. Somit kann man die Tabellen direkt miteinander vergleichen und erhält 13658 Krebspatienten. Mit diesen Informationen kann man das Bayes-netz skizzieren, dass in Abbildung 3.1 **??** zu sehen ist.

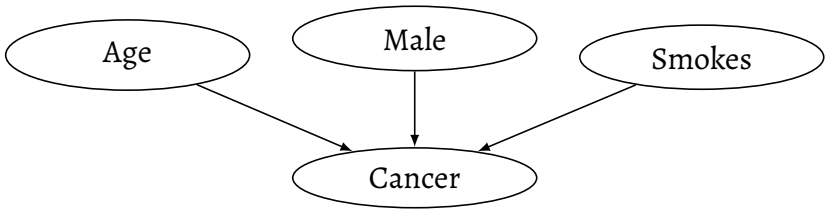
Alle Bayes-netze sind noch nicht richtig referenziert. Muss mir noch was einfallen lassen für Bayes-Netze generell.

.

Age		
$P(A = [0, 40])$	$P(A = (40, 60]$	$P(A = (60, 89])$
0.225	0.244	0.531

Male
$P(M)$
0.559

Smokes
$P(S)$
0.428



Cancer			
Age	Smokes	Male	$P(C \mid A, M, S)$
<= 40	T	T	0.0247
<= 40	T	F	0,027
<= 40	F	T	0.022
<= 40	F	F	0.025
40 – 60	T	T	0.182
40 – 60	T	F	0.242
40 – 60	F	T	0.15
40 – 60	F	F	0.2
> 60	T	T	0.322
> 60	T	F	0.3
> 60	F	T	0.271
> 60	F	F	0.237

Das Bayes-Netz wird in ein Angreifermodell nach dem Schema aus 2.2 übertragen und als beliefProgram.pbl gespeichert. Für das Alter werden dabei die *annotated disjunction* verwendet, da

Die initStatements werden nach dem Schema von 2.2 initialisiert, indem die Tabellen jeweils ein Knoten aus dem Bayes-Netz darstellen. Die Benutzer und die Sicherheitsregeln für Krebs können hierbei beliebig erstellt werden, sollten jedoch nicht zu hoch sein, da sonst die Laufzeit ungeeignet für eine Implementierung ist.
Für das füllen der Tabellen werden die *HADM_ID*s verwendet. Dafür wurde ein Bash-Skript geschrieben, dass alle *HADM_ID* s aus der Datenbank mit den benötigten Werten filtert und anschließend im passendem Format speichert.

3.2 eICU

eICU ist eine Datenbank, die aus einer großen Anzahl von Daten aus verschiedenen Krankenhäusern der USA besteht. Dabei ist MIMIC III nicht Teil der eICU, wodurch dies ein völlig unabhängigen Datensatz darstellt. Alle Tabellen wurden so Pseudonymisiert, dass diese dem HIPAA Standard entsprechen. Dadurch kommt es auch, dass Patienten mit einem Alter über 89 in der Tabelle nicht existieren, da diese mit „>89“ dargestellt werden.

Die Daten stammen aus dem Jahr 2014 bis 2015 und wurden dabei zufällig aus verschiedenen Krankenhäusern der USA gewählt und anschließend wurde jedem Krankenhausaufenthalt und Patient eine eindeutige Identifikationsnummer zugeordnet.
Die Datenbank enthält 200859 Krankenhauseinweisungen für 139367 Patienten aus 208 verschiedenen Krankenhäusern. Zu den Daten gehören Virtalparameter, Messungen, Pflegepläne, Art und Schweregrad der Krankheit, Diagnoseinformationen, Behandlungsinformationen und mehr.

Im folgenden wird für jeden Patienten in der eICU Datenbank die Information, dass dieser Krebs hat, abgesichert. Die Wahrscheinlichkeitswerte des Vorwissens vom Angreifermodell werden aus der eICU Datenbank ausgelesen und somit hat jeder Angreifer wieder ein perfektes Vorwissen.

reicht wenn man schreibt perfekt?

Dabei werden die Krankenhauseinweisungen als id gewählt.
Risikofaktoren und somit Abhängigkeiten für Krebs sind in dem Fall Alter, Geschlecht und eine Chemotherapie oder andere Onkologische Maßnahmen .[14, 8] Um diese Daten aus der eICU Datenbank zu erhalten werden die folgenden Tabellen benötigt :

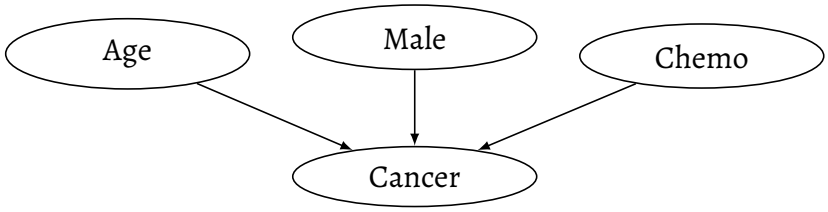
1. Die Tabelle *ADMISSIONSDX* stellt für jede Station in der eine Diagnose gestellt wurde, die Diagnoseinformationen bereit. Die Tabelle enthält Informationen zu demographischen Daten, Ein- und Ausweisungszeiten und erste Einweisungsinformationen.
2. Die Tabelle *PATIENT* enthält genauere Informationen über einen Patienten. Dabei ist jeder Patient einer eindeutigen *patientUnitStayID* zugeordnet, wodurch Patienten mehrfach auftauchen können, wenn diese mehrfach Eingewiesen wurden. Dies ist immer die Ausgangstabelle, da alle Krankenhausaufenthalte betrachtet werden. Zu den Daten gehören Informationen über das Geschlecht, Alter, ethnische Zugehörigkeit und mehr.
3. Die Tabelle *diagnosis* enthält Diagnosedaten zu jedem Krankenhausaufenthalt. Diese werden im ICD9 Format gespeichert.
4. Die Tabelle *TREATMENT* enthält Behandlungsinformationen über den Krankenhausaufenthalt.

Die Intervalle für das Alter wurden wie bei MIMIC III gewählt und das Alter kann einfach aus der Tabelle *PATIENT* aus dem Attribut *age* ausgelesen werden. Somit erhält man für die Intervalle $(0, 40] = 23091$, $(40, 60] = 57065$ und $(60, 89] = 120608$ Patienten. Dabei wurden 95 Patienten nicht berücksichtigt, weil das Attribut *age* leer war.
Nach dem selben Prinzip wie beim Alter kann auch das Geschlecht ausgelesen werden, indem das Attribut *gender* aus der Tabelle *PATIENT* ausgelesen wird. Somit erhält man eine Verteilung von 92303 Frauen und 108379 Männern. Hierbei sind wieder 177 nicht berücksichtigt, weil das Attribut *gender* für diese leer war.
Ob ein Patient eine Chemotherapie oder andere Onkologische Maßnahmen hatte, lässt sich aus der Tabelle *TREATMENT* auslesen, indem man nach dem Suchwort „*oncology*“ und „*chemotherap*“sucht. Dabei sind diese eine garantierte Aussage darüber, ob dieser Patient Krebs hat. In der eICU Datenbank gibt es 2387 Patienten, bei denen dies der Fall ist.

Um die Krebspatienten zu erhalten, werden aus der *ADMISSIONSDX* Tabelle die *icd9-codes* für Krebs gefiltert. Das führt dazu, dass 8179 Krebspatienten in der Datenbank enthalten sind. Daraus ergibt sich folgendes Bayes-Netz:

Age		
$P(A = [0, 40])$	$P(A = (40, 60]$	$P(A = (60, 89])$
0, 115	0, 284	0, 6

Male (M)	Oncology $P(O)$
0, 46	0,012



Cancer			
Age	Ocnology	Male	$P(C \mid A, M, O)$
<= 40	T	T	1
<= 40	T	F	1
<= 40	F	T	0.016
<= 40	F	F	0.02
40 – 60	T	T	1
40 – 60	T	F	1
40 – 60	F	T	0.033
40 – 60	F	F	0.042
> 60	T	T	1
> 60	T	F	1
> 60	F	T	0.044
> 60	F	F	0.034

3.3 Synthea

Synthea [13, 4] ist eine open-source Software, mit der Patientendatenbanken generiert werden können. Die demographischen Daten werden dabei anhand von öffentlich Zugänglichen demographischen Daten vom US Census Bureau [5] generiert. Synthea unterstützt dabei die Generierung der Datenbank in den Formaten ccda, fhir, text und CSV. Im folgendem Fall werden ausschließlich CSV-Dateien generiert.
Um Synthea für Angerona kompatibel zu machen, müssen die generierten Patientenid's von UUID zu einer eindeutig identifizierbaren id, die nur aus Zahlen besteht umgewandelt werden, weil Angerona als input nur Zahlen erlaubt und die UUID noch Sonderzeichen und Buchstaben enthält. Dies kann implementiert werden, indem im Source Ordner von Synthea in der LifeCycleModule.java die Zeile 136 ersetzt wird durch folgende :

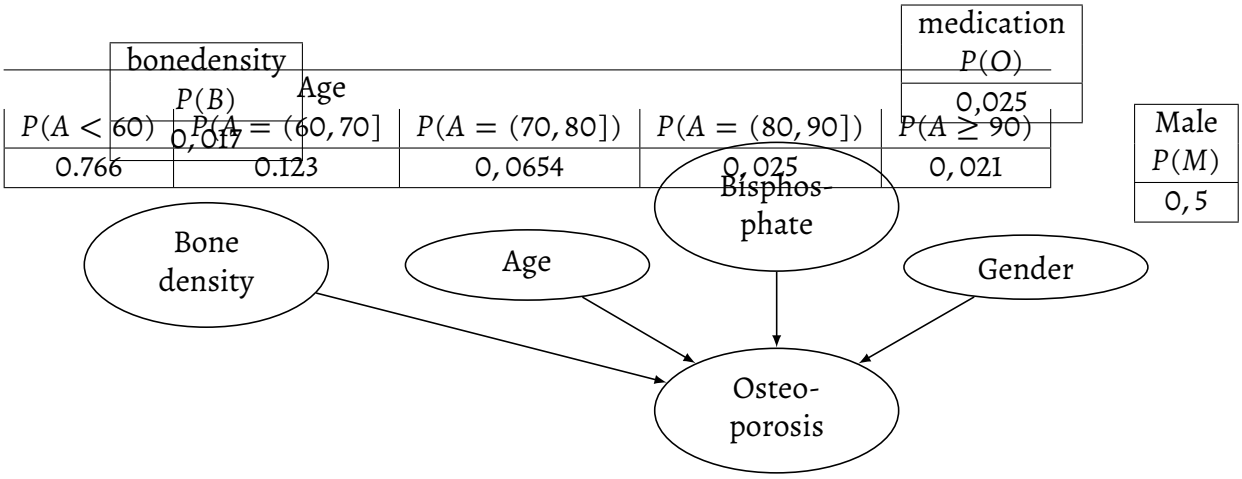
```
attributes.put(Person.ID, String.format("\%040d", new
    BigInteger(UUID.randomUUID().toString().replace("-", ""), 16)));
```

Anschließend kann man mit dem Kommando `.\run_synthea -p [anzahl] -o false` die CSV-Dateien generieren und in eine beliebige Datenbank übertragen. Mit -o false wird genau die Anzahl der Patienten, die übergeben werden, da sonst auch tote Patienten generiert werden würden, die Synthea Standardmäßig mitzählt.

Um die Daten für das Vorwissen des Angreifers zu modellieren, wird das Generic Module Framework von Synthea verwendet. Dadurch lassen sich die Abhängigkeiten für Krankheiten oder sonstige Einweisungsgründe ablesen, indem man in der `Module Gallary` eine beliebige öffnet. Im folgenden wird das Angreifermodell für die Krankheit Osteoporose definiert. Aus der *Module Gallary* geht hervor, dass Osteoporose abhängig ist von folgenden Faktoren :

1. **Geschlecht**: Frauen haben ein erhöhtes Risiko an Osteoporose zu erkranken als Männer. Dabei haben Männer eine 0.6 Fache geringere Wahrscheinlichkeit als Frauen.
2. **Alter**: Personen ab 60 Jahren können erst an Osteoporose erkranken. Dabei steigt die Wahrscheinlichkeit mit dem Alter in 10er Schritten immer mehr an bis zum Alter 90, wodurch sich ein Intervall von < 60, 60-70 , 70-80, 80-90 und ≥ 90 ergibt.
3. **Knochendichte(bone-density)**: Wenn die Knochendichte zwischen -3.8 und -2.5 liegt, ist dies ein garantiertes Indiz für Osteoporose.
4. **Medikament** Bisphosphate: Wenn ein Patient das Medikament Bisphosphate verschrieben bekommen hat, dann ist dies ebenfalls ein garantiertes Indiz für Osteoporose.

Das Vorwissen für das Geschlecht wurde aus der demographics.csv im Synthea Verzeichniss ausgelesen, indem der durchschnitt aller Städte gebildet wurde, wobei ein Wahrscheinlichkeitswert von 0.5 berechnet wurde. Die Daten für die Medikamente, Alter und die Knochendichte wurden aus einer Tabelle mit 100.000 Patienten gesampelt.
Im folgenden Bayes-Netz werden die Wahrscheinlichkeitswerte für $P(A < 60)$ nicht gelistet, weil diese die Wahrscheinlichkeit 0 haben. Ebenfalls werden die Wahrscheinlichkeiten für $P(B = TRUE)$ und $P(O = TRUE)$ nicht gelistet, weil diese die Wahrscheinlichkeit 1 haben. Dadurch ergibt sich folgendes Bayes-Netz :



Osteoporosis				
Age	Male	Medication	Bone density	$P(O \mid A, M, O, B)$
60 – 70	T	F	F	0, 0714
60 – 70	F	F	F	0.1
70 – 80	T	F	F	0.0714
70 – 80	F	F	F	0.1
80 – 90	T	F	F	0.143
80 – 90	F	F	F	0.2

Ein erweitertes Angreifermodell mit mehr Abhängigkeiten wurde durch hinzufügen des Moduls *Injury* erreicht. Hierfür wurde das Angreifermodell mit dem Vorwissen für ein Knochenbruch erweitert. Ein Knochenbruch ist dabei Abhängig von Osteoporose unnd ein Knochenbruch wird unterteilt in Armbruch, Knöchelbruch, Gelenkbruch, Rippenbruch, Schlüsselbeinbruch und gebrochene Hüfte. Jedoch lässt sich dies nicht in einem für Angerona konformen Bayes-Netz modellieren, da dadurch die Bedingung verletzt wird, dass das Bayes-Netz ein Poly-tree sein muss. Durch die genannte Modellierung würde nämlich ein Zyklus entstehen zwischen den Knoten Osteoporose ↔Knochenbruch ↔Art des Bruches, weil z.B. ein Armbruch abhängig ist vom Knochenbruch und von Osteoporose.
Deshalb wurde die Wahrscheinlichkeit Knochenbruch als Knoten ohne Abhängigkeit definiert und die Wahrscheinlichkeit für ein Knochenbruch aus einer generierten Patientendatenbank mit 100.000 Patienten gesampled

gesampled? Vielleicht umschreiben

. Die Wahrscheinlichkeiten für die einzelnen Brüche können aus der Module Gallary für *Injury* ausgelesen werden. Somit ergibt sich folgendes Bayes-Netz :

3.4 Angreifer modellierung

3.5 Vor und nachteile

4

Laufzeiten der Patientendatenbanken

Um die folgenden Experimente und Auswertung zu reproduzieren, sollten erstmal die Anwendungen und Einstellungen vorgestellt werden. Der genutzte Computer hat einen AMD Ryzen 2600x Sechs-Kern Prozessor die mit 3.6GHz laufen, 16GB RAM mit 2400MHz und nutzt Windows 10 Enterprise LTSC als Betriebssystem. Das Projekt wurde jedoch aus Kompatibilitätsgründen auf einer virtuellen Maschine ausgeführt. Die VM läuft auf Oracle VM VirtualBox Version 6.1.6 und nutzt Ubuntu 20.04.1 LTS. Dabei wurden der VM Sechs von Zwölf Threads und 8GB RAM bereitgestellt.

Alle Datenbanken laufen auf einer Virtuellen Maschine in einem PowerEdge R530 Server. Dieser Server hat zwei Intel Xeon E5-2620 v3 Prozessoren 2,4GHz mit jeweils 6 Core / 12 Threads und 64GB DDR4-SDRAM. Als Datenbanksystem wird PostgreSQL Version 9.5.24 verwendet.

5

Zusammenfassung und Ausblick

This template document got much longer than I had initially intended with more and more hints and comments becoming part of the text. The reason is, of course, that writing a thesis is not easy since there are a *lot* of things to consider. However, you have six months to write your thesis, so you stand a decent chance to get most things right.

Do some great scientific research now and report on it in a thesis that is a pleasure to read.

Liste der noch zu erledigenden Punkte

Literatur

[1] In: URL: https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html (besucht am 30. 12. 2020).

[2] In: URL: https://mguarnieri.github.io/publication/csf2017/ (besucht am 30. 12. 2020).

[3] URL: https://medicine.yale.edu/intmed/vacs/instruments/.

[4] URL: https://github.com/synthetichealth/synthea.

[5] URL: https://www.census.gov/.

[6] In: 30. Dez. 2020. URL: https://dtai.cs.kuleuven.be/problog/index.html#.

[7] Jan. 2020. URL: https://de.statista.com/statistik/daten/studie/1099197/umfrage/anteil-der-raucher-in-der-eu-nach-geschlecht/#professional (besucht am 12. 01. 2021).

[8] Andrei Barasch Douglas E. Morse, D. K. E. E. Smoking, Gender, and Age as Risk Factors for Site-Specific Intraoral Squamous Cell Carcinoma. A Case-Series Analysis. In: 25. Aug. 1993. URL: https://acsjournals.onlinelibrary.wiley.com/doi/pdfdirect/10.1002/1097-0142(19940201)73:3%3C509::AID-CNCR2820730303%3E3.0.CO;2-X (besucht am 22. 01. 2020).

[9] Beutelspacher, A. *„Das ist o. B. d. A. trivial!": Tipps und Tricks zur Formulierung mathematischer Gedanken (Mathematik für Studienanfänger)*. Ninth, updated edition. Vieweg+Teubner Verlag, 2009. DOI: 10.1007/978-3-8348-9075-7. URL: https://apps.who.int/iris/bitstream/handle/10665/39473/9241541334-eng.pdf?sequence=1&isAllowed=y.

[10] *Datenschutz-Grundverordnung(DSGVO)*. URL: https://dsgvo-gesetz.de/ (besucht am 22. 01. 2020).

[11] Golibrzuch, P., Möller, R. und Kaya, A. A study of the rough set approach for image understanding. In: Jan. 2021.

[12] Ingelheim, A. Das erste Jahr DSGVO - Eine Bestandsaufnahme. In: Apr. 2019. URL: http://www.ctan.org/pkg/varioref.

[13] Jason Walonoski Mark Kramer, J. N. A. Q. C. M. D. H. C. D. K. D. T. G. S. M. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic lectronic health care record. In: URL: https://watermark.silverchair.com/ocx079.pdf?token=AQECAHi208BE49Ooan9kkhW_Ercy7Dm3ZL_9Cf3qfKAc485ysgAAAqEwggKdBgkqhkiG9w0BBwagggKOMIICigIBADCCAoMGCSqGOuKibqkLHwcuMDot2-MtOyLcGc57WFcQzzHLtRHHcu8JDHx8do9Xm_v-N3MBKn4BK42JxgtPavGnFnWPfLGxAFB4CMwfPgL90o62vbj27v2VbLt1rhGc14FtHFMYi3v9EA07-5PT0z4JBXiwR0E7dr9x7sR4l8zVqmLfUkcvCcH6qxfA1g4xHP2bFUYIDgQUsiwOJ20uiQThkJU-Ze2ahgZ4qW9LZ-yJO3eczuZDPw_

nsMhmEx3ocl3HcZmREf1nqsakMvuLL0Jq1y - beb7y3dyuEzh6QLeme5mx5089mHyRlh22t _ ksbEP _ uV5cduUrQxjihPxc1SVTUCCzGXAty _ L8GO8XA - K0rv - _UGUEwmSOMl8t0z8BchfXMNVOdrK9iGJsP39FUdfIGdt2R382SsBupHMeF9AoaFQL pyd7O5v5oB0Yp2C3S6XiJL3xsMCC8H8OQk3SEBS5I4npew2Jo8xlrlhHQpXVkWea46Y MALkttRAqi9MDocU2ms1m086niEWj60DfaTVK66LJYPwd _ XSTIrwSQoZgeWxGPhsaYPzADT0Q-uDXDY1C8SGM29GxOzfwtZ29Bst8OvO5jChH_ W0-PXMIPbnIaLc1gmMWockWwaixjed.

[14] Kawsar Ahmed Abdullah-Al-Emran, T. J. R. F. M. M. T. R. F. A. Early Detection of Lung Cancer Risk Using Data Mining. In: 2013.

[15] Kučera, M., Tsankov, P., Gehr, T., Guarnieri, M. und Vechev, M. Synthesis of Probabilistic Privacy Enforcement. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, S. 391–408. ISBN: 9781450349468. DOI: 10.1145/3133956.3134079. URL: https://doi.org/10.1145/3133956.3134079%7D.

[16] Luc De Raedt×, A. K. und Toivonen, H. *ProbLog: A Probabilistic Prolog and its Application in Link Discovery*. Techn. Ber. Machine Learning Lab, Albert-Ludwigs-University Freibur, 2007.

[17] Ness, R. B. Influence of the HIPAA Privacy Rule on Health Research. Sophisticated Bibliographies in LaTeX. In: American Medical Association.

[18] Peter Norvig, S. R. und *Artificial Intelligence, A Modern Approach*. Third Edition. Pearson, 2010.

[19] Robertson, W. *MIMIC-III, a freely accessible critical care database*. 24. Feb. 2016. URL: file:///C:/Users/Gewen/Downloads/sdata201635%20(2).pdf (besucht am 24. 01. 2020).