Artificial Intelligence 2
Summer Semester 2024

– Lecture Notes –

Dr. Dennis Müller

Professur für Wissensrepräsentation und -verarbeitung
Informatik, FAU Erlangen-Nürnberg
dennis.mueller@FAU.de

2024-04-14

0.1 Preface

Disclaimer: This document is adapted from the notes for the course of the same name by Prof. Dr. Michael Kohlhase. It should be assumed by default that all credit goes primarily to him; whereas all mistakes should be assumed to be mine.

0.1.1 Course Concept

Objective: The course aims at giving students a solid (and often somewhat theoretically oriented) foundation of the basic concepts and practices of artificial intelligence. The course will predominantly cover symbolic AI – also sometimes called "good old-fashioned AI (GofAI)" – in the first semester and offers the very foundations of statistical approaches in the second. Indeed, a full account sub symbolic, machine learning based AI deserves its own specialization courses and needs much more mathematical prerequisites than we can assume in this course.
Context: The course "Artificial Intelligence" (AI 1 & 2) at FAU Erlangen is a two-semester course in the "Wahlpflichtbereich" (specialization phase) in semesters 5/6 of the Bachelor program "Computer Science" at FAU Erlangen. It is also available as a (somewhat remedial) course in the "Vertiefungsmodul Künstliche Intelligenz" in the Computer Science Master's program.
Prerequisites: AI-1 & 2 builds on the mandatory courses in the FAU Bachelor's program, in particular the course "Grundlagen der Logik in der Informatik" [Glo], which already covers a lot of the materials usually presented in the "knowledge and reasoning" part of an introductory AI course. The AI 1& 2 course also minimizes overlap with the course.

The course is relatively elementary, we expect that any student who attended the mandatory CS courses at FAU Erlangen can follow it.
Open to external students:

Other Bachelor programs are increasingly co-opting the course as specialization option. There is no inherent restriction to computer science students in this course. Students with other study biographies – e.g. students from other Bachelor programs our external Master's students should be able to pick up the prerequisites when needed.

0.1.2 Course Contents

Goal: To give students a solid foundation of the basic concepts and practices of the field of Artificial Intelligence. The course will be based on Russell/Norvig's book "*Artificial Intelligence; A modern Approach*" [RN09]
Artificial Intelligence I (the first semester): introduces AI as an area of study, discusses "rational agents" as a unifying conceptual paradigm for AI and covers problem solving, search, constraint propagation, logic, knowledge representation, and planning.
Artificial Intelligence II (the second semester): is more oriented towards exposing students to the basics of statistically based AI: We start out with reasoning under uncertainty, setting the foundation with Bayesian Networks and extending this to rational decision theory. Building on this we cover the basics of machine learning.

0.1.3 This Document

Format: The document mixes the slides presented in class with comments of the instructor to give students a more complete background reference.
Caveat: This document is made available for the students of this course only. It is still very much a draft and will develop over the course of the current course and in coming academic years. **Licensing:** This document is licensed under a Creative Commons license that requires attribution, allows commercial use, and allows derivative works as long as these are licensed under the same license. **Knowledge Representation Experiment:** This document is also an experiment in knowledge representation. Under the hood, it uses the sTEX package [Koh08; sTeX], a TEX/LATEX extension for semantic markup, which allows to export the contents into

active documents that adapt to the reader and can be instrumented with services based on the explicitly represented meaning of the documents.

0.1.4 Acknowledgments

Materials: Most of the materials in this course is based on Russel/Norvik's book "Artificial Intelligence — A Modern Approach" (AIMA [RN95]). Even the slides are based on a LaTeX-based slide set, but heavily edited. The section on search algorithms is based on materials obtained from Bernhard Beckert (then Uni Koblenz), which is in turn based on AIMA. Some extensions have been inspired by an AI course by Jörg Hoffmann and Wolfgang Wahlster at Saarland University in 2016. Finally Dennis Müller suggested and supplied some extensions on AGI. Florian Rabe, Max Rapp and Katja Berčič have carefully re-read the text and pointed out problems.

All course materials have bee restructured and semantically annotated in the sTeX format, so that we can base additional semantic services on them.

AI Students: The following students have submitted corrections and suggestions to this and earlier versions of the notes: Rares Ambrus, Ioan Sucan, Yashodan Nevatia, Dennis Müller, Simon Rainer, Demian Vöhringer, Lorenz Gorse, Philipp Reger, Benedikt Lorch, Maximilian Lösch, Luca Reeb, Marius Frinken, Peter Eichinger, Oskar Herrmann, Daniel Höfer, Stephan Mattejat, Matthias Sonntag, Jan Urfei, Tanja Würsching, Adrian Kretschmer, Tobias Schmidt, Maxim Onciul, Armin Roth, Liam Corona, Tobias Völk, Lena Voigt, Yinan Shao, Michael Girstl, Matthias Vietz, Anatoliy Cherepantsev, Stefan Musevski, Matthias Lobenhofer, Philipp Kaludercic, Diwarkara Reddy, Martin Helmke, Stefan Müller, Dominik Mehlich, Paul Martini, Vishwang Dave, Arthur Miehlich, Christian Schabesberger, Vishaal Saravanan, Simon Heilig, Michelle Fribrance, Wenwen Wang, Xinyuan Tu, Lobna Eldeeb.

0.1.5 Recorded Syllabus

The recorded syllabus – a record the progress of the course in the academic year 2024– is in the course page in the ALeA system at `https://courses.voll-ki.fau.de/course-home/ai-1`. The table of contents in the AI-2 notes at `https://courses.voll-ki.fau.de` indicates the material covered to date in yellow.

The recorded syllabus of AI-2 can be found at `https://courses.voll-ki.fau.de/course-home/ai-2`. For the topics planned for this course, see subsection 0.1.2.

Contents

Chapter 1

Administrativa

We will now go through the ground rules for the course. This is a kind of a social contract between the instructor and the students. Both have to keep their side of the deal to make learning as efficient and painless as possible.

About this course...

- AI1 and AI2 are "traditionally" taught by Prof. Michael Kohlhase (since 2016, on sabbatical this semester)
- This is the first time I'm teaching AI2 as a lecturer! ☺

But I've been a member of Prof. Kohlhase's research group since 2015 (Ph.D. 2019)

⇒ I'm familiar with the course content (Lead TA 2016 – 2019)

⇒ I've adopted *and adapted* his course material. The topics are the same, *but* I changed some notations, clarified and changed some definitions, restructured some parts (Hopefully for the better!)

⇒ Feel free to check out older versions of the course material *but* don't rely on them *entirely* (especially for exam prep!)

Also: I'm working on my habilitation currently

⇒ Teaching this course is part of that

⇒ Please take the course evaluation seriously ;) (I'm still learning and it helps me improve!)

Dates, Links, Materials

- **Lectures**: Tuesday 16:15 – 17:45 **H9**, Thursday 10:15 – 11:45 **H8**
- **Tutorials**:
  - Thursday 14:15 – 15:45 *Room 11501.04.023*
  - Friday 10:15 – 11:45 *Room 11501.02.019*
  - Friday 14:15 – 15:45 *Zoom:* `https://fau.zoom.us/j/97169402146`

- ▷ Monday 12:15 – 13:45 *Zoom:* `https://fau.zoom.us/j/97169402146`
- ▷ Tuesday 08:15 – 09:45 *Room 11302.02.134-113*

(Starting thursday in week 2 (25.04.2024))

- ▷ **studon**: `https://www.studon.fau.de/studon/goto.php?target=crs_5645530` (Used for announcements, e.g. homeworks, and homework submissions)
- ▷ **Video streams / recordings**: `https://www.fau.tv/course/id/3816`
- ▷ **Lecture notes / slides / exercises:** `https://kwarc.info/teaching/AI/` (Most importantly: `notes2.pdf` and `slides2.pdf`)
- ▷ **ALeA**: `https://courses.voll-ki.fau.de/course-home/ai-2`: Lecture notes, forum, **tuesday quizzes**, flashcards,...

Textbook: *Russel/Norvig: Artificial Intelligence, A modern Approach* [RN09]. Make sure that you read the edition ≥ 3 ⇝ vastly improved over ≤ 2.

AI-2 Homework Assignments

Homework Assignments: Every thursday (starting in the second week)
Small individual problem/programming/proof tasks

⚠ Homeworks give no bonus points, but without trying you are unlikely to pass the exam.

Homework/Tutorial Discipline:

- ▷ Start early! (many assignments need more than one evening's work)
- ▷ Don't start by sitting at a blank screen (talking & study group help)
- ▷ Humans will be trying to understand the text/code/math when grading it. (For those that *do* get graded – see later)
- ▷ Go to the tutorials, discuss with your TA! (they are there for you!)

- ▷ Homeworks will be posted on `kwarc.info/teaching/AI/assignments`. (Announced on studon)
- ▷ Sign up for AI-2 under `https://www.studon.fau.de/crs4941850.html`.
- ▷ Homeworks are handed in electronically there. (plain text, program files, PDF)
- ▷ Do not sign up for the "AI-2 Übungen" on StudOn (we do not use them)

It is very well-established experience that without doing the homework assignments (or something similar) on your own, you will not master the concepts, you will not even be able to ask sensible questions, and take very little home from the course. Just sitting in the course and nodding is not enough! If you have questions please make sure you discuss them with the instructor, the teaching assistants, or your fellow students. There are three sensible venues for such discussions: online in the lecture, in the tutorials, which we discuss now, or in the course forum – see below. Finally, it is always a very good idea to form study groups with your friends.

Tutorials for Artificial Intelligence 1

Weekly tutorials starting in week two – Lead TA: Florian Rabe (KWARC Postdoc, Privatdozent) (Room: 11.137 @ Händler building, florian.rabe@fau.de)

The tutorials:

- ▷ reinforce what was taught in class.
- ▷ allow you to ask any question you have in a protected environment.
- ▷ discuss the (solutions to) *homework assignments*

Caveat: We cannot grade all submissions :((too many students, too few TAs)
Group submission has not worked well in the past (too many freeloaders)
Likely solution: We will grade *one* exercise per week – but you should attempt all of them!

Life-saving advice: Go to your tutorial, and prepare for it by having looked at the slides and the homework assignments!

Doing your homework is probably even *more* important (and predictive of exam success) than attending the lecture!

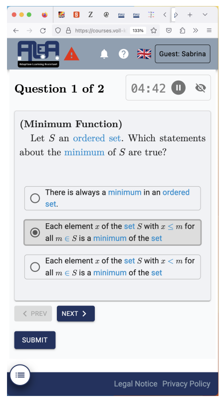
Tuesday Quizzes

Tuesday Quizzes: Every tuesday we start the lecture with a 10 min online quiz – the tuesday quiz – about the material from the previous week. (starts in week 2) **Motivations:** We do this to

- ▷ keep you prepared and working continuously. (primary)
- ▷ update the ALeA learner model (fringe benefit)
- ▷ give *bonus points* for the exam! (as an incentive)

The tuesday quiz will be given in the ALEA system

- ▷ https://courses.voll-ki.fau.de/quiz-dash/ai-2
- ▷ You have to be logged into ALEA!
- ▷ You can take the quiz on your laptop or phone, ...
- ▷ ... in the lecture or at home ...
- ▷ ... via WLAN or 4G Network. (do not overload)
- ▷ Quizzes will only be available 16:15-16:25!



Now we come to a topic that is always interesting to the students: the grading scheme.

Assessment, Grades

- ▷ **Overall (Module) Grade:**
  - ▷ Grade via the exam (Klausur) $\rightsquigarrow$ 100% of the grade.
  - ▷ Up to 10% bonus on-top for an exam with $\geq 50\%$ points. ($\leq 50\% \rightsquigarrow$ no bonus)
  - ▷ Bonus points $\widehat{=}$ percentage sum of the best 10 tuesday quizzes divided by 100.
- ▷ **Exam:** 90 minutes exam conducted in presence on paper (∼ Oct. 1. 2023)
- ▷ **Retake Exam:** 90 min exam six months later (∼ April 1. 2024)
- ▷ ⚠ You have to register for exams in campo in the first month of classes.
- ▷ **Note:** You can de-register from an exam on campo up to three working days before.

Due to the current AI hype, the course Artificial Intelligence is very popular and thus many degree programs at FAU have adopted it for their curricula. Sometimes the course setup that fits for the CS program does not fit the other's very well, therefore there are some special conditions. I want to state here.

⚠ Special Admin Conditions ⚠

- ▷ Some degree programs do not "import" the course Artificial Intelligence, and thus you may not be able to register for the exam via `https://campus.fau.de`.

- ▷ Just send me an e-mail and come to the exam, we will issue a "Schein".
- ▷ Tell your program coordinator about AI-1/2 so that they remedy this situation

▷ In "Wirtschafts-Informatik" you can only take AI-1 and AI-2 together in the "Wahlpflichtbereich".

- ▷ ECTS credits need to be divisible by five ⇝ $7.5 + 7.5 = 15$.

I can only warn of what I am aware, so if your degree program lets you jump through extra hoops, please tell me and then I can mention them here.

The ALeA System

Prerequisites

▷ **Remember: AI-1** dealt with situations with "complete information" and strictly computable, "perfect" solutions to problems. (i.e. tree search, logical inference, planning, etc.)

▷ **AI-2** will focus on *probabilistic* scenarios by introducing uncertain situations, and *approximate* solutions to problems. (Bayesian networks, Markov models, machine learning, etc.)

The following should therefore be seen as "*weak prerequisites*":

▷ AI-1 (in particular: PEAS, propositional logic/first-order logic (mostly the syntax), some logic programming)

▷ (very) elementary complexity theory. (big Oh and friends)

▷ rudimentary probability theory (e.g. from stochastics)

▷ basic linear algebra (vectors, matrices,...)

▷ basic real analysis (primarily:(partial) derivatives)

Meaning: I will *assume* you know these things, but some of them we will recap, and what you don't know will make things slightly harder for you, but by no means prohibitively difficult.

"Strict" Prerequisites

▷ **Mathematical Literacy**: Mathematics is the language that computer scientists express their ideas in ("*A search problem is a tuple* $(N, S, G, ...)$ *such that...*")

Note: This is a skill that can be *learned*, and more importantly, *practiced!* Not having/honing this skill *will* make things more difficult for you. Be aware of this and, if necessary, work on it – it will pay off, not only in this course.

▷ motivation, interest, curiosity, hard work. (AI-2 is non-trivial)

Note: Grades correlate significantly with invested effort; including, but not limited to: time spent on exercises, being here, asking questions, talking to your peers,...

What you should learn here...

▷ In the broadest sense: *A bunch of tools for your toolchest* (i.e. various (quasi-mathematical) models, first and foremost)

▷ the underlying *principles* of these models (assumptions, limitations, the math behind them ...)

▷ the ability to describe real-world problems in terms of these models, **where adequate** (...and knowing **when** they are adequate!), and

▷ the ideas behind effective *algorithms* that solve these problems (and to understand them well enough to implement them)

Note: You will likely never get payed to implement an algorithm that e.g. solves Bayesian networks. (They already exist)

But you might get payed to *recognize* that some given problem *can be* represented as a Bayesian network!

Or: you can recognize that it is *similar to* a Bayesian network, and reuse the underlying principles to develop new specialized tools.

In other words: Many things you learn here are *means to an end* (e.g. understanding the underlying *ideas* behind algorithms), not the end itself. But the best way to understand these means is to first treat them as an end in themselves.

Compare two employees

"We have the following problem and we need a solution: ..."

Employee 1: Deep Learning can do everything: "I just need ≈1.5 million labeled examples of potentially sensitive data, a GPU cluster for training, and a few weeks to train, tweak and finetune the model.

But *then* I can solve the problem... with a confidence of 95%, within 40 seconds of inference per input. Oh, as long as the input isn't longer than 15`unit`, or I will need to retrain on a bigger input layer..."

Employee 2: "...while you were talking, I quickly built a custom UI for an off-the-shelve `<problem>` solver that runs on a medium-sized potato and returns a *provably correct* result in a few milliseconds. For inputs longer than 1000`unit`, you might need a slightly bigger potato though..."

Moral of the story: Know your *tools* well enough to select the right one for the job.

Obviously, that is not to say that machine learning is not a useful tool! (It is!)

If your job is to e.g. filter customer support requests, or to recognize cats in pictures, trying to write a prolog program from scratch is probably the wrong approach: Just use a language model / image model and finetune it on a classification head.

But it is also not the only tool, and it is not always the right tool for the job – despite what some people might tell you. And even in scenarios where machine learning *can* yield decent results, it is not always the *best* tool. (Some people care about efficiency, explainability, etc ;))

Do use the opportunity to discuss the AI-2 topics with others. After all, one of the non-trivial skills you want to learn in the course is how to talk about Artificial Intelligence topics. And that takes practice, practice, and practice.

Chapter 2

Overview over AI and Topics of AI-II

We restart the new semester by reminding ourselves of (the problems, methods, and issues of) Artificial Intelligence, and what has been achived so far.

2.1 What is Artificial Intelligence?

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/21701`.
The first question we have to ask ourselves is "What is Artificial Intelligence?", i.e. how can we define it. And already that poses a problem since the natural definition *like human intelligence, but artificially realized* presupposes a definition of intelligence, which is equally problematic; even Psychologists and Philosophers – the subjects nominally "in charge" of natural intelligence – have problems defining it, as witnessed by the plethora of theories e.g. found at [WHI].

What is Artificial Intelligence? Definition

- ▷ **Definition 2.1.1 (According to Wikipedia).** Artificial Intelligence (AI) is intelligence exhibited by machines
- ▷ **Definition 2.1.2 (also).** Artificial Intelligence (AI) is a sub-field of computer science that is concerned with the automation of intelligent behavior.
- ▷ **BUT:** it is already difficult to define intelligence precisely.
- ▷ **Definition 2.1.3 (Elaine Rich).** Artificial Intelligence (AI) studies how we can make the computer do things that humans can still do better at the moment.

Maybe we can get around the problems of defining "what artificial intelligence is", by just describing the necessary components of AI (and how they interact). Let's have a try to see whether that is more informative.

What is Artificial Intelligence? Components

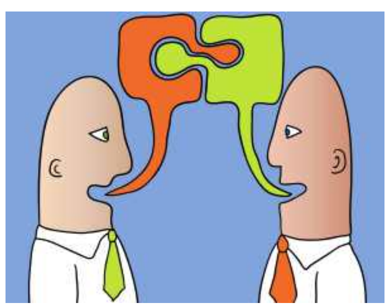
- **Elaine Rich:** AI studies how we can make the computer do things that humans can still do better at the moment.
- This needs a combination of

the ability to learn

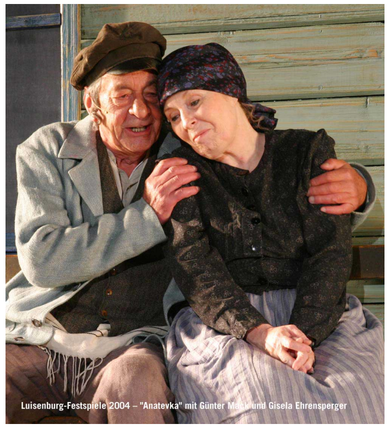
Inference

Perception

Language understanding



Emotion



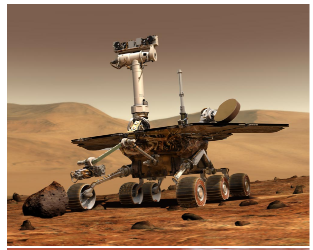
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG Dennis Müller: Artificial Intelligence 2 14 2024-04-14

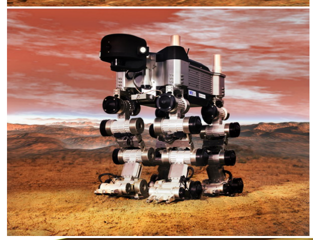
Note that list of components is controversial as well. Some say that it lumps together cognitive capacities that should be distinguished or forgets others, ... We state it here much more to get AI-2 students to think about the issues than to make it normative.

2.2 Artificial Intelligence is here today!

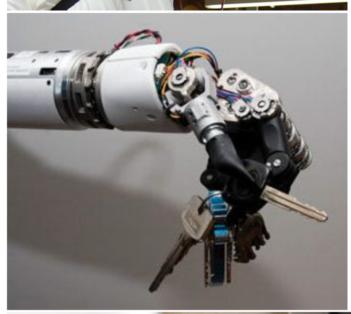
A Video Nugget covering this section can be found at `https://fau.tv/clip/id/21697`.
The components of Artificial Intelligence are quite daunting, and none of them are fully understood, much less achieved artificially. But for some tasks we can get by with much less. And indeed that is what the field of Artificial Intelligence does in practice – but keeps the lofty ideal around. This practice of “trying to achieve AI in selected and restricted domains” (cf. the discussion starting with slide **??**) has borne rich fruits: systems that meet or exceed human capabilities in such areas. Such systems are in common use in many domains of application.

Artificial Intelligence is here today!



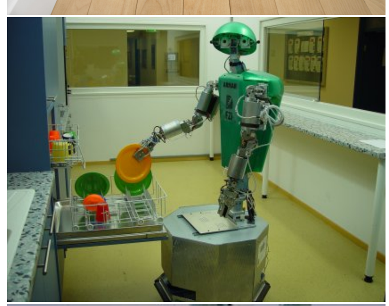


- ▷ in outer space
  - ▹ in outer space systems need autonomous control:
  - ▹ remote control impossible due to time lag



- ▷ in artificial limbs
  - ▹ the user controls the prosthesis via existing nerves, can e.g. grip a sheet of paper.

- ▷ in household appliances
  - ▹ The iRobot Roomba vacuums, mops, and sweeps in corners, . . . , parks, charges, and discharges.
  - ▹ general robotic household help is on the horizon.



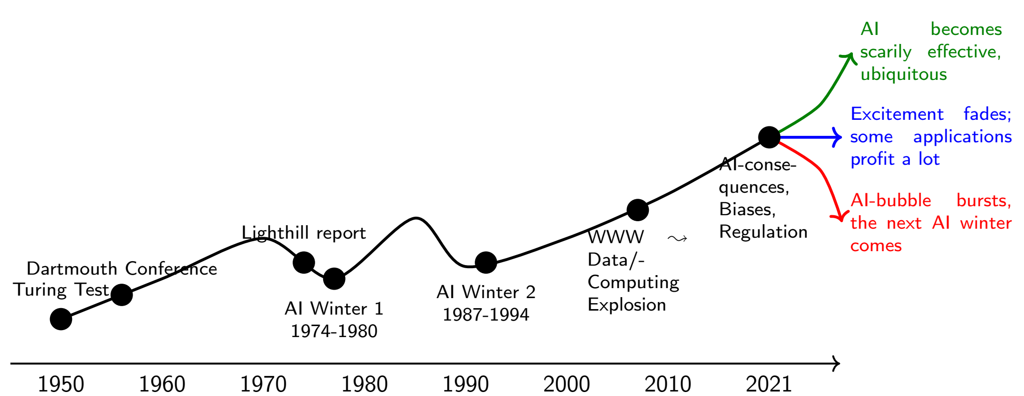
- ▷ in hospitals
  - ▹ in the USA 90% of the prostate operations are carried out by RoboDoc
  - ▹ Paro is a cuddly robot that eases solitude in nursing homes.

We will conclude this section with a note of caution.

The AI Conundrum

- ▷ **Observation:** Reserving the term "Artificial Intelligence" has been quite a land grab!
- ▷ **But:** researchers at the Dartmouth Conference (1956) really thought they would solve/reach AI in two/three decades.
- ▷ **Consequence:** AI still asks the big questions.
- ▷ **Another Consequence:** AI as a field is an incubator for many innovative technologies.
- ▷ **AI Conundrum:** Once AI solves a subfield it is called "computer science". (becomes a separate subfield of CS)
- ▷ **Example 2.2.1.** Functional/Logic Programming, automated theorem proving, Planning, machine learning, Knowledge Representation, ...
- ▷ **Still Consequence:** AI research was alternatingly flooded with money and cut off brutally.

The current AI Hype — Part of a longer Story



2.3 Ways to Attack the AI Problem

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/21717`.

There are currently three main avenues of attack to the problem of building artificially intelligent systems. The (historically) first is based on the symbolic representation of knowledge about the world and uses inference-based methods to derive new knowledge on which to base action decisions. The second uses statistical methods to deal with uncertainty about the world state and learning methods to derive new (uncertain) world assumptions to act on.

Four Main Approaches to Artificial Intelligence

- ▷ **Definition 2.3.1.** Symbolic AI is a subfield of AI based on the assumption that many aspects of intelligence can be achieved by the manipulation of symbols, combining them into meaning-carrying structures (expressions) and manipulating them (using processes) to produce new expressions.
- ▷ **Definition 2.3.2.** Statistical AI remedies the two shortcomings of symbolic AI approaches: that all concepts represented by symbols are crisply defined, and that all aspects of the world are knowable/representable in principle. Statistical AI adopts sophisticated mathematical models of uncertainty and uses them to create more accurate world models and reason about them.
- ▷ **Definition 2.3.3.** Subsymbolic AI (also called connectionism or neural AI) is a subfield of AI that posits that intelligence is inherently tied to brains, where information is represented by a simple sequence pulses that are processed in parallel via simple calculations realized by neurons, and thus concentrates on neural computing.
- ▷ **Definition 2.3.4.** Embodied AI posits that intelligence cannot be achieved by reasoning about the state of the world (symbolically, statistically, or connectivist), but must be embodied i.e. situated in the world, equipped with a "body" that can interact with it via sensors and actuators. Here, the main method for realizing intelligent behavior is by learning from the world.

As a consequence, the field of Artificial Intelligence (AI) is an engineering field at the intersection of computer science (logic, programming, applied statistics), cognitive science (psychology, neuroscience), philosophy (can machines think, what does that mean?), linguistics (natural language understanding), and mechatronics (robot hardware, sensors).
Subsymbolic AI and in particular machine learning is currently hyped to such an extent, that many people take it to be synonymous with "Artificial Intelligence". It is one of the goals of this course to show students that this is a very impoverished view.

Two ways of reaching Artificial Intelligence?

- ▷ We can classify the AI approaches by their coverage and the analysis depth (they are complementary)

Analysis ↑ vs. Coverage →	Narrow	Wide
Deep	symbolic AI-1	not there yet cooperation?
Shallow	no-one wants this	statistical/sub symbolic AI-2

- ▷ **This semester** we will cover foundational aspects of symbolic AI (deep/narrow processing)

▷ **next semester** concentrate on statistical/subsymbolic AI. (shallow/wide-coverage)

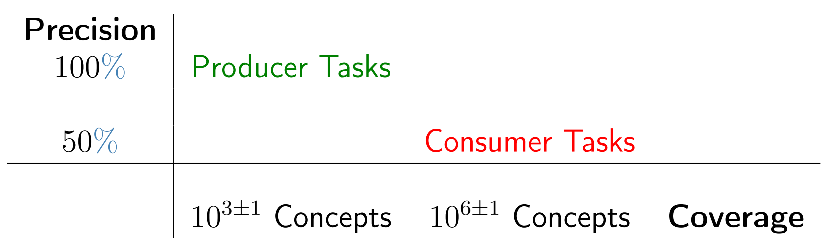
Dennis Müller: Artificial Intelligence 2 19 2024-04-14

We combine the topics in this way in this course, not only because this reproduces the historical development but also as the methods of statistical and subsymbolic AI share a common basis.
It is important to notice that all approaches to AI have their application domains and strong points. We will now see that exactly the two areas, where symbolic AI and statistical/subsymbolic AI have their respective fortes correspond to natural application areas.

Environmental Niches for both Approaches to AI

▷ **Observation:** There are two kinds of applications/tasks in AI

- ▷ Consumer tasks: consumer grade applications have tasks that must be fully generic and wide coverage. (e.g. machine translation like Google Translate)
- ▷ Producer tasks: producer grade applications must be high-precision, but can be domain-specific (e.g. multilingual documentation, machinery-control, program verification, medical technology)



▷ **General Rule:** Subsymbolic AI is well suited for consumer tasks, while symbolic AI is better suited for producer tasks.

▷ A domain of producer tasks I am interested in: mathematical/technical documents.

Dennis Müller: Artificial Intelligence 2 20 2024-04-14

An example of a producer task – indeed this is where the name comes from – is the case of a machine tool manufacturer T, which produces digitally programmed machine tools worth multiple million Euro and sells them into dozens of countries. Thus T must also comprehensive machine operation manuals, a non-trivial undertaking, since no two machines are identical and they must be translated into many languages, leading to hundreds of documents. As those manual share a lot of semantic content, their management should be supported by AI techniques. It is critical that these methods maintain a high precision, operation errors can easily lead to very costly machine damage and loss of production. On the other hand, the domain of these manuals is quite restricted. A machine tool has a couple of hundred components only that can be described by a comple of thousand attribute only.

Indeed companies like T employ high-precision AI techniques like the ones we will cover in this course successfully; they are just not so much in the public eye as the consumer tasks.

2.4 AI in the KWARC Group

The KWARC Research Group

- **Observation:** The ability to represent knowledge about the world and to draw logical inferences is one of the central components of intelligent behavior.
- **Thus:** reasoning components of some form are at the heart of many AI systems.
- **KWARC Angle:** Scaling up (web-coverage) without dumbing down (too much)
  - Content markup instead of full formalization (too tedious)
  - User support and quality control instead of "The Truth" (elusive anyway)
  - use Mathematics as a test tube (⚠ Mathematics $\widehat{=}$ Anything Formal ⚠)
  - care more about applications than about philosophy (we cannot help getting this right anyway as logicians)
- The KWARC group was established at Jacobs Univ. in 2004, moved to FAU Erlangen in 2016
- see `http://kwarc.info` for projects, publications, and links

Overview: KWARC Research and Projects

Applications: eMath 3.0, Active Documents, Active Learning, Semantic Spreadsheets/CAD/CAM, Change Mangagement, Global Digital Math Library, Math Search Systems, SMGloM: Semantic Multilingual Math Glossary, Serious Games, ...		
Foundations of Math: ▷ MathML, *OpenMath* ▷ advanced Type Theories ▷ MMT: Meta Meta Theory ▷ Logic Morphisms/Atlas ▷ Theorem Prover/CAS Interoperability ▷ Mathematical Models/Simulation	**KM & Interaction**: ▷ Semantic Interpretation (aka. Framing) ▷ math-literate interaction ▷ MathHub: math archives & active docs ▷ Active documents: embedded semantic services ▷ Model-based Education	**Semantization**: ▷ LaTeXML: LaTeX → XML ▷ sTeX: Semantic LaTeX ▷ invasive editors ▷ Context-Aware IDEs ▷ Mathematical Corpora ▷ Linguistics of Math ▷ ML for Math Semantics Extraction
Foundations: Computational Logic, Web Technologies, OMDoc/MMT		

Research Topics in the KWARC Group

- We are always looking for bright, motivated KWARCies.
- We have topics in for all levels! (Enthusiast, Bachelor, Master, Ph.D.)
- List of current topics: `https://gl.kwarc.info/kwarc/thesis-projects/`

- ▹ Automated Reasoning: Maths Representation in the Large
- ▹ Logics development, (Meta)^n-Frameworks
- ▹ Math Corpus Linguistics: Semantics Extraction
- ▹ Serious Games, Cognitive Engineering, Math Information Retrieval, Legal Reasoning, ...

▷ We always try to find a topic at the intersection of your and our interests.

▷ We also often have positions!. (HiWi, Ph.D.: $\frac{1}{2}$, PostDoc: full)

2.5 Agents and Environments in AI2

This part of the course notes addresses inference and agent decision making in partially observable environments, i.e. where we only know probabilities instead of certainties whether propositions are true/false. We cover basic probability theory and – based on that – Bayesian Networks and simple decision making in such environments. Finally we extend this to probabilistic temporal models and their decision theory.

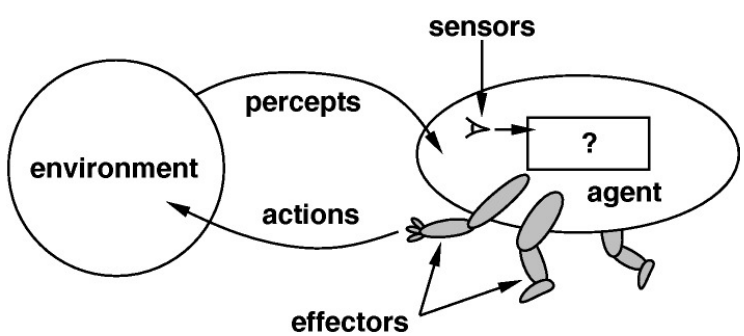
2.5.1 Recap: Rational Agents as a Conceptual Framework

A Video Nugget covering this subsection can be found at `https://fau.tv/clip/id/27585`.

Agents and Environments

▷ **Definition 2.5.1.** An agent is anything that

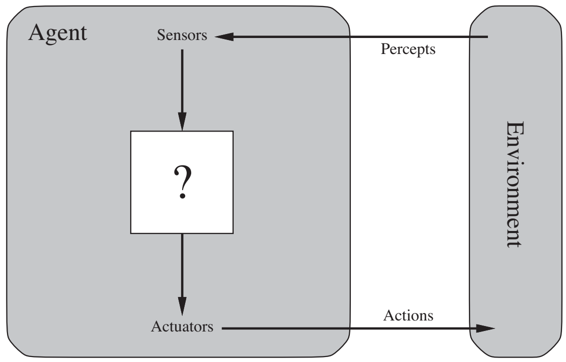
- ▹ perceives its environment via sensors (a means of sensing the environment)
- ▹ acts on it with actuators (means of changing the environment).



▷ **Example 2.5.2.** Agents include humans, robots, softbots, thermostats, etc.

Agent Schema: Visualizing the Internal Agent Structure

▷ **Agent Schema:** We will use the following kind of agent schema to visualize the internal structure of an agent:



Different agents differ on the contents of the white box in the center.

Rationality

- **Idea:** Try to design agents that are successful! (aka. "do the right thing")
- **Definition 2.5.3.** A performance measure is a function that evaluates a sequence of environments.
- **Example 2.5.4.** A performance measure for a vacuum cleaner could
  - award one point per "square" cleaned up in time T?
  - award one point per clean "square" per time step, minus one per move?
  - penalize for $> k$ dirty squares?
- **Definition 2.5.5.** An agent is called rational, if it chooses whichever action maximizes the expected value of the performance measure given the percept sequence to date.
- **Question:** Why is rationality a good quality to aim for?

Consequences of Rationality: Exploration, Learning, Autonomy

- **Note:** a rational agent need not be perfect
  - only needs to maximize expected value (rational $\neq$ omniscient)
    - need not predict e.g. very unlikely but catastrophic events in the future
  - percepts may not supply all relevant information (rational $\neq$ clairvoyant)
    - if we cannot perceive things we do not need to react to them.
    - but we may need to try to find out about hidden dangers (exploration)
  - action outcomes may not be as expected (rational $\neq$ successful)

- but we may need to take action to ensure that they do (more often) (learning)
- **Note:** rational ⇝ exploration, learning, autonomy
- **Definition 2.5.6.** An agent is called autonomous, if it does not rely on the prior knowledge about the environment of the designer.
- Autonomy avoids fixed behaviors that can become unsuccessful in a changing environment. (anything else would be irrational)
- The agent has to learn all relevant traits, invariants, properties of the environment and actions.

PEAS: Describing the Task Environment

- **Observation:** To design a rational agent, we must specify the task environment in terms of performance measure, environment, actuators, and sensors, together called the PEAS components.
- **Example 2.5.7.** When designing an automated taxi:
  - **Performance measure:** safety, destination, profits, legality, comfort, ...
  - **Environment:** US streets/freeways, traffic, pedestrians, weather, ...
  - **Actuators:** steering, accelerator, brake, horn, speaker/display, ...
  - **Sensors:** video, accelerometers, gauges, engine sensors, keyboard, GPS, ...
- **Example 2.5.8 (Internet Shopping Agent).** The task environment:
  - Performance measure: price, quality, appropriateness, efficiency
  - Environment: current and future WWW sites, vendors, shippers
  - Actuators: display to user, follow URL, fill in form
  - Sensors: HTML pages (text, graphics, scripts)

Environment types

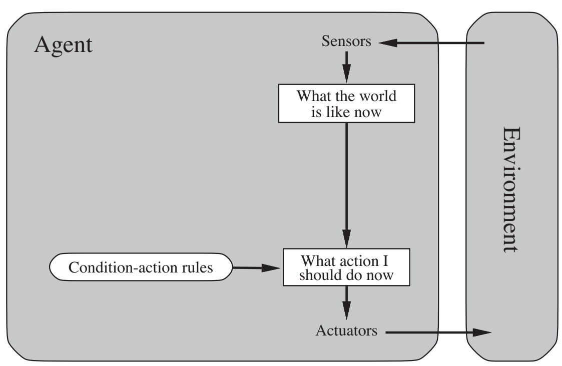
- **Observation 2.5.9.** *Agent design is largely determined by the type of environment it is intended for.*
- **Problem:** There is a vast number of possible kinds of environments in AI.
- **Solution:** Classify along a few "dimensions". (independent characteristics)
- **Definition 2.5.10.** For an agent a we classify the environment e of a by its type, which is one of the following. We call e

1. fully observable, iff the a's sensors give it access to the complete state of the environment at any point in time, else partially observable.

2. deterministic, iff the next state of the environment is completely determined by the current state and a's action, else stochastic.
3. episodic, iff a's experience is divided into atomic episodes, where it perceives and then performs a single action. Crucially, the next episode does not depend on previous ones. Non-episodic environments are called sequential.
4. dynamic, iff the environment can change without an action performed by a, else static. If the environment does not change but a's performance measure does, we call e semidynamic.
5. discrete, iff the sets of e's state and a's actions are countable, else continuous.
6. single agent, iff only a acts on e; else multi agent (when must we count parts of e as agents?)

Simple reflex agents

- **Definition 2.5.11.** A simple reflex agent is an agent a that only bases its actions on the last percept: so the agent function simplifies to $f_a : \mathcal{P} \to \mathcal{A}$.
- **Agent Schema:**

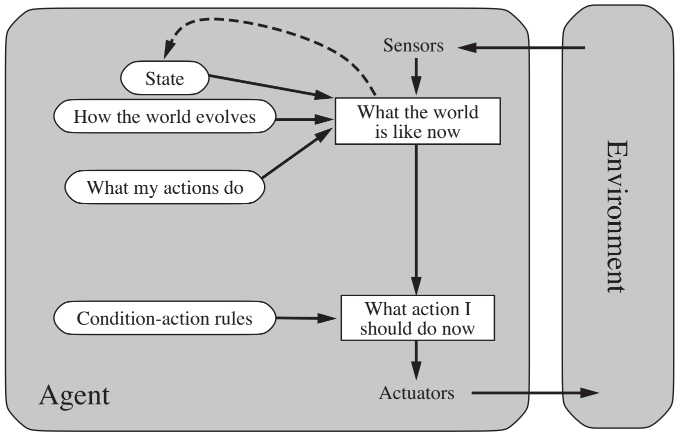


- **Example 2.5.12 (Agent Program).**

procedure Reflex−Vacuum−Agent [location,status] **returns** an action
  if status = Dirty **then** ...

Model-based Reflex Agents: Idea

- **Idea:** Keep track of the state of the world we cannot see in an internal model.
- **Agent Schema:**



Model-based Reflex Agents: Definition

- ▷ **Definition 2.5.13.** A model-based agent is an agent whose actions depend on
  - ▷ a world model: a set $\mathcal{S}$ of possible states.
  - ▷ a sensor model S that given a state s and a percepts p determines a new state $S(s,p)$.
  - ▷ a transition model T, that predicts a new state $T(s,a)$ from a state s and an action a.
  - ▷ An action function f that maps (new) states to an actions.

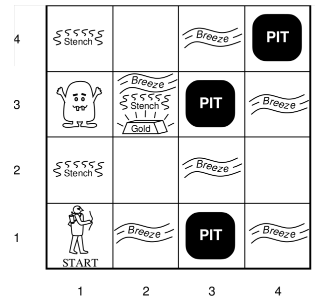
  If the world model of a model-based agent A is in state s and A has taken action a, A will transition to state $s' = T(S(p,s),a)$ and take action $a' = f(s')$.
- ▷ **Note:** As different percept sequences lead to different states, so the agent function $f_a : \mathcal{P}^* \rightarrow \mathcal{A}$ no longer depends only on the last percept.
- ▷ **Example 2.5.14 (Tail Lights Again).** Model-based agents can do the ?? if the states include a concept of tail light brightness.

2.5.2 Sources of Uncertainty

A Video Nugget covering this subsection can be found at `https://fau.tv/clip/id/27582`.

Sources of Uncertainty in Decision-Making

Where's that d... Wumpus?
And where am I, anyway??



- ▷ **Non-deterministic actions:**
  - ▷ "When I try to go forward in this dark cave, I might actually go forward-left or forward-right."
- ▷ **Partial observability with unreliable sensors:**
  - ▷ "Did I feel a breeze right now?";
  - ▷ "I think I might smell a Wumpus here, but I got a cold and my nose is blocked."
  - ▷ "According to the heat scanner, the Wumpus is probably in cell [2,3]."
- ▷ **Uncertainty about the domain behavior:**
  - ▷ "Are you *sure* the Wumpus never moves?"

Unreliable Sensors

- ▷ **Robot Localization:** Suppose we want to support localization using landmarks to narrow down the area.
- ▷ **Example 2.5.15.** *If you see the Eiffel tower, then you're in Paris.*
- ▷ **Difficulty:** Sensors can be imprecise.
  - ▷ Even if a landmark is perceived, we cannot conclude with certainty that the robot is at that location.
  - ▷ *This is the half-scale Las Vegas copy, you dummy.*
  - ▷ Even if a landmark is *not* perceived, we cannot conclude with certainty that the robot is *not* at that location.
  - ▷ *Top of Eiffel tower hidden in the clouds.*
- ▷ Only the probability of being at a location increases or decreases.

2.5.3 Agent Architectures based on Belief States

We are now ready to proceed to environments which can only partially observed and where are our actions are non deterministic. Both sources of uncertainty conspire to allow us only partial knowledge about the world, so that we can only optimize "expected utility" instead of "actual utility" of our actions.

World Models for Uncertainty

- ▷ **Problem:** We do not know with certainty what state the world is in!
- ▷ **Idea:** Just keep track of all the possible states it could be in.
- ▷ **Definition 2.5.16.** A model-based agent has a world model consisting of
  - ▷ a belief state that has information about the possible states the world may be in, and
  - ▷ a sensor model that updates the belief state based on sensor information
  - ▷ a transition model that updates the belief state based on actions.
- ▷ **Idea:** The agent environment determines what the world model can be.
- ▷ In a fully observable, deterministic environment,
  - ▷ we can observe the initial state and subsequent states are given by the actions alone.
  - ▷ thus the belief state is a singleton (we call its member the world state) and the transition model is a function from states and actions to states: a transition function.

That is exactly what we have been doing until now: we have been studying methods that build on descriptions of the "actual" world, and have been concentrating on the progression from atomic to factored and ultimately structured representations. Tellingly, we spoke of "world states" instead of "belief states"; we have now justified this practice in the brave new belief-based world models by the (re-) definition of "world states" above. To fortify our intuitions, let us recap from a belief-state-model perspective.

World Models by Agent Type in AI-1

- ▷ **Search-based Agents:** In a fully observable, deterministic environment
  - ▷ goal-based agent with world state $\widehat{=}$ "current state"
  - ▷ no inference. (goal $\widehat{=}$ goal state from search problem)
- ▷ **CSP-based Agents:** In a fully observable, deterministic environment
  - ▷ goal-based agent withworld state $\widehat{=}$ constraint network,
  - ▷ inference $\widehat{=}$ constraint propagation. (goal $\widehat{=}$ satisfying assignment)
- ▷ **Logic-based Agents:** In a fully observable, deterministic environment
  - ▷ model-based agent with world state $\widehat{=}$ logical formula
  - ▷ inference $\widehat{=}$ e.g. DPLL or resolution.
- ▷ **Planning Agents:** In a fully observable, deterministic, environment
  - ▷ goal-based agent with world state $\widehat{=}$ PL0, transition model $\widehat{=}$ STRIPS,
  - ▷ inference $\widehat{=}$ state/plan space search. (goal: complete plan/execution)

Let us now see what happens when we lift the restrictions of total observability and determinism.

World Models for Complex Environments

- In a fully observable, but stochastic environment,
  - the belief state must deal with a set of possible states.
  - ⇝ generalize the transition function to a transition relation.
- **Note:** This even applies to online problem solving, where we can just perceive the state. (e.g. when we want to optimize utility)
- In a deterministic, but partially observable environment,
  - the belief state must deal with a set of possible states.
  - we can use transition functions.
  - We need a sensor model, which predicts the influence of percepts on the belief state – during update.
- In a stochastic, partially observable environment,
  - mix the ideas from the last two. (sensor model + transition relation)

Preview: New World Models (Belief) ⇝ new Agent Types

- **Probabilistic Agents:** In a partially observable environment
  - belief state $\widehat{=}$ Bayesian networks,
  - inference $\widehat{=}$ probabilistic inference.
- **Decision-Theoretic Agents:** In a partially observable, stochastic environment
  - belief state + transition model $\widehat{=}$ decision networks,
  - inference $\widehat{=}$ maximizing expected utility.
- We will study them in detail this semester.

Overview: AI2

- Basics of probability theory (probability spaces, random variables, conditional probabilities, independence,...)
- Probabilistic reasoning: Computing the *a posteriori* probabilities of events given evidence, causal reasoning (Representing distributions efficiently, Bayesian networks,...)
- Probabilistic Reasoning over time (Markov chains, Hidden Markov models,...)

⇒ We can update our world model episodically based on observations (i.e. sensor data)

- ▷ Decision theory: Making decisions under uncertainty (Preferences, Utilities, Decision networks, Markov Decision Procedures,...)

⇒ We can choose the right action based on our world model and the likely outcomes of our actions

- ▷ Machine learning: Learning from data (Decision Trees, Classifiers, Neural Networks,...)

Part I

Reasoning with Uncertain Knowledge

This part of the course notes addresses inference and agent decision making in partially observable environments, i.e. where we only know probabilities instead of certainties whether propositions are true/false. We cover basic probability theory and – based on that – Bayesian Networks and simple decision making in such environments. Finally we extend this to probabilistic temporal models and their decision theory.

Chapter 3

Quantifying Uncertainty

3.1 Probability Theory

Probabilistic Models

- **Definition 3.1.1 (Mathematically (slightly simplified)).** A probability space or (probability model) is a pair $\langle \Omega, P \rangle$ such that:
  - Ω is a set of outcomes (called the sample space),
  - P is a function $\mathcal{P}(\Omega) \to [0,1]$, such that:
    - $P(\Omega) = 1$ and
    - $P(\bigcup_i A_i) = \sum_i P(A_i)$ for all pairwise disjoint $A_i \in \mathcal{P}(\Omega)$.

  P is called a probability measure.

  These properties are called the Kolmogorov axioms.

- **Intuition:** We run some experiment, the outcome of which is any $\omega \in \Omega$. $P(X)$ is the probability that the result of the experiment is *any one* of the outcomes in X. Naturally, the probability that *any* outcome occurs is 1 (hence $P(\Omega) = 1$). The probability of pairwise disjoint sets of outcomes should just be the sum of their probabilities.

- **Example 3.1.2 (Dice throws).** Assume we throw a (fair) die two times. Then the sample space is $\{(i,j) | 1 \leq i,j \leq 6\}$. We define P by letting $P(\{A\}) = \frac{1}{36}$ for every $A \in \Omega$.

  Since the probability of any outcome is the same, we say P is uniformly distributed

The definition is simplified in two places: Firstly, we assume that P is defined on the full power set. This is not always possible, especially if Ω is uncountable. In that case we need an additional set of "events" instead, and lots of mathematical machinery to make sure that we can safely take unions, intersections, complements etc. of these events.

Secondly, we would technically only demand that P is additive on countably many disjoint sets.

In this course we will assume that our sample space is at most countable anyway; usually even finite.

Random Variables

In practice, we are rarely interested in the *specific* outcome of an experiment, but rather in some *property* of the outcome. This is especially true in the very common situation where we don't even *know* the precise probabilities of the individual outcomes.

▷ **Example 3.1.3.** The probability that the *sum* of our two dice throws is 7 is $P(\{(i,j) \in \Omega | i+j=7\}) = P(\{(6,1),(1,6),(5,2),(2,5),(4,3),(3,4)\}) = \frac{6}{36} = \frac{1}{6}$.

▷ **Definition 3.1.4 (Again, slightly simplified).** Let D be a set. A random variable is a function $X\colon \Omega \to D$. We call D (somewhat confusingly) the domain of X, denoted $\mathrm{dom}(X)$.

For $x \in D$, we define the probability of x as $P(X=x) := P(\{\omega \in \Omega | X(\omega) = x\})$.

▷ **Definition 3.1.5.** We say that a random variable X is finite domain, iff its domain $\mathrm{dom}(X)$ is finite and Boolean, iff $\mathrm{dom}(X) = \{\mathsf{T}, \mathsf{F}\}$.

For a Boolean random variable, we will simply write $P(X)$ for $P(X = \mathsf{T})$ and $P(\neg X)$ for $P(X = \mathsf{F})$.

Note that a random variable, according to the formal definition, is *neither* random *nor* a variable: It is a function with clearly defined domain and codomain – and what we call the domain of the "variable" is actually its codomain... are you confused yet? ☺

This confusion is a side-effect of the *mathematical* formalism. In practice, a random variable is some indeterminate value that results from some statistical experiment – i.e. it is *random*, because the result is not predetermined, and it is a variable, because it can take on different values.

It just so happens that if we want to model this scenario *mathematically*, a function is the most natural way to do so.

Some Examples

▷ **Example 3.1.6.** Summing up our two dice throws is a random variable $S\colon \Omega \to [2,12]$ with $X((i,j)) = i+j$. The probability that they sum up to 7 is written as $P(S=7) = \frac{1}{6}$.

▷ **Example 3.1.7.** The first and second of our two dice throws are random variables $\mathrm{First}, \mathrm{Second}\colon \Omega \to [1,6]$ with $\mathrm{First}((i,j)) = i$ and $\mathrm{Second}((i,j)) = j$.

▷ *Remark 3.1.8.* Note, that the *identity* $\Omega \to \Omega$ is a random variable as well.

▷ **Example 3.1.9.** We can model $\mathrm{toothache}$, cavity and $\mathrm{gingivitis}$ as Boolean random variables, with the underlying probability space being...?? ¯_(ツ)_/¯

▷ **Example 3.1.10.** We can model tomorrow's weather as a random variable with domain $\{\mathtt{sunny}, \mathtt{rainy}, \mathtt{foggy}, \mathtt{warm}, \mathtt{cloudy}, \mathtt{humid}, \ldots\}$, with the underlying probability space being...?? ¯_(ツ)_/¯

⇒ This is why *probabilistic reasoning* is necessary: We can rarely reduce probabilistic scenarios down to clearly defined, fully known probability spaces and derive all the interesting things from there.

But: The definitions here allow us to *reason* about probabilities and random variables in a *mathematically* rigorous way, e.g. to make our intuitions and assumptions precise, and prove our methods to be *sound*.

Propositions

This is nice and all, but in practice we are interested in "compound" probabilities like:

"What is the probability that the sum of our two dice throws is 7, but neither of the two dice is a 3?"

Idea: Reuse the syntax of propositional logic and define the logical connectives for random variables!

Example 3.1.11. We can express the above as: $P(\neg(\text{First} = 3) \wedge \neg(\text{Second} = 3) \wedge (S = 7))$

Definition 3.1.12. Let X_1, X_2 be random variables, $x_1 \in \text{dom}(X_1)$ and $x_2 \in \text{dom}(X_2)$. We define:

1. $P(X_1 \neq x_1) := P(\neg(X_1 = x_1)) := P(\{\omega \in \Omega | X_1(\omega) \neq x_1\}) = 1 - P(X_1 = x_1)$.
2. $P((X_1 = x_1) \wedge (X_2 = x_2)) := P(\{\omega \in \Omega | (X_1(\omega) = x_1) \wedge (X_2(\omega) = x_2)\}) = P(\{\omega \in \Omega | X_1(\omega) = x_1\} \cap \{\omega \in \Omega | X_2(\omega) = x_2\})$.
3. $P((X_1 = x_1) \vee (X_2 = x_2)) := P(\{\omega \in \Omega | (X_1(\omega) = x_1) \vee (X_2(\omega) = x_2)\}) = P(\{\omega \in \Omega | X_1(\omega) = x_1\} \cup \{\omega \in \Omega | X_2(\omega) = x_2\})$.

It is also common to write $P(A, B)$ for $P(A \wedge B)$

Example 3.1.13. $P((\text{First} \neq 3) \wedge (\text{Second} \neq 3) \wedge (S = 7)) = P(\{(1,6),(6,1),(2,5),(5,2)\}) = \frac{1}{9}$

Events

Definition 3.1.14 (Again slightly simplified). Let $\langle\Omega, P\rangle$ be a probability space. An event is a subset of Ω.

Definition 3.1.15 (Convention). We call an event (by extension) anything that *represents* a subset of Ω: any statement formed from the logical connectives and values of random variables, on which $P(\cdot)$ is defined.

Problem 1.1

Remember: We can define $A \vee B := \neg(\neg A \wedge \neg B)$, T$:= A \vee \neg A$ and F$:= \neg$T – is this compatible with the definition of probabilities on propositional formulae? And why is $P(X_1 \neq x_1) = 1 - P(X_1 = x_1)$?

Problem 1.2 (Inclusion-Exclusion-Principle)

Show that $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$.

Problem 1.3

Show that $P(A) = P(A \wedge B) + P(A \wedge \neg B)$

Conditional Probabilities

- ▷ As we gather new information, our beliefs (*should*) change, and thus our probabilities!
- ▷ **Example 3.1.16.** Your "probability of missing the connection train" increases when you are informed that your current train has 30 minutes delay.
- ▷ **Example 3.1.17.** The "probability of cavity" increases when the doctor is informed that the patient has a toothache.
- ▷ **Example 3.1.18.** The probability that $S = 3$ is clearly higher if I know that $\mathrm{First} = 1$ than otherwise – or if I know that $\mathrm{First} = 6$!
- ▷ **Definition 3.1.19.** Let A and B be events where $P(B) \neq 0$. The conditional probability of A given B is defined as:

  $$P(A|B) := \frac{P(A \wedge B)}{P(B)}$$

  We also call $P(A)$ the prior probability of A, and $P(A|B)$ the posterior probability.
- ▷ **Intuition:** If we *assume* B to hold, then we are only interested in the "part" of Ω where A is true *relative to* B.

  Alternatively: We restrict our sample space Ω to the subset of outcomes where B holds. We then define a new probability space on this subset by scaling the probability measure so that it sums to 1 – which we do by dividing by $P(B)$. (We "*update our beliefs* based on new evidence")

Examples

- ▷ **Example 3.1.20.** If we assume $\mathrm{First} = 1$, then $P(S = 3|\mathrm{First} = 1)$ should be precisely $P(\mathrm{Second} = 2) = \frac{1}{6}$. We check:

  $$P(S = 3|\mathrm{First} = 1) = \frac{P((S = 3) \wedge (\mathrm{First} = 1))}{P(\mathrm{First} = 1)} = \frac{1/36}{1/6} = \frac{1}{6}$$

- ▷ **Example 3.1.21.** Assume the prior probability $P(\mathrm{cavity})$ is 0.122. The probability that a patient has both a cavity and a toothache is $P(\mathrm{cavity} \wedge \mathrm{toothache}) = 0.067$. The probability that a patient has a toothache is $P(\mathrm{toothache}) = 0.15$.

  If the patient complains about a toothache, we can update our estimation by computing the posterior probability:

  $$P(\mathrm{cavity}|\mathrm{toothache}) = \frac{P(\mathrm{cavity} \wedge \mathrm{toothache})}{P(\mathrm{toothache})} = \frac{0.067}{0.15} = 0.45.$$

- ▷ **Note:** We just computed the probability of some underlying *disease* based on the presence of a *symptom*!

  Or more generally: We computed the probability of a *cause* from observing its *effect*.

Some Rules

Equations on unconditional probabilities have direct analogues for conditional probabilities.

Problem 1.4

Convince yourself of the following:

- $P(A|C) = 1 - P(\neg A|C)$.
- $P(A|C) = P(A \wedge B|C) + P(A \wedge \neg B|C)$.
- $P(A \vee B|C) = P(A|C) + P(B|C) - P(A \wedge B|C)$.

But **not on the right hand side!**

Problem 1.5

Find *counterexamples* for the following (**false**) claims:

- $P(A|C) = 1 - P(A|\neg C)$
- $P(A|C) = P(A|B \wedge C) + P(A|B \wedge \neg C)$.
- $P(A|B \vee C) = P(A|B) + P(A|C) - P(A|B \wedge C)$.

Bayes' Rule

- **Note:** By definition, $P(A|B) = \frac{P(A \wedge B)}{P(B)}$. In practice, we often know the conditional probability already, and use it to compute the probability of the conjunction instead: $P(A \wedge B) = P(A|B) \cdot P(B) = P(B|A) \cdot P(A)$.

- **Theorem 3.1.22 (Bayes' Theorem).** *Given propositions A and B where $P(A) \neq 0$ and $P(B) \neq 0$, we have:*

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- *Proof:*

  1. $P(A|B) = \frac{P(A \wedge B)}{P(B)} = \frac{P(B|A) \cdot P(A)}{P(B)}$

  ...okay, that was straightforward... what's the big deal?

- **(Somewhat Dubious) Claim:** Bayes' Rule is the entire scientific method condensed into a single equation!

This is an extreme overstatement, but there is a grain of truth in it.

Bayes' Theorem - Why the Hype?

Say we have a *hypothesis* H about the world. (e.g. "The universe had a beginning")
We have *some prior belief* $P(H)$.
We gather *evidence* E. (e.g. "We observe a cosmic microwave background at 2.7K everywhere")

Bayes' Rule tells us how to *update our belief* in H based on H's ability to *predict* E (the *likelihood* $P(E|H)$) – *and*, importantly, the ability of *competing hypotheses* to predict the *same* evidence. (This is actually how scientific hypotheses should be evaluated)

$$\underbrace{P(H|E)}_{\text{posterior}} = \frac{P(E|H)\cdot P(H)}{P(E)} = \frac{\overbrace{P(E|H)}^{\text{likelihood}}\cdot\overbrace{P(H)}^{\text{prior}}}{\underbrace{P(E|H)}_{\text{likelihood}}\underbrace{P(H)}_{\text{prior}}+\underbrace{P(E|\neg H)P(\neg H)}_{\text{competition}}}$$

...if I keep gathering evidence and update, ultimately the impact of the prior belief will diminish.

"You're entitled to your own priors, but not your own likelihoods"

Independence

▷ **Question:** What is the probability that $S = 7$ *and* the patient has a toothache?

Or less contrived: What is the probability that the patient has a gingivitis *and* a cavity?

▷ **Definition 3.1.23.** Two events A and B are called independent, iff $P(A\wedge B) = P(A)\cdot P(B)$.

Two random variables X_1, X_2 are called independent, iff for all $x_1 \in \mathrm{dom}(X_1)$ and $x_2 \in \mathrm{dom}(X_2)$, the events $X_1 = x_1$ and $X_2 = x_2$ are independent.

We write $A \perp B$ or $X_1 \perp X_2$, respectively.

▷ **Theorem 3.1.24.** *Equivalently: Given events A and B with $P(B) \neq 0$, then A and B are independent iff $P(A|B) = P(A)$ (equivalently: $P(B|A) = P(B)$).*

▷ *Proof:*

1. ⇒ By definition, $P(A|B) = \frac{P(A\wedge B)}{P(B)} = \frac{P(A)\cdot P(B)}{P(B)} = P(A)$,
2. ⇐ Assume $P(A|B) = P(A)$. Then $P(A \wedge B) = P(A|B)\cdot P(B) = P(A)\cdot P(B)$.

▷ **Note:** Independence asserts that two events are *"not related"* – the probability of one does not depend on the other.

Mathematically, we can *determine* independence by checking whether $P(A \wedge B) = P(A)\cdot P(B)$.

In practice, this is impossible to check. Instead, we *assume* independence based on *domain knowledge*, and then *exploit* this to compute $P(A \wedge B)$.

Independence (Examples)

▷ **Example 3.1.25.**

▷ $\mathrm{First} = 2$ and $\mathrm{Second} = 3$ are independent – more generally, First and Second are independent (The outcome of the first die does not affect the outcome of the second die)

Quick check: $P((\mathrm{First} = a) \wedge (\mathrm{Second} = b)) = \frac{1}{36} = P(\mathrm{First} = a) \cdot P(\mathrm{Second} = b)$ ✓

▷ First and S are **not** independent. (The outcome of the first die affects the sum of the two dice.) Counterexample: $P((\mathrm{First} = 1) \wedge (S = 4)) = \frac{1}{36} \neq P(\mathrm{First} = 1) \cdot P(S = 4) = \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{72}$

▷ **But:** $P((\mathrm{First} = a) \wedge (S = 7)) = \frac{1}{36} = \frac{1}{6} \cdot \frac{1}{6} = P(\mathrm{First} = a) \cdot P(S = 7)$ – so the events $\mathrm{First} = a$ and $S = 7$ *are* independent. (Why?)

▷ **Example 3.1.26.**

▷ Are cavity and toothache independent?

...since cavities can cause a toothache, that would probably be a bad design decision...

▷ Are cavity and gingivitis independent? Cavities do not cause gingivitis, and gingivitis does not cause cavities, so... yes... right? (...as far as I know. I'm not a dentist.)

Probably not! A patient who has cavities has probably worse dental hygiene than those who don't, and is thus more likely to have gingivitis as well.

⇒ cavity may be *evidence* that raises the probabilty of gingivitis, even if they are not directly causally related.

Conditional Independence – Motivation

▷ A dentist can diagnose a cavity by using a *probe*, which may (or may not) *catch* in a cavity.

▷ Say we know from clinical studies that $P(\mathrm{cavity}) = 0.2$, $P(\mathrm{toothache}|\mathrm{cavity}) = 0.6$, $P(\mathrm{toothache}|\neg\mathrm{cavity}) = 0.1$, $P(\mathrm{catch}|\mathrm{cavity}) = 0.9$, and $P(\mathrm{catch}|\neg\mathrm{cavity}) = 0.2$.

▷ Assume the patient complains about a toothache, and our probe indeed catches in the aching tooth. What is the likelihood of having a cavity $P(\mathrm{cavity}|\mathrm{toothache} \wedge \mathrm{catch})$?

⇒ Use Bayes' rule:

$$P(\mathrm{cavity}|\mathrm{toothache} \wedge \mathrm{catch}) = \frac{P(\mathrm{toothache} \wedge \mathrm{catch}|\mathrm{cavity}) \cdot P(\mathrm{cavity})}{P(\mathrm{toothache} \wedge \mathrm{catch})}$$

▷ Note: $P(\mathrm{toothache} \wedge \mathrm{catch}) = P(\mathrm{toothache} \wedge \mathrm{catch}|\mathrm{cavity}) \cdot P(\mathrm{cavity}) + P(\mathrm{toothache} \wedge \mathrm{catch}|\neg\mathrm{cavity}) \cdot P(\neg\mathrm{cavity})$

⇒ Now we're only missing $P(\mathrm{toothache} \wedge \mathrm{catch}|\mathrm{cavity} = b)$ for $b \in \{\mathsf{T}, \mathsf{F}\}$.

... Now what?

▷ Are toothache and catch independent, maybe? **No**: Both have a common (possible) cause, cavity.

Also, there's this pesky $P(\cdot|\mathrm{cavity})$ in the way. . . ...wait a minute...

Conditional Independence – Definition

▷ *Assuming* the patient has (or does not have) a cavity, the events toothache and catch are independent: Both are caused by a cavity, but they don't influence each other otherwise.

i.e. cavity "contains all the information" that links toothache and catch in the first place.

▷ **Definition 3.1.27.** Given events A, B, C with $P(C) \neq 0$, then A and B are called conditionally independent given C, iff $P(A \wedge B|C) = P(A|C) \cdot P(B|C)$.

Equivalently: iff $P(A|B \wedge C) = P(A|C)$, or $P(B|A \wedge C) = P(B|C)$.

Let Y be a random variable. We call two random variables X_1, X_2 conditionally independent given Y, iff for all $x_1 \in \mathbf{dom}(X_1)$, $x_2 \in \mathbf{dom}(X_2)$ and $y \in \mathbf{dom}(Y)$, the events $X_1 = x_1$ and $X_2 = x_2$ are conditionally independent given $Y = y$.

▷ **Example 3.1.28.** Let's assume toothache and catch are conditionally independent given cavity/$\neg$cavity. Then we can finally compute:

$$P(\text{cavity}|\text{toothache} \wedge \text{catch}) = \frac{P(\text{toothache}\wedge\text{catch}|\text{cavity}) \cdot P(\text{cavity})}{P(\text{toothache}\wedge\text{catch})}$$

$$= \frac{P(\text{toothache}|\text{cavity}) \cdot P(\text{catch}|\text{cavity}) \cdot P(\text{cavity})}{P(\text{toothache}|\text{cavity}) \cdot P(\text{catch}|\text{cavity}) \cdot P(\text{cavity}) + P(\text{toothache}|\neg\text{cavity}) \cdot P(\text{catch}|\neg\text{cavity}) \cdot P(\neg\text{cavity})} = \frac{0.6 \cdot 0.9 \cdot 0.2}{0.6 \cdot 0.9 \cdot 0.2 + 0.1 \cdot 0.2 \cdot 0.8} = 0.87$$

Conditional Independence

▷ **Lemma 3.1.29.** *If A and B are conditionally independent given C, then $P(A|B \wedge C) = P(A|C)$*

Proof:

$$P(A|B \wedge C) = \frac{P(A \wedge B \wedge C)}{P(B \wedge C)} = \frac{P(A \wedge B|C) \cdot P(C)}{P(B \wedge C)} = \frac{P(A|C) \cdot P(B|C) \cdot P(C)}{P(B \wedge C)} = \frac{P(A|C) \cdot P(B \wedge C)}{P(B \wedge C)} = P(A|C)$$

▷ **Question:** If A and B are conditionally independent given C, does this imply that A and B are independent? **No.** See previous slides for a counterexample.

▷ **Question:** If A and B are independent, does this imply that A and B are also conditionally independent given C? **No.** For example: First and Second are independent, but not conditionally independent given $S = 4$.

▷ **Question:** Okay, so what if A, B and C are *all* pairwise independent? Are A and B conditionally independent given C *now*? **Still no.** Remember: First $= a$, Second $= b$ and $S = 7$ are all independent, but First and Second are not conditionally independent given $S = 7$.

▷ **Question:** When can we infer conditional independence from a "more general" notion of independence?

We need *mutual independence*. Roughly: A set of events is called *mutually* independent, if every event is independent from *any conjunction of the others*. (Not really relevant for this course though)

Summary

- ▷ Probability spaces serve as a mathematical model (and hence justification) for everything related to probabilities.
- ▷ The "atoms" of any statement of probability are the random variables. (Important special cases: Boolean and finite domain)
- ▷ We can define probabilities on compund (propositional logical) statements, with (outcomes of) random variables as "propositional variables".
- ▷ Conditional probabilities represent *posterior probabilities* given some observed outcomes.
- ▷ independence and conditional independence are strong assumptions that allow us to simplify computations of probabilities
- ▷ Bayes' Theorem

So much about the math...

We now have a mathematical setup for probabilities.

But: The math does not tell us what probabilities *are*:
Assume we can mathematically derive this to be the case: *the probability of rain tomorrow is* 0.3. What does this even *mean*?

- ▷ **Frequentist:** The probability of an event is the limit of its relative frequency in a large number of trials.

  In other words: "In 30% of the cases where we have similar weather conditions, it rained the next day."

Objection: Okay, but what about *unique* events? "The probability of me passing the exam is 80%" – does this mean anything, if I only take the exam once? Am I comparable to "similar students"? What counts as sufficiently "similar"?

- ▷ **Bayesian:** Probabilities are *degrees of belief*. It means you **should** be 30% confident that it will rain tomorrow.

Objection: And why *should* I? Is this not purely *subjective* then?

Pragmatics

Pragmatically, both interpretations amount to the same thing: I should *act as if* I'm 30% confident that it will rain tomorrow. (Whether by fiat, or because in 30% of comparable cases, it rained.)

Objection: Still: why should I? And why should my beliefs follow the seemingly arbitrary

Kolmogorov axioms?

- ▷ [DF31]: If an agent has a belief that violates the Kolmogorov axioms, then there exists a combination of "bets" on propositions so that the agent *always* loses money.
- ▷ **In other words:** If your beliefs are not consistent with the mathematics, and you *act in accordance with your beliefs*, there is a way to exploit this inconsistency to your disadvantage.
- ▷ ...and, more importantly, your AI agents! ☺

3.2 Probabilistic Reasoning Techniques

Okay, now how do I implement this?

This is a computer science course. We need to implement this stuff.

Do we... implement random variables as functions? Is a probability space a... class maybe?

No. As mentioned, we rarely know the probability space entirely. Instead we will use probability distributions, which are just arrays (of arrays of...) of probabilities.

And then we represent *those* are sparse as possible, by exploiting independence, conditional independence, ...

Probability Distributions

- ▷ **Definition 3.2.1.** The probability distribution for a random variable X, written $\mathbb{P}(X)$, is the vector of probabilities for the (ordered) domain of X.
- ▷ **Note:** The values in a probability distribution are all positive and sum to 1. (Why?)
- ▷ **Example 3.2.2.** $\mathbb{P}(\mathrm{First}) = \mathbb{P}(\mathrm{Second}) = \langle\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6}\rangle$. (Both First and Second are uniformly distributed)
- ▷ **Example 3.2.3.** The probability distribution $\mathbb{P}(S)$ is $\langle\frac{1}{36},\frac{1}{18},\frac{1}{12},\frac{1}{9},\frac{5}{36},\frac{1}{6},\frac{5}{36},\frac{1}{9},\frac{1}{12},\frac{1}{18},\frac{1}{36}\rangle$. Note the symmetry, with a "peak" at 7 – the random variable is (*approximately*, because our domain is discrete rather than continuous) normally distributed (or gaussian distributed, or follows a bell-curve,...).
- ▷ **Example 3.2.4.** Probability distributions for Boolean random variables are naturally *pairs* (probabilities for T and F), e.g.:

$$\mathbb{P}(\mathrm{toothache}) = \langle 0.15, 0.85\rangle$$
$$\mathbb{P}(\mathrm{cavity}) = \langle 0.122, 0.878\rangle$$

- ▷ More generally:

  Definition 3.2.5. A probability distribution is a vector $\mathbf{v}$ of values $\mathbf{v}_i \in [0,1]$ such that $\sum_i \mathbf{v}_i = 1$.

The Full Joint Probability Distribution

- **Definition 3.2.6.** Given random variables $X_1, \ldots, X_n$, the full joint probability distribution, denoted $\mathbb{P}(X_1, \ldots, X_n)$, is the n-dimensional array of size $|D_1 \times \ldots \times D_n|$ that lists the probabilities of all conjunctions of values of the random variables.
- **Example 3.2.7.** $\mathbb{P}(\text{cavity}, \text{toothache}, \text{gingivitis})$ could look something like this:

	toothache		¬toothache	
	gingivitis	¬gingivitis	gingivitis	¬gingivitis
cavity	0.007	0.06	0.005	0.05
¬cavity	0.08	0.003	0.045	0.75

- **Example 3.2.8.** $\mathbb{P}(\text{First}, S)$

First \ S	2	3	4	5	6	7	8	9	10	11	12
1	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	0	0	0	0	0
2	0	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	0	0	0	0
3	0	0	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	0	0	0
4	0	0	0	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	0	0
5	0	0	0	0	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	0
6	0	0	0	0	0	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$	$\frac{1}{36}$

Note that if we know the value of First, the value of S is completely determined by the value of Second.

Conditional Probability Distributions

- **Definition 3.2.9.** Given random variables X and Y, the conditional probability distribution of X given Y, written $\mathbb{P}(X|Y)$ is the table of all conditional probabilities of values of X given values of Y.
- For sets of variables analogously: $\mathbb{P}(X_1, \ldots, X_n | Y_1, \ldots, Y_m)$.
- **Example 3.2.10.** $\mathbb{P}(\text{cavity}|\text{toothache})$:

	toothache	¬toothache
cavity	$P(\text{cavity}\mid\text{toothache}) = 0.45$	$P(\text{cavity}\mid\neg\text{toothache}) = 0.065$
¬cavity	$P(\neg\text{cavity}\mid\text{toothache}) = 0.55$	$P(\neg\text{cavity}\mid\neg\text{toothache}) = 0.935$

- **Example 3.2.11.** $\mathbb{P}(\text{First}|S)$

First \ S	2	3	4	5	6	7	8	9	10	11	12
1	1	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	0	0	0	0	0
2	0	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{5}$	0	0	0	0
3	0	0	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{5}$	$\frac{1}{4}$	0	0	0
4	0	0	0	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{5}$	$\frac{1}{4}$	$\frac{1}{3}$	0	0
5	0	0	0	0	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{5}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{2}$	0
6	0	0	0	0	0	$\frac{1}{6}$	$\frac{1}{5}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{2}$	1

▷ **Note:** Every "column" of a conditional probability distribution is itself a probability distribution.
(Why?)

Convention

We now "lift" multiplication and division to the level of whole probability distributions:

▷ **Definition 3.2.12.** Whenever we use $\mathbb{P}$ in an equation, we take this to mean a *system of equations*, for each value in the domains of the random variables involved.

Example 3.2.13.

▷ $\mathbb{P}(X,Y) = \mathbb{P}(X|Y) \cdot \mathbb{P}(Y)$ represents the system of equations $P(X = x \wedge Y = y) = P(X = x|Y = y) \cdot P(Y = y)$ for all x, y in the respective domains.

▷ $\mathbb{P}(X|Y){:=}\frac{\mathbb{P}(X,Y)}{\mathbb{P}(Y)}$ represents the system of equations $P(X = x|Y = y){:=}\frac{P((X=x) \wedge (Y=y))}{P(Y=y)}$

▷ Bayes' Theorem: $\mathbb{P}(X|Y) = \frac{\mathbb{P}(Y|X) \cdot \mathbb{P}(X)}{\mathbb{P}(Y)}$ represents the system of equations $P(X = x|Y = y) = \frac{P(Y=y|X=x) \cdot P(X=x)}{P(Y=y)}$

So, what's the point?

▷ Obviously, the probability distribution contains all the information about a specific random variable we need.

▷ **Observation:** The full joint probability distribution of variables $X_1, \ldots, X_n$ contains *all* the information about the random variables *and their conjunctions* we need.

▷ **Example 3.2.14.** We can read off the probability $P(\text{toothache})$ from the full joint probability distribution as $0.007 + 0.06 + 0.08 + 0.003 = 0.15$, and the probability $P(\text{toothache} \wedge \text{cavity})$ as $0.007 + 0.06 = 0.067$

▷ We can actually implement this! (They're just (nested) arrays)

But just as we often don't have a fully specified probability space to work in, we often don't have a full joint probability distribution for our random variables either.

▷ Also: Given random variables $X_1, \ldots, X_n$, the full joint probability distribution has $\prod_{i=1}^{n} |\text{dom}(X_i)|$ entries! ($\mathbb{P}(\text{First}, S)$ already has 60 entries!)

⇒ The rest of this section deals with keeping things small, by *computing* probabilities instead of *storing* them all.

Probabilistic Reasoning

- **Probabilistic reasoning** refers to inferring probabilities of events from the probabilities of other events

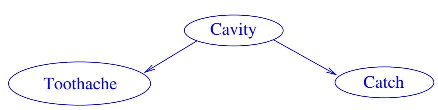
  as opposed to determining the probabilities e.g. *empirically*, by gathering (sufficient amounts of *representative*) data and counting.

- **Note:** In practice, we are *primarily* interested in, and have access to, conditional probabilities rather than the unconditional probabilities of conjunctions of events:
  - We don't reason in a vacuum: Usually, we have some evidence and want to infer the posterior probability of some related event. (e.g. infer a plausible *cause* given some *symptom*)

    ⇒ we are interested in the conditional probability $P(\text{hypothesis}|\text{observation})$.
  - *"80% of patients with a cavity complain about a toothache"* (i.e. $P(\text{toothache}|\text{cavity})$) is more the kind of data people actually collect and publish than *"1.2% of the general population have both a cavity and a toothache"* (i.e. $P(\text{cavity} \wedge \text{toothache})$).
  - Consider the probe catching in a cavity. The probe is a diagnostic tool, which is usually evaluated in terms of its *sensitivity* $P(\text{catch}|\text{cavity})$ and *specificity* $P(\neg\text{catch}|\neg\text{cavity})$. (You have probably heard these words a lot since 2020...)

Naive Bayes Models

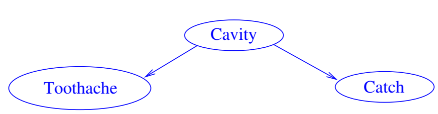
Consider again the dentistry example with random variables cavity, toothache, and catch. We assume cavity **causes** both toothache and catch, and that toothache and catch are conditionally independent given cavity:



We likely know the *sensitivity* $P(\text{catch}|\text{cavity})$ and *specificity* $P(\neg\text{catch}|\neg\text{cavity})$, which jointly give us $\mathbb{P}(\text{catch}|\text{cavity})$, and from medical studies, we should be able to determine $P(\text{cavity})$ (the *prevalence* of cavities in the population) and $\mathbb{P}(\text{toothache}|\text{cavity})$.

This kind of situation is surprisingly common, and deserves a name

Naive Bayes Models



Definition 3.2.15. A naive Bayes model (or, less accurately, Bayesian classifier, or, derogatorily, idiot Bayes model) consists of:

1. random variables $C, E_1, \ldots, E_n$ such that all the $E_1, \ldots, E_n$ are conditionally independent given C,
2. the probability distribution $\mathbb{P}(C)$, and
3. the conditional probability distributions $\mathbb{P}(E_i|C)$.

We call C the cause and the $E_1, \ldots, E_n$ the effects of the model.

Convention: Whenever we draw a graph of random variables, we take the arrows to connect *causes* to their direct *effects*, and assert that unconnected nodes are conditionally independent given all their ancestors. We will make this more precise later.

Can we compute the full joint probability distribution $\mathbb{P}(\text{cavity}, \text{toothache}, \text{catch})$ from this information?

Recovering the Full Joint Probability Distribution

▷ **Lemma 3.2.16 (Product rule).** $\mathbb{P}(X, Y) = \mathbb{P}(X|Y) \cdot \mathbb{P}(Y)$.

We can generalize this to more than two variables, by repeatedly applying the product rule:

▷ **Lemma 3.2.17 (Chain rule).** *For any sequence of random variables $X_1, \ldots, X_n$:*

$$\mathbb{P}(X_1, \ldots, X_n) = \mathbb{P}(X_1|X_2, \ldots, X_n) \cdot \mathbb{P}(X_2|X_3, \ldots X_n) \cdot \ldots \cdot \mathbb{P}(X_{n-1}|X_n) \cdot P(X_n)$$

.

Hence:

▷ **Theorem 3.2.18.** *Given a naive Bayes model with effects $E_1, \ldots, E_n$ and cause C, we have*

$$\mathbb{P}(C, E_1, \ldots, E_n) = \mathbb{P}(C) \cdot \prod_{i=1}^{n} \mathbb{P}(E_i|C).$$

Proof: Using the chain rule:

1. $\mathbb{P}(E_1, \ldots, E_n, C) = \mathbb{P}(E_1|E_2, \ldots, E_n, C) \cdot \ldots \cdot \mathbb{P}(E_n|C) \cdot \mathbb{P}(C)$
2. Since all the E_i are conditionally independent, we can drop them on the right hand sides of the $\mathbb{P}(E_j|..., C)$

Marginalization

Great, so now we can compute $\mathbb{P}(C|E_1, \ldots, E_n) = \frac{\mathbb{P}(C, E_1, \ldots, E_n)}{\mathbb{P}(E_1, \ldots, E_n)}$...
...except that we don't know $\mathbb{P}(E_1, \ldots, E_n)$:-/
...except that we can compute the full joint probability distribution, so we can recover it:

Lemma 3.2.19 (Marginalization). *Given random variables $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_m$, we have $\mathbb{P}(X_1, \ldots, X_n) = \sum_{y_1 \in \mathrm{dom}(Y_1), \ldots, y_m \in \mathrm{dom}(Y_m)} \mathbb{P}(X_1, \ldots, X_n, Y_1 = y_1, \ldots, Y_m = y_m)$.*

(This is just a fancy way of saying "we can add the relevant entries of the full joint probability distribution")

Example 3.2.20. Say we observed toothache = T and catch = T. Using marginalization, we can compute

$$P(\text{cavity}|\text{toothache} \wedge \text{catch}) = \frac{P(\text{cavity} \wedge \text{toothache} \wedge \text{catch})}{P(\text{toothache} \wedge \text{catch})}$$
$$= \frac{P(\text{cavity} \wedge \text{toothache} \wedge \text{catch})}{\sum_{c \in \{\text{cavity}, \neg\text{cavity}\}} P(c \wedge \text{toothache} \wedge \text{catch})}$$
$$= \frac{P(\text{cavity}) \cdot P(\text{toothache}|\text{cavity}) \cdot P(\text{catch}|\text{cavity})}{\sum_{c \in \{\text{cavity}, \neg\text{cavity}\}} P(c) \cdot P(\text{toothache}|c) \cdot P(\text{catch}|c)}$$

Unknowns

What if we don't know catch? (I'm not a dentist, I don't have a probe...)

We split our effects into $\{E_1, \ldots, E_n\} = \{O_1, \ldots, O_{n_O}\} \cup \{U_1, \ldots, U_{n_U}\}$ – the *observed* and *unknown* random variables.

Let $D_U := \text{dom}(U_1) \times \ldots \times \text{dom}(U_{n_u})$. Then

$$\mathbb{P}(C|O_1, \ldots, O_{n_O}) = \frac{\mathbb{P}(C, O_1, \ldots, O_{n_O})}{\mathbb{P}(O_1, \ldots, O_{n_O})}$$
$$= \frac{\sum_{u \in D_U} \mathbb{P}(C, O_1, \ldots, O_{n_O}, U_1 = u_1, \ldots, U_{n_u} = u_{n_u})}{\sum_{c \in \text{dom}(C)} \sum_{u \in D_U} \mathbb{P}(O_1, \ldots, O_{n_O}, C = c, U_1 = u_1, \ldots, U_{n_u} = u_{n_u})}$$
$$= \frac{\sum_{u \in D_U} \mathbb{P}(C) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C) \cdot \prod_{j=1}^{n_U} \mathbb{P}(U_j = u_j|C)}{\sum_{c \in \text{dom}(C)} \sum_{u \in D_U} P(C = c) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C = c) \cdot \prod_{j=1}^{n_U} P(U_j = u_j|C = c)}$$
$$= \frac{\mathbb{P}(C) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C) \cdot (\sum_{u \in D_U} \prod_{j=1}^{n_U} \mathbb{P}(U_j = u_j|C))}{\sum_{c \in \text{dom}(C)} P(C = c) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C = c) \cdot (\sum_{u \in D_U} \prod_{j=1}^{n_U} P(U_j = u_j|C = c))}$$

...oof...

Unknowns

$$\mathbb{P}(C|O_1, \ldots, O_{n_O}) = \frac{\mathbb{P}(C) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C) \cdot (\sum_{u \in D_U} \prod_{j=1}^{n_U} \mathbb{P}(U_j = u_j|C))}{\sum_{c \in \text{dom}(C)} P(C = c) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C = c) \cdot (\sum_{u \in D_U} \prod_{j=1}^{n_U} P(U_j = u_j|C = c))}$$

First, note that $\sum_{u \in D_U} \prod_{j=1}^{n_U} P(U_j = u_j|C = c) = 1$ (We're summing over all possible events on the (conditionally independent) $U_1, \ldots, U_{n_U}$ given $C = c$)

$$\mathbb{P}(C|O_1, \ldots, O_{n_O}) = \frac{\mathbb{P}(C) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C)}{\sum_{c \in \text{dom}(C)} P(C = c) \cdot \prod_{i=1}^{n_O} \mathbb{P}(O_i|C = c)}$$

Secondly, note that the *denominator* is

1. the same for any given observations $O_1, \ldots, O_{n_O}$, independent of the value of C, and
2. the *sum* over all the *numerators* in the full distribution.

That is: The denominator only serves to *scale* what is *almost* already the distribution $\mathbb{P}(C|O_1,\ldots,O_{n_O})$ to sum up to 1.

Normalization

Definition 3.2.21 (Normalization). Given a vector $w{:=}\langle w_1,\ldots,w_k\rangle$ of numbers in $[0,1]$ where $\sum_{i=1}^{k} w_i \leq 1$.

Then the normalized vector $\alpha(w)$ is defined (component-wise) as

$$(\alpha(w))_i{:=}\frac{w_i}{\sum_{j=1}^{k} w_j}.$$

Note that $\sum_{i=1}^{k} \alpha(w)_i = 1$, i.e. $\alpha(w)$ is a probability distribution.

This finally gives us:

Theorem 3.2.22 (Inference in a Naive Bayes model). *Let $C, E_1,\ldots,E_n$ a naive Bayes model and $E_1,\ldots,E_n = O_1,\ldots,O_{n_O},U_1,\ldots,U_{n_U}$.*

Then

$$\mathbb{P}(C|O_1 = o_1,\ldots,O_{n_O} = o_{n_O}) = \alpha(\mathbb{P}(C)\cdot\prod_{i=1}^{n_O}\mathbb{P}(O_i = o_i|C))$$

Note, that this is entirely independent of the *unknown* random variables $U_1,\ldots,U_{n_U}$!

Also, note that this is just a fancy way of saying "first, compute all the numerators, then divide all of them by their sums".

Dentistry Example

Putting things together, we get:

$$\begin{aligned}\mathbb{P}(\text{cavity}|\text{toothache} = \mathsf{T}) &= \alpha(\mathbb{P}(\text{cavity})\cdot\mathbb{P}(\text{toothache} = \mathsf{T}|\text{cavity}))\\ &= \alpha(\langle P(\text{cavity})\cdot P(\text{toothache}|\text{cavity}), P(\neg\text{cavity})\cdot P(\text{toothache}|\neg\text{cavity})\rangle)\end{aligned}$$

Say we have $P(\text{cavity}) = 0.1$, $P(\text{toothache}|\text{cavity}) = 0.8$, and $P(\text{toothache}|\neg\text{cavity}) = 0.05$. Then

$$\mathbb{P}(\text{cavity}|\text{toothache} = \mathsf{T}) = \alpha(\langle 0.1\cdot 0.8, 0.9\cdot 0.05\rangle) = \alpha(\langle 0.08, 0.045\rangle)$$

$0.08 + 0.045 = 0.125$, hence

$$\mathbb{P}(\text{cavity}|\text{toothache} = \mathsf{T}) = \langle\frac{0.08}{0.125},\frac{0.045}{0.125}\rangle = \langle 0.64, 0.36\rangle$$

Naive Bayes Classification

We can use a naive Bayes model as a very simple *classifier*:

▷ Assume we want to classify newspaper articles as one of the categories *politics*, *sports*, *business*, *fluff*, etc. based on the words they contain.

▷ Given a large set of articles, we can determine the relevant probabilities by counting the occurrences of the categories $\mathbb{P}(\text{category})$, and of words per category – i.e. $\mathbb{P}(\text{word}_i|\text{category})$ for some (huge) list of words $(\text{word}_i)_{i=1}^n$.

▷ We assume that the occurrence of each word is conditionally independent of the occurrence of any other word given the category of the document. (This assumption is clearly wrong, but it makes the model simple and often works well in practice.) (⇒ "Idiot Bayes model")

▷ Given a new article, we just count the occurrences k_i of the words in it and compute

$$\mathbb{P}(\text{category}|\text{word}_1 = k_1, \ldots, \text{word}_n = k_n) = \alpha(\mathbb{P}(\text{category}) \cdot \prod_{i=1}^{n} \mathbb{P}(\text{word}_i = k_i|\text{category}))$$

▷ We then choose the category with the highest probability.

Inference by Enumeration

The rules we established for naive Bayes models, i.e. Bayes's theorem, the product rule and chain rule, marginalization and normalization, are *general* techniques for probabilistic reasoning, and their usefulness is not limited to the naive Bayes models.

More generally:

Theorem 3.2.23. *Let* $Q, E_1, \ldots, E_{n_E}, U_1, \ldots, U_{n_U}$ *be random variables and* $D{:=}\text{dom}(U_1) \times \ldots \times \text{dom}(U_{n_U})$. *Then*

$$\mathbb{P}(Q|E_1 = e_1, \ldots, E_{n_E} = e_{n_e}) = \alpha(\sum_{u \in D} \mathbb{P}(Q, E_1 = e_1, \ldots, E_{n_E} = e_{n_e}, U_1 = u_1, \ldots, U_{n_U} = u_{n_U}))$$

.

We call Q *the* ***query variable****,* $E_1, \ldots, E_{n_E}$ *the* ***evidence****, and* $U_1, \ldots, U_{n_U}$ *the* ***unknown*** *(or* ***hidden****)* ***variables****, and computing a conditional probability this way* ***enumeration****.*

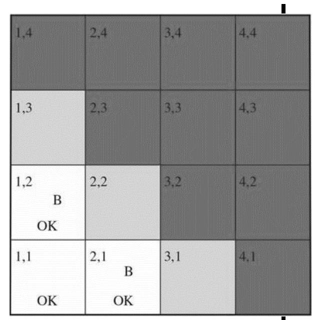
Note that this is just a "mathy" way of saying we

1. sum over all relevant entries of the full joint probability distribution of the variables, and
2. normalize the result to yield a probability distribution.

We will fortify our intuition about naive Bayes models with a variant of the Wumpus world we looked at **??** to understand whether logic was up to the job of guiding an agent in the Wumpus cave.

Example: The Wumpus is Back

- ▷ We have a maze where
  - ▷ Every cell except $[1,1]$ possibly contains a *pit*, with 20% probability.
  - ▷ pits cause a *breeze* in neighboring cells (we forget the wumpus and the gold for now)
- ▷ Where should the agent go, if there is a breeze at $[1,2]$ and $[2,1]$?
- ▷ Pure logical inference can conclude nothing about which square is *most likely* to be safe!



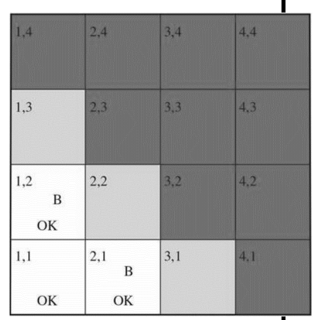
We can model this using the Boolean random variables:

- ▷ $P_{i,j}$ for $i,j \in \{1,2,3,4\}$, stating there is a pit at square $[i,j]$, and
- ▷ $B_{i,j}$ for $(i,j) \in \{(1,1),(1,2),(2,1)\}$, stating there is a breeze at square $[i,j]$

⇒ let's apply our machinery!

Wumpus: Probabilistic Model

First: Let's try to compute the full joint probability distribution $\mathbb{P}(P_{1,1},\ldots,P_{4,4},B_{1,1},B_{1,2},B_{2,1})$.



1. By the product rule, this is equal to $\mathbb{P}(B_{1,1},B_{1,2},B_{2,1}|P_{1,1},\ldots,P_{4,4})\cdot\mathbb{P}(P_{1,1},\ldots,P_{4,4})$.
2. Note that $\mathbb{P}(B_{1,1},B_{1,2},B_{2,1}|P_{1,1},\ldots,P_{4,4})$ is either 1 (if all the $B_{i,j}$ are consistent with the positions of the pits $P_{k,l}$) or 0 (otherwise).
3. Since the pits are spread independently, we have $\mathbb{P}(P_{1,1},\ldots,P_{4,4}) = \prod_{i,j=1,1}^{4,4}\mathbb{P}(P_{i,j})$

⇒ We know all of these probabilities.

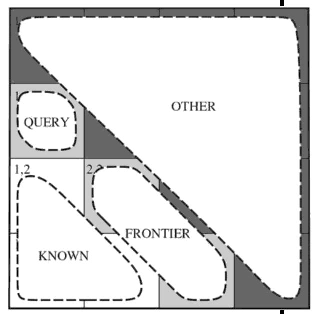
⇒ We can now use enumeration to compute
$\mathbb{P}(P_{i,j}| < known >) = \alpha(\sum_{<unknowns>}\mathbb{P}(P_{i,j}, < known >, < unknowns >))$

Wumpus Continued

Problem: We only know $P_{i,j}$ for three fields. If we want to compute e.g. $P_{1,3}$ via enumeration, that leaves $2^{4^2-4} = 4096$ terms to sum over!

Let's do better.

- Let $b{:=}\neg B_{1,1}\wedge B_{1,2}\wedge B_{2,1}$ (All the breezes we know about)
- Let $p{:=}\neg P_{1,1}\wedge\neg P_{1,2}\wedge\neg P_{2,1}$. (All the pits we know about)
- Let $F{:=}\{P_{3,1}\wedge P_{2,2},\neg P_{3,1}\wedge P_{2,2},P_{3,1}\wedge\neg P_{2,2},P_{3,1}\wedge\neg P_{2,2}\}$ (the current *"frontier"*)
- Let O be (the set of assignments for) all the other variables $P_{i,j}$. (i.e. except p, F and our query $P_{1,3}$)



Then the observed breezes b are conditionally independent of O given p and F. (Whether there is a pit anywhere else does not influence the breezes we observe.)

$\Rightarrow$ $P(b|P_{1,3},p,O,F)=P(b|P_{1,3},p,F)$. Let's exploit this!

Optimized Wumpus

$$\begin{aligned}\mathbb{P}(P_{1,3}|p,b)&=\alpha(\sum_{o\in O,f\in F}\mathbb{P}(P_{1,3},b,p,f,o))=\alpha(\sum_{o\in O,f\in F}P(b|p,o,f)\cdot\mathbb{P}(P_{1,3},p,f,o))\\&=\alpha(\sum_{f\in F}\sum_{o\in O}P(b|p,f)\cdot\mathbb{P}(P_{1,3},p,f,o))=\alpha(\sum_{f\in F}P(b|p,f)\cdot(\sum_{o\in O}\mathbb{P}(P_{1,3},p,f,o)))\\&=\alpha(\sum_{f\in F}P(b|p,f)\cdot(\sum_{o\in O}\mathbb{P}(P_{1,3})\cdot P(p)\cdot P(f)\cdot P(o)))\\&=\alpha(\mathbb{P}(P_{1,3})\cdot P(p)\cdot(\sum_{f\in F}\underbrace{P(b|p,f)}_{\in\{0,1\}}\cdot P(f)\cdot\underbrace{(\sum_{o\in O}P(o))}_{=1}))\end{aligned}$$

$\Rightarrow$ this is just a sum over the frontier, i.e. 4 terms ☺

So: $\mathbb{P}(P_{1,3}|p,b)=\alpha(\langle 0.2\cdot(0.8)^3\cdot(1\cdot 0.04+1\cdot 0.16+1\cdot 0.16+0),0.8\cdot(0.8)^3\cdot(1\cdot 0.04+1\cdot 0.16+0+0)\rangle)\approx\langle 0.31,0.69\rangle$

Analogously: $\mathbb{P}(P_{3,1}|p,b)=\langle 0.31,0.69\rangle$ and $\mathbb{P}(P_{2,2}|p,b)=\langle 0.86,0.14\rangle$ ($\Rightarrow$ avoid $[2,2]$!)

Cooking Recipe

In general, when you want to reason probabilistically, a good heuristic is:

1. Try to frame the full joint probability distribution in terms of the probabilities you know. Exploit product rule/chain rule, independence, conditional independence, marginalization **and domain knowledge** (as e.g. $\mathbb{P}(b|p,f)\in\{0,1\}$)

$\Rightarrow$ the problem can be solved at all!

2. **Simplify**: Start with the equation for enumeration:

$$\mathbb{P}(Q|E_1,\ldots)=\alpha(\sum_{u\in U}\mathbb{P}(Q,E_1,\ldots,U_1=u_1,\ldots))$$

3. Substitute by the result of 1., and again, exploit all of our machinery
4. Implement the resulting (system of) equation(s)
5. ???
6. Profit

Summary

- Probability distributions and conditional probability distributions allow us to represent random variables as convenient datastructures in an implementation (Assuming they are finite domain...)
- The full joint probability distribution allows us to compute all probabilities of statements about the random variables contained (But possibly inefficient)
- Marginalization and normalization are the specific techniques for extracting the *specific* probabilities we are interested in from the full joint probability distribution.
- The product and chain rule, exploiting (conditional) independence, Bayes' Theorem, and of course *domain specific* knowledge allow us to do so much more efficiently.
- Naive Bayes models are one example where all these techniques come together.

Chapter 4

Probabilistic Reasoning: Bayesian Networks

4.1 Introduction

John, Mary, and My Brand-New Alarm

Example 4.1.1 (From Russell/Norvig).

- ▷ I got very valuable stuff at home. So I bought an alarm. Unfortunately, the alarm just rings at home, doesn't call me on my mobile.
- ▷ I've got two neighbors, Mary and John, who'll call me if they hear the alarm.
- ▷ The problem is that, sometimes, the alarm is caused by an earthquake.
- ▷ Also, John might confuse the alarm with his telephone, and Mary might miss the alarm altogether because she typically listens to loud music.

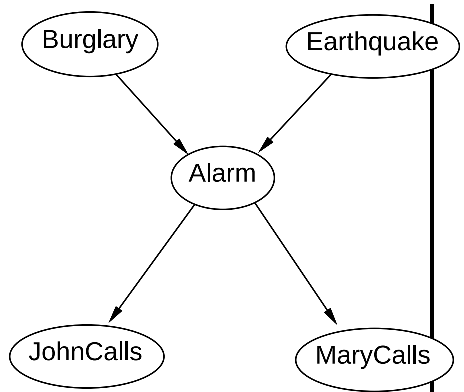
⇒ Random variables: `Burglary`, `Earthquake`, `Alarm`, `John`, `Mary`.

Given that both John and Mary call me, what is the probability of a burglary?

⇒ This is *almost* a naive Bayes model, but with multiple causes (`Burglary` and `Earthquake`) for the `Alarm`, which in turn may cause `John` and/or `Mary`.

John, Mary, and My Alarm: Assumptions

We assume:



- ▷ We (should) know $\mathbb{P}(\mathtt{Alarm}|\mathtt{Burglary},\mathtt{Earthquake})$, $\mathbb{P}(\mathtt{John}|\mathtt{Alarm})$, and $\mathbb{P}(\mathtt{Mary}|\mathtt{Alarm})$.
- ▷ $\mathtt{Burglary}$ and $\mathtt{Earthquake}$ are independent.
- ▷ $\mathtt{John}$ and $\mathtt{Mary}$ are conditionally independent given $\mathtt{Alarm}$.
- ▷ Moreover: Both $\mathtt{John}$ and $\mathtt{Mary}$ are conditionally independent of *any other random variables* in the graph given $\mathtt{Alarm}$. (Only $\mathtt{Alarm}$ causes them, and everything else only causes them indirectly *through* $\mathtt{Alarm}$)

First Step: Construct the full joint probability distribution,
Second Step: Use **enumeration** to compute $\mathbb{P}(\mathtt{Burglary}|\mathtt{John}=\mathsf{T},\mathtt{Mary}=\mathsf{T})$.

John, Mary, and My Alarm: The Distribution

$$\begin{aligned}
&\mathbb{P}(\mathtt{John},\mathtt{Mary},\mathtt{Alarm},\mathtt{Burglary},\mathtt{Earthquake})\\
=&\mathbb{P}(\mathtt{John}|\mathtt{Mary},\mathtt{Alarm},\mathtt{Burglary},\mathtt{Earthquake})\cdot\mathbb{P}(\mathtt{Mary}|\mathtt{Alarm},\mathtt{Burglary},\mathtt{Earthquake})\\
&\cdot\mathbb{P}(\mathtt{Alarm}|\mathtt{Burglary},\mathtt{Earthquake})\cdot\mathbb{P}(\mathtt{Burglary}|\mathtt{Earthquake})\cdot\mathbb{P}(\mathtt{Earthquake})\\
=&\mathbb{P}(\mathtt{John}|\mathtt{Alarm})\cdot\mathbb{P}(\mathtt{Mary}|\mathtt{Alarm})\cdot\mathbb{P}(\mathtt{Alarm}|\mathtt{Burglary},\mathtt{Earthquake})\cdot\mathbb{P}(\mathtt{Burglary})\cdot\mathbb{P}(\mathtt{Earthquake})
\end{aligned}$$

We plug into the equation for enumeration:

$$\mathbb{P}(\mathtt{Burglary}|\mathtt{John}=\mathsf{T},\mathtt{Mary}=\mathsf{T})=\alpha(\mathbb{P}(\mathtt{Burglary})\sum_{a\in\{\mathsf{T},\mathsf{F}\}}P(\mathtt{John}|\mathtt{Alarm}=a)\cdot P(\mathtt{Mary}|\mathtt{Alarm}=a)$$

$$\cdot\sum_{q\in\{\mathsf{T},\mathsf{F}\}}\mathbb{P}(\mathtt{Alarm}=a|\mathtt{Burglary},\mathtt{Earthquake}=q)P(\mathtt{Earthquake}=q))$$

⇒ Now let's scale things up to arbitrarily many variables!

Bayesian Networks: Definition

Definition 4.1.2. A Bayesian network consists of

1. a directed acyclic graph $\langle\mathcal{X},E\rangle$ of random variables $\mathcal{X}=\{X_1,\ldots,X_n\}$, and
2. a conditional probability distribution $\mathbb{P}(X_i|\mathrm{Parents}(X_i))$ for every $X_i\in\mathcal{X}$ (also called the CPT for conditional probability table)

such that every X_i is conditionally independent of any conjunctions of non-descendents of X_i given $\mathrm{Parents}(X_i)$.

Definition 4.1.3. Let $\langle\mathcal{X},E\rangle$ be a directed acyclic graph, $X\in\mathcal{X}$, and E^* the reflexive transitive closure of E. The non-descendents of X are the elements of the set $\mathrm{NonDesc}(X){:=}\{Y|(X,Y)\notin E^*\}\backslash\mathrm{Parents}(X)$.

Note that the roots of the graph are conditionally independent given the empty set; i.e. they are independent.

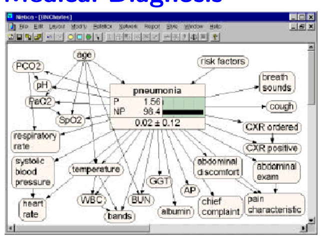
Theorem 4.1.4. *The full joint probability distribution of a Bayesian network $\langle \mathcal{X}, E\rangle$ is given by*

$$\mathbb{P}(X_1,\ldots,X_n) = \prod_{X_i\in\mathcal{X}} \mathbb{P}(X_i|\mathrm{Parents}(X_i))$$

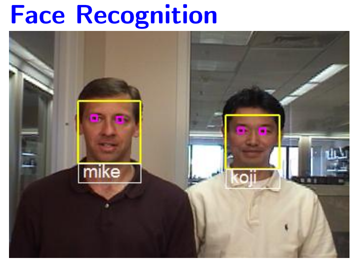
Some Applications

▷ A ubiquitous problem: Observe "symptoms", need to infer "causes".

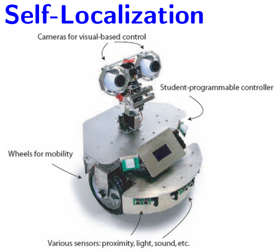
Medical Diagnosis



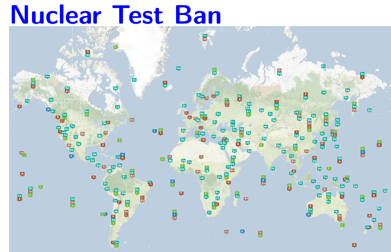
Face Recognition



Self-Localization



Nuclear Test Ban



4.2 Constructing Bayesian Networks

Compactness of Bayesian Networks

▷ **Definition 4.2.1.** Given random variables $X_1,\ldots,X_n$ with finite domains $D_1,\ldots,D_n$, the size of $\mathcal{B}{:=}\langle\{X_1,\ldots,X_n\}, E\rangle$ is defined as

$$\mathrm{size}(\mathcal{B}){:=}\sum_{i=1}^{n} |D_i| \cdot \prod_{X_j\in\mathrm{Parents}(X_i)} |D_j|$$

▷ **Note:** $\mathrm{size}(\mathcal{B}) \mathrel{\widehat{=}}$ The total number of entries in the conditional probability distributions.

▷ **Note:** Smaller BN $\rightsquigarrow$ need to assess less probabilities, more efficient inference.

▷ **Observation 4.2.2.** *Explicit full joint probability distribution has size* $\prod_{i=1}^{n} |D_i|$.

▷ **Observation 4.2.3.** *If* $|\mathrm{Parents}(X_i)| \leq k$ *for every* X_i*, and* $D_{\max}$ *is the largest random variable domain, then* $\mathrm{size}(\mathcal{B}) \leq n|D_{\max}|^{k+1}$.

▷ **Example 4.2.4.** For $|D_{\max}| = 2$, $n = 20$, $k = 4$ we have $2^{20} = 1048576$ probabilities, but a Bayesian network of size $\leq 20 \cdot 2^5 = 640 \ldots$!

▷ In the *worst case*, $\mathrm{size}(\mathcal{B}) = n \cdot \prod_{i=1}^{n} |D_i|$, namely if every variable depends on all its predecessors in the chosen variable ordering.

▷ **Intuition:** BNs are compact – i.e. of small size – if each variable is directly influenced only by few of its predecessor variables.

Keeping Networks Small

To keep our Bayesian networks small, we can:

1. **Reduce the number of edges:** ⇒ Order the variables to allow for exploiting conditional independence (causes before effects), or
2. **represent the conditional probability distributions efficiently:**
   (a) For Boolean random variables X, we only need to store $\mathbb{P}(X = \mathsf{T}|\mathrm{Parents}(X))$ ($\mathbb{P}(X = \mathsf{F}|\mathrm{Parents}(X)) = 1 - \mathbb{P}(X = \mathsf{T}|\mathrm{Parents}(X))$) (Cuts the number of entries in half!)
   (b) Introduce different **kinds** of nodes exploiting domain knowledge; e.g. deterministic and noisy disjunction nodes.

Reducing Edges: Variable Order Matters

Given a set of random variables $X_1, \ldots, X_n$, consider the following (impractical, but illustrative) pseudo-algorithm for constructing a Bayesian network:

▷ **Definition 4.2.5 (BN construction algorithm).**

1. Initialize $BN := \langle\{X_1, \ldots, X_n\}, E\rangle$ where $E = \emptyset$.
2. Fix any variable ordering, $X_1, \ldots, X_n$.
3. **for** $i := 1, \ldots, n$ **do**
   a. Choose a minimal set $\mathrm{Parents}(X_i) \subseteq \{X_1, \ldots, X_{i-1}\}$ such that
   $$\mathbb{P}(X_i|X_{i-1}, \ldots, X_1) = \mathbb{P}(X_i|\mathrm{Parents}(X_i))$$
   b. For each $X_j \in \mathrm{Parents}(X_i)$, insert (X_j, X_i) into E.
   c. Associate X_i with $\mathbb{P}(X_i|\mathrm{Parents}(X_i))$.

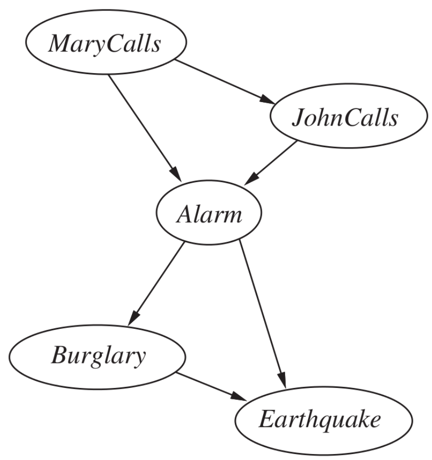
▷ **Attention:** *Which* variables we need to include into $\mathrm{Parents}(X_i)$ depends on what "$\{X_1, \ldots, X_{i-1}\}$" is . . . !

▷ **Thus:** The size of the resulting BN depends on the chosen variable ordering $X_1, \ldots, X_n$.

▷ **In Particular:** The size of a Bayesian network is *not* a fixed property of the domain. It depends on the skill of the designer.

John and Mary Depend on the Variable Order!

▷ **Example 4.2.6.** `Mary`, `John`, `Alarm`, `Burglary`, `Earthquake`.

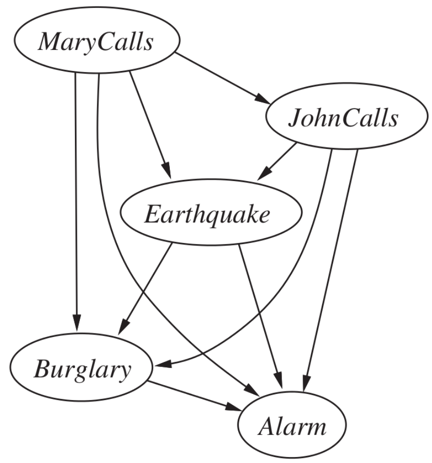


Note: For **??** we try to determine whether – given different value assignments to potential parents – the probability of X_i being true differs? If yes, we include these parents. In the particular case:

1. M to J yes because the common cause may be the alarm.
2. M, J to A yes because they may have heard alarm.
3. A to B yes because if A then higher chance of B.
4. However, M/J to B no because M/J only react to the alarm so if we have the value of A then values of M/J don't provide more information about B.
5. A to E yes because if A then higher chance of E.
6. B to E yes because, if A and not B then chances of E are higher than if A and B.

John and Mary Depend on the Variable Order! Ctd.

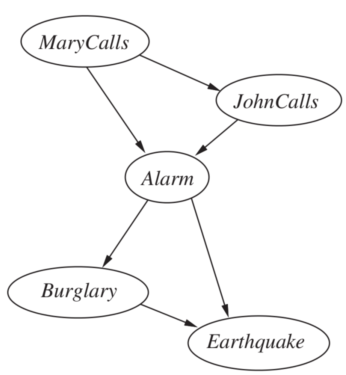
▷ **Example 4.2.7.** `Mary`, `John`, `Earthquake`, `Burglary`, `Alarm`.

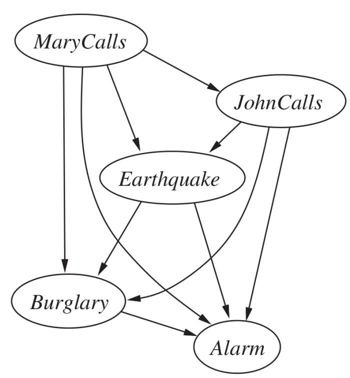


Again: Given different value assignments to potential parents, does the probability of X_i being true differ? If yes, include these parents.

1. M to J as before.
2. M, J to E as probability of E is higher if M/J is true.
3. Same for B; E to B because, given M and J are true, if E is true as well then prob of B is lower than if E is false.
4. $M/J/B/E$ to A because if $M/J/B/E$ is true (even when changing the value of just one of these) then probability of A is higher.

John and Mary, What Went Wrong?





- **Intuition:** These BNs link from *effects* to their *causes*!
  - ⇒ Even though `Mary` and `John` are conditionally independent given `Alarm`, this is not exploited, since `Alarm` is not ordered before `Mary` and `John`!

⇒ **Rule of Thumb:** We should order causes before symptoms.

Representing Conditional Distributions: Deterministic Nodes

Definition 4.2.8. A node X in a Bayesian network is called deterministic, if its value is completely determined by the values of $\mathrm{Parents}(X)$.

Example 4.2.9. The *sum of two dice throws* S is entirely determined by the values of the two dice $First$ and $Second$.

Example 4.2.10. In the *Wumpus* example, the *breezes* are entirely determined by the *pits*

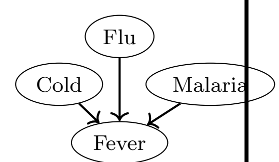
⇒ *Deterministic* nodes model direct, *causal* relationships.
⇒ If X is deterministic, then $P(X|\mathrm{Parents}(X)) \in \{0, 1\}$

⇒ we can replace the conditional probability distribution $\mathbb{P}(X|\mathrm{Parents}(X))$ by a boolean function.

Representing Conditional Distributions: Noisy Nodes

Sometimes, values of nodes are "almost deterministic":

Example 4.2.11 (Inhibited Causal Dependencies).
Assume the network on the right contains *all* possible causes of fever. (Or add a dummy-node for "other causes")
If there is a fever, then *one* of them (at least) must be the cause, but none of them *necessarily* cause a fever: The causal relation between parent and child is inhibited.



⇒ We can model the inhibitions by individual inhibition factors q_d.

Definition 4.2.12. The conditional probability distribution of a noisy disjunction node X with $\mathrm{Parents}(X) = X_1, \ldots, X_n$ in a Bayesian network is given by $P(X|X_1, \ldots, X_n) = 1 - \prod_{\{j|X_j=\mathsf{T}\}} q_j$, where the q_i are the inhibition factors of $X_i \in \mathrm{Parents}(X)$, defined as $q_i := P(\neg X|\neg X_1, \ldots, \neg X_{i-1}, X_i, \neg X_{i+}$

⇒ Instead of a distribution with 2^k parameters, we only need k parameters!

Representing Conditional Distributions: Noisy Nodes

▷ **Example 4.2.13.** Assume the following inhibition factors for Example 4.2.11:

$$\begin{aligned} q_{\mathrm{cold}} &= P(\neg\mathrm{fever}|\mathrm{cold}, \neg\mathrm{flu}, \neg\mathrm{malaria}) = 0.6 \\ q_{\mathrm{flu}} &= P(\neg\mathrm{fever}|\neg\mathrm{cold}, \mathrm{flu}, \neg\mathrm{malaria}) = 0.2 \\ q_{\mathrm{malaria}} &= P(\neg\mathrm{fever}|\neg\mathrm{cold}, \neg\mathrm{flu}, \mathrm{malaria}) = 0.1 \end{aligned}$$

If we model Fever as a noisy disjunction node, then the general rule $P(X_i|\mathrm{Parents}(X_i)) =$

$\prod_{\{j|X_j=\mathsf{T}\}} q_j$ for the CPT gives the following table:

Cold	Flu	Malaria	$P(\text{Fever})$	$P(\neg\text{Fever})$
F	F	F	0.0	1.0
F	F	T	0.9	**0.1**
F	T	F	0.8	**0.2**
F	T	T	0.98	$0.02 = 0.2 \cdot 0.1$
T	F	F	0.4	**0.6**
T	F	T	0.94	$0.06 = 0.6 \cdot 0.1$
T	T	F	0.88	$0.12 = 0.6 \cdot 0.2$
T	T	T	0.988	$0.012 = 0.6 \cdot 0.2 \cdot 0.1$

Representing Conditional Distributions: Summary

- Note that deterministic nodes and noisy disjunction nodes are just two examples of "specialized" kinds of nodes in a Bayesian network.
- In general, noisy logical relationships in which a variable depends on k parents can be described by $\mathcal{O}(k)$ parameters instead of $\mathcal{O}(2^k)$ for the full conditional probability table. This can make assessment (and learning) tractable.
- **Example 4.2.14.** The CPCS network [Pra+94] uses noisy-OR and noisy-MAX distributions to model relationships among diseases and symptoms in internal medicine. With 448 nodes and 906 links, it requires only 8,254 values instead of 133,931,430 for a network with full conditional probability distributions.

4.3 Inference in Bayesian Networks

Probabilistic Inference Tasks in Bayesian Networks

Remember:

Definition 4.3.1 (Probabilistic Inference Task). Let $X_1,\ldots,X_n = Q_1,\ldots,Q_{n_Q},E_1,\ldots,E_{n_E},U_1,\ldots,U_{n_U}$ be a set of random variables, a probabilistic inference task.

We wish to compute the conditional probability distribution $\mathbb{P}(Q_1,\ldots,Q_{n_Q}|E_1 = e_1,\ldots,E_{n_E} = e_{n_E})$.

We call

- a $Q_1,\ldots,Q_{n_Q}$ the query variables,
- a $E_1,\ldots,E_{n_E}$ the evidence variables, and
- $U_1,\ldots,U_{n_U}$ the hidden variables.

We know the full joint probability distribution: $\mathbb{P}(X_1,\ldots,X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i|\text{Parents}(X_i))$

And we know about enumeration:

$$\mathbb{P}(Q_1, \ldots, Q_{n_Q} | E_1 = e_1, \ldots, E_{n_E} = e_{n_E}) =$$
$$\alpha(\sum_{u \in D_U} \mathbb{P}(Q_1, \ldots, Q_{n_Q}, E_1 = e_1, \ldots, E_{n_E} = e_{n_E}, U_1 = u_1, \ldots, U_{n_U} = u_{n_U}))$$

(where $D_U = \mathbf{dom}(U_1) \times \ldots \times \mathbf{dom}(U_{n_U})$)

Enumeration: The Alarm-Example

Remember our example: $\mathbb{P}(\texttt{Burglary} | \texttt{John}, \texttt{Mary})$
(hidden variables: $\texttt{Alarm}, \texttt{Earthquake}$)

$= \alpha(\sum_{b_a, b_e \in \{\mathsf{T}, \mathsf{F}\}} P(\texttt{John}, \texttt{Mary}, \texttt{Alarm} = b_a, \texttt{Earthquake} = b_e, \texttt{Burglary}))$
$= \alpha(\sum_{b_a, b_e \in \{\mathsf{T}, \mathsf{F}\}} P(\texttt{John} | \texttt{Alarm} = b_a) \cdot P(\texttt{Mary} | \texttt{Alarm} = b_a)$
$\cdot \mathbb{P}(\texttt{Alarm} = b_a | \texttt{Earthquake} = b_e, \texttt{Burglary}) \cdot P(\texttt{Earthquake} = b_e) \cdot \mathbb{P}(\texttt{Burglary}))$

⇒ These are 5 factors in 4 summands ($b_a, b_e \in \{\mathsf{T}, \mathsf{F}\}$) over two cases ($\texttt{Burglary} \in \{\mathsf{T}, \mathsf{F}\}$),
⇒ 38 arithmetic operations (+3 for α)

General worst case: $\mathcal{O}(n2^n)$

Let's do better!

Enumeration: First Improvement

Some abbreviations: $j := \texttt{John}, m := \texttt{Mary}, a := \texttt{Alarm}, e := \texttt{Earthquake}, b := \texttt{Burglary}$,

$$\mathbb{P}(b | j, m) = \alpha(\sum_{b_a, b_e \in \{\mathsf{T}, \mathsf{F}\}} P(j | a = b_a) \cdot P(m | a = b_a) \cdot \mathbb{P}(a = b_a | e = b_e, b) \cdot P(e = b_e) \cdot \mathbb{P}(b))$$

Let's "optimize":

$$\mathbb{P}(b | j, m) = \alpha(\mathbb{P}(b) \cdot (\sum_{b_e \in \{\mathsf{T}, \mathsf{F}\}} P(e = b_e) \cdot (\sum_{b_a \in \{\mathsf{T}, \mathsf{F}\}} \mathbb{P}(a = b_a | e = b_e, b) \cdot P(j | a = b_a) \cdot P(m | a = b_a))))$$

⇒ 3 factors in 2 summand + 2 factors in 2 summands + two factors in the outer product, over two cases = 28 arithmetic operations (+3 for α)

Second Improvement: Variable Elimination 1

Consider $\mathbb{P}(j | b = \mathsf{T})$. Using enumeration:

$$= \alpha(P(b) \cdot (\sum_{b_e \in \{\mathsf{T}, \mathsf{F}\}} P(e = b_e) \cdot (\sum_{a_e \in \{\mathsf{T}, \mathsf{F}\}} P(a = a_e | e = b_e, b) \cdot \mathbb{P}(j | a = a_e) \cdot \underbrace{(\sum_{a_m \in \{\mathsf{T}, \mathsf{F}\}} P(m = a_m | a = a_e))))}_{=1})$$

$\Rightarrow \mathbb{P}(\mathtt{John}|\mathtt{Burglary} = \mathsf{T})$ does not depend on $\mathtt{Mary}$ (duh...)

More generally:

Lemma 4.3.2. *Given a query* $\mathbb{P}(Q_1, \ldots, Q_{n_Q}|E_1 = e_1, \ldots, E_{n_E} = e_{n_E})$*, we can ignore (and remove) all hidden leafs of the Bayesian network.*

...doing so yields new leafs, which we can then ignore again, etc., until:

Lemma 4.3.3. *Given a query* $\mathbb{P}(Q_1, \ldots, Q_{n_Q}|E_1 = e_1, \ldots, E_{n_E} = e_{n_E})$*, we can ignore (and remove) all hidden variables that are not ancestors of any of the* $Q_1, \ldots, Q_{n_Q}$ *or* $E_1, \ldots, E_{n_E}$.

Enumeration: First Algorithm

Assume the $X_1, \ldots, X_n$ are topologically sorted (causes before effects)

function ENUMERATE-QUERY(Q,$\langle E_1 = e_1, \ldots, E_{n_E} = e_{n_E}\rangle$)
  $P := \langle\rangle$ /* $= \mathbb{P}(Q|E_i = e_i)$ */
  $X_1, \ldots, X_n :=$ variables filtered according to ??, topologically sorted
  for all $q \in \mathbf{dom}(Q)$ **do**
    $P_i :=$ ENUMALL($\langle X_1, \ldots, X_n\rangle$,$\langle E_1 = e_1, \ldots, E_{n_E} = e_{n_E}, Q = q\rangle$)
  return $\alpha(P)$

function ENUMALL($\langle Y_1, \ldots, Y_{n_Y}\rangle$,$\langle A_1 = a_1, \ldots, A_{n_A} = a_{n_A}\rangle$)
  /* By construction, $\mathrm{Parents}(Y_1) \subset \{A_1, \ldots, A_{n_A}\}$ */
  if $n_y = 0$ **then return** 1.0
  else if $Y_1 = A_j$ **then return** $P(A_j = a_j|\mathrm{Parents}(A_j))\cdot$ENUMALL($\langle Y_2, \ldots, Y_{n_Y}\rangle$,$\langle A_1 = a_1, \ldots, A_{n_A} = a_{n_A}\rangle$)
  else return $\sum_{y \in \mathbf{dom}(Y_1)} P(Y_1 = y|\mathrm{Parents}(Y_1))\cdot$ENUMALL($\langle Y_2, \ldots, Y_{n_Y}\rangle$,$\langle A_1 = a_1, \ldots, A_{n_A} = a_{n_A}, Y_1 = y\rangle$)

General worst case: $\mathcal{O}(2^n)$ – better, but still not great

Enumeration: Example

Variable order: b, e, a, j, m

$$\triangleright\ P_0 := P(b) \cdot \left[+ \begin{array}{l} P(e) \cdot \left[+ \begin{array}{l} P(a|b,e) \cdot P(j|a) \cdot P(m|a) \cdot 1.0 \\ P(\neg a|b,e) \cdot P(j|\neg a) \cdot P(m|\neg a) \cdot 1.0 \end{array}\right. \\ P(\neg e) \cdot \left[+ \begin{array}{l} P(a|b,\neg e) \cdot P(j|a) \cdot P(m|a) \cdot 1.0 \\ P(\neg a|b,\neg e) \cdot P(j|\neg a) \cdot P(m|\neg a) \cdot 1.0 \end{array}\right. \end{array}\right.$$

$$\triangleright\ P_1 := P(\neg b) \cdot \left[+ \begin{array}{l} P(e) \cdot \left[+ \begin{array}{l} P(a|\neg b,e) \cdot P(j|a) \cdot P(m|a) \cdot 1.0 \\ P(\neg a|\neg b,e) \cdot P(j|\neg a) \cdot P(m|\neg a) \cdot 1.0 \end{array}\right. \\ P(\neg e) \cdot \left[+ \begin{array}{l} P(a|\neg b,\neg e) \cdot P(j|a) \cdot P(m|a) \cdot 1.0 \\ P(\neg a|\neg b,\neg e) \cdot P(j|\neg a) \cdot P(m|\neg a) \cdot 1.0 \end{array}\right. \end{array}\right.$$

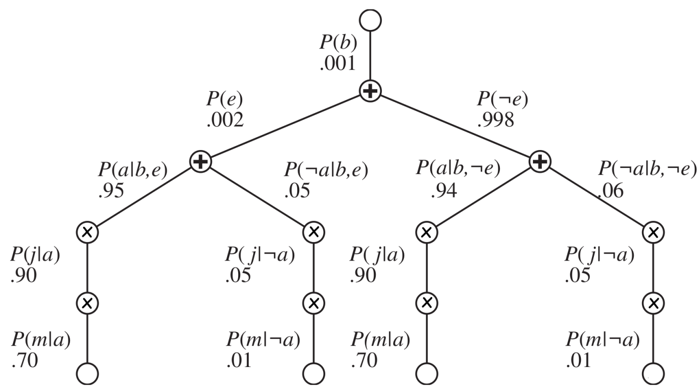
$$\Leftarrow \langle \frac{P_0}{P_0+P_1}, \frac{P_1}{P_0+P_1} \rangle$$

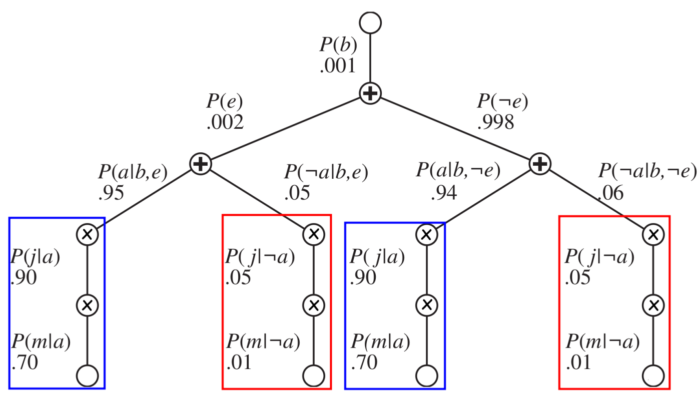
$$\mathbb{P}(b|j = \mathsf{T}, m = \mathsf{T}) = \alpha(\mathbb{P}(b) \cdot (\sum_{b_e \in \{\mathsf{T},\mathsf{F}\}} P(e = b_e) \cdot (\sum_{b_a \in \{\mathsf{T},\mathsf{F}\}} \mathbb{P}(a = b_a|e = b_e, b) \cdot P(j|a = b_a) \cdot P(m|a = b_a))))$$

The Evaluation of $P(b|j,m)$ as a "Search Tree"

$$\mathbb{P}(b|j,m) = \alpha(\mathbb{P}(b)\cdot(\sum_{b_e\in\{T,F\}} P(e=b_e)\cdot(\sum_{b_a\in\{T,F\}} \mathbb{P}(a=b_a|e=b_e,b)\cdot P(j|a=b_a)\cdot P(m|a=b_a))))$$

Note: ENUMERATE-QUERY corresponds to depth-first traversal of an arithmetic expression-tree:





Variable Elimination 2

$$\mathbb{P}(b|j,m) = \alpha(\mathbb{P}(b)\cdot(\sum_{b_e\in\{T,F\}} P(e=b_e)\cdot(\sum_{b_a\in\{T,F\}} \mathbb{P}(a=b_a|e=b_e,b)\cdot P(j|a=b_a)\cdot P(m|a=b_a))))$$

The last two factors $P(j|a=b_a), P(m|a=b_a)$ only depend on a, but are "trapped" behind the summation over e, hence computed twice in two distinct recursive calls to ENUMALL

Idea: Instead of left-to-right (top-down DFS), operate right-to-left (bottom-up) and store intermediate "factors" along with their "dependencies":

$$\underbrace{\alpha(\mathbb{P}(b)}_{\mathbf{f}_7(b)}\cdot\underbrace{(\sum_{b_e\in\{T,F\}}\underbrace{P(e=b_e)}_{\mathbf{f}_5(e)}\cdot\underbrace{(\sum_{b_a\in\{T,F\}}\underbrace{\mathbb{P}(a=b_a|e=b_e,b)}_{\mathbf{f}_3(a,b,e)}\cdot\underbrace{P(j|a=b_a)}_{\mathbf{f}_2(a)}\cdot\underbrace{P(m|a=b_a)}_{\mathbf{f}_1(a)}))}_{\mathbf{f}_4(b,e)})}_{\mathbf{f}_6(b)}$$

Variable Elimination: Example

We only show variable elimination by example: (implementation details get tricky, but the idea is simple)

$\mathbb{P}(b) \cdot (\sum_{b_e \in \{\mathsf{T},\mathsf{F}\}} P(e = b_e) \cdot (\sum_{b_a \in \{\mathsf{T},\mathsf{F}\}} \mathbb{P}(a = b_a | e = b_e, b) \cdot P(j | a = b_a) \cdot P(m | a = b_a)))$

Assume reverse topological order of variables: m, j, a, e, b

- ▷ m is an evidence variable with value T and dependency a, which is a hidden variable. We introduce a new "factor" $\mathbf{f}(a) := \mathbf{f}_1(a) := \langle P(m|a), P(m|\neg a) \rangle$.
- ▷ j works analogously, $\mathbf{f}_2(a) := \langle P(j|a), P(j|\neg a) \rangle$. We "multiply" with the existing factor, yielding $\mathbf{f}(a) := \langle \mathbf{f}_1(a) \cdot \mathbf{f}_2(a), \mathbf{f}_1(\neg a) \cdot \mathbf{f}_2(\neg a) \rangle = \langle P(m|a) \cdot P(j|a), P(m|\neg a) \cdot P(j|\neg a) \rangle$
- ▷ a is a hidden variable with dependencies e (hidden) and b (query).
  1. We introduce a new "factor" $\mathbf{f}_3(a, e, b)$, a $2 \times 2 \times 2$ table with the relevant conditional probabilities $\mathbb{P}(a|e, b)$.
  2. We multiply each entry of $\mathbf{f}_3$ with the relevant entries of the existing factor $\mathbf{f}$, yielding $\mathbf{f}(a, e, b)$.
  3. We "sum out" the resulting factor over a, yielding a new factor $\mathbf{f}(e, b) = \mathbf{f}(a, e, b) + \mathbf{f}(\neg a, e, b)$.
- ▷ ...

⇒ can speed things up by a factor of 1000! (or more, depending on the order of variables!)

The Complexity of Exact Inference

- ▷ **Definition 4.3.4.** A graph G is called singly connected, or a polytree (otherwise multiply connected), if there is at most one undirected path between any two nodes in G.
- ▷ **Theorem 4.3.5 (Good News).** *On singly connected Bayesian networks, variable elimination runs in polynomial time.*
- ▷ Is our BN for Mary & John a polytree? (Yes.)
- ▷ **Theorem 4.3.6 (Bad News).** *For multiply connected Bayesian networks, probabilistic inference is $\#\mathbf{P}$-hard.* ($\#\mathbf{P}$ *is harder than* $\mathbf{NP}$*, i.e.* $\mathbf{NP} \subseteq \#\mathbf{P}$)
- ▷ **So?:** Life goes on ... In the hard cases, if need be we can throw exactitude to the winds and approximate.
- ▷ **Example 4.3.7.** Sampling techniques as in MCTS.

4.4 Conclusion

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/29228`.

Summary

- ▷ Bayesian networks (BN) are a wide-spread tool to model uncertainty, and to reason about it. A BN represents conditional independence relations between random variables. It consists of a graph encoding the variable dependencies, and of conditional probability tables (CPTs).
- ▷ Given a variable ordering, the BN is small if every variable depends on only a few of its predecessors.
- ▷ Probabilistic inference requires to compute the probability distribution of a set of query variables, given a set of evidence variables whose values we know. The remaining variables are hidden.
- ▷ Inference by enumeration takes a BN as input, then applies Normalization+Marginalization, the chain rule, and exploits conditional independence. This can be viewed as a tree search that branches over all values of the hidden variables.
- ▷ Variable elimination avoids unnecessary computation. It runs in polynomial time for poly-tree BNs. In general, exact probabilistic inference is $\#\mathbf{P}$-hard. Approximate probabilistic inference methods exist.

Topics We Didn't Cover Here

- ▷ **Inference by sampling**: A whole zoo of methods for doing this exists.
- ▷ **Clustering**: Pre-combining subsets of variables to reduce the running time of inference.
- ▷ **Compilation to SAT**: More precisely, to "weighted model counting" in CNF formulas. Model counting extends DPLL with the ability to determine the number of satisfying interpretations. Weighted model counting allows to define a mass for each such interpretation (= the probability of an atomic event).
- ▷ **Dynamic BN**: BN with one slice of variables at each "time step", encoding probabilistic behavior over time.
- ▷ **Relational BN**: BN with predicates and object variables.
- ▷ **First-order BN**: Relational BN with quantification, i.e. probabilistic logic. E.g., the BLOG language developed by Stuart Russel and co-workers.

Reading:

- *Chapter 14: Probabilistic Reasoning* of [RN03].
  - Section 14.1 roughly corresponds to my "What is a Bayesian Network?".
  - Section 14.2 roughly corresponds to my "What is the Meaning of a Bayesian Network?" and "Constructing Bayesian Networks".The main change I made here is to *define* the semantics of the BN in terms of the conditional independence relations, which I find clearer than RN's definition that uses the reconstructed full joint probability distribution instead.
  - Section 14.4 roughly corresponds to my "Inference in Bayesian Networks". RN give full details on variable elimination, which makes for nice ongoing reading.

- Section 14.3 discusses how CPTs are specified in practice.
- Section 14.5 covers approximate sampling-based inference.
- Section 14.6 briefly discusses relational and first-order BNs.
- Section 14.7 briefly discusses other approaches to reasoning about uncertainty.

All of this is nice as additional background reading.

Chapter 5

Temporal Probability Models

5.1 Modeling Time and Uncertainty

Stochastic Processes

The world changes in *stochastically predictable ways*.

Example 5.1.1.

- ▷ The weather changes, but the weather tomorrow is somewhat predictable *given* today's weather and other factors, (which in turn (somewhat) depends on yesterday's weather, which in turn...)
- ▷ the stock market changes, but the stock price tomorrow is probably related to today's price,
- ▷ A patient's blood sugar changes, but their blood sugar is related to their blood sugar 10 minutes ago (in particular if they didn't eat anything in between)

How do we model this?

Definition 5.1.2. Let $\langle\Omega, P\rangle$ a probability space and $\langle S, \preceq\rangle$ a (not necessarily *totally*) ordered set.

A sequence of random variables $(X_t)_{t\in S}$ with $\mathbf{dom}(X_t) = D$ is called a stochastic process over the time structure S.

Intuition: X_t models the outcome of the random variable X at time step t. The sample space Ω corresponds to the set of all possible sequences of outcomes.

Note: We will almost exclusively use $\langle S, \preceq\rangle = \langle\mathbb{N}, \leq\rangle$.

Definition 5.1.3. Given a stochastic process X_t over S and $a, b \in S$ with $a \preceq b$, we write $\mathbf{X}_{a:b}$ for the sequence $X_a, X_{a+1}, \ldots, X_{b-1}, X_b$ and $E^{=e}_{a:b}$ for $E_a = e_a, \ldots, E_b = e_b$.

Stochastic Processes (Running Example)

Example 5.1.4 (Umbrellas). You are a security guard in a secret underground facility, want to know it if is raining outside. Your only source of information is whether the director comes in with an umbrella.

- ▷ We have a stochastic process $\texttt{Rain}_0, \texttt{Rain}_1, \texttt{Rain}_2, \ldots$ of hidden variables, and
- ▷ a related stochastic process $\texttt{Umbrella}_0, \texttt{Umbrella}_1, \texttt{Umbrella}_2, \ldots$ of evidence variables.

...and a combined stochastic process $\langle \texttt{Rain}_0, \texttt{Umbrella}_0 \rangle, \langle \texttt{Rain}_1, \texttt{Umbrella}_1 \rangle, \ldots$

Note that $\texttt{Umbrella}_t$ only depends on $\texttt{Rain}_t$, not on e.g. $\texttt{Umbrella}_{t-1}$ (except indirectly through $\texttt{Rain}_t$ / $\texttt{Rain}_{t-1}$).

Definition 5.1.5. We call a stochastic process of *hidden* variables a state variable.

Markov Processes

Idea: Construct a Bayesian network from these variables (parents?)
...without everything exploding in size...?

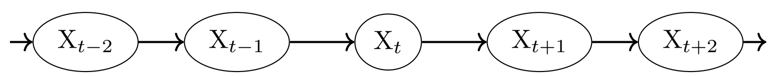
Definition 5.1.6. Let $(X_t)_{t \in S}$ a stochastic process. X has the (nth order) Markov property iff X_t only depends on a bounded subset of $\mathbf{X}_{0:t-1}$ – i.e. for all $t \in S$ we have $\mathbb{P}(X_t|X_0, \ldots X_{t-1}) = \mathbb{P}(X_t|X_{t-n}, \ldots X_{t-1})$ for some $n \in S$.

A stochastic process with the Markov property for some n is called a (nth order) Markov process.

Important special cases:

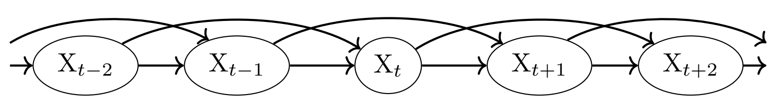
Definition 5.1.7.

- ▷ First-order Markov property: $\mathbb{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbb{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$



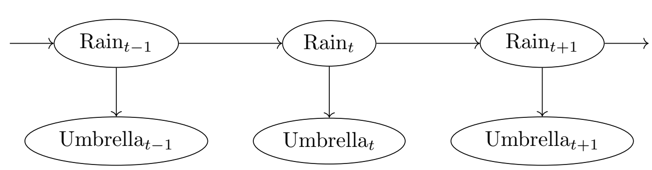
A first order Markov process is called a Markov chain.

- ▷ Second-order Markov property: $\mathbb{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbb{P}(\mathbf{X}_t|\mathbf{X}_{t-2}, \mathbf{X}_{t-1})$



Markov Process Example: The Umbrella

Example 5.1.8 (Umbrellas continued). We model the situation in a Bayesian network:



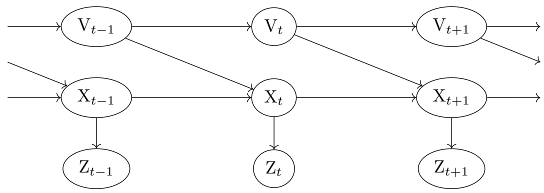
Problem: This network does not actually have the First-order Markov property...

Possible fixes: We have two ways to fix this:

1. Increase the order of the Markov process. (more dependencies ⇒ more complex inference)
2. Add more state variables, e.g., Temp_t, Pressure_t. (more information sources)

Markov Process Example: Robot Motion

Example 5.1.9 (Random Robot Motion). Assume we want to track a robot wandering randomly on the X/Y plane, whose position we can only observe roughly (e.g. by approximate GPS coordinates:) Markov chain



- ▷ the velocity V_i may change unpredictably.
- ▷ the exact position X_i depends on previous position X_{i-1} and velocity V_{i-1}
- ▷ the position X_i influences the observed position Z_i.

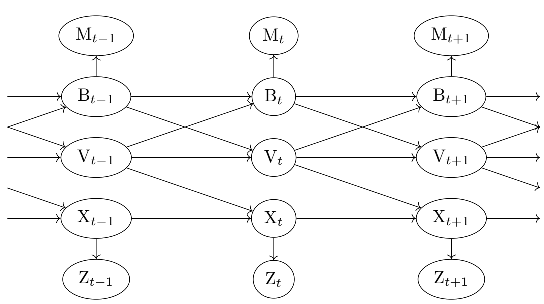
Example 5.1.10 (Battery Powered Robot). If the robot has a *battery*, the Markov property is violated!

- ▷ Battery exhaustion has a systematic effect on the change in velocity.
- ▷ This depends on how much power was used by all previous manoeuvres.

Markov Process Example: Robot Motion

Idea: We can restore the Markov property by including a state variable for the charge level B_t. (Better still: Battery level sensor)

Example 5.1.11 (Battery Powered Robot Motion).



- ▷ Battery level B_i is influenced by previous level $B_{i-1}$and velocity V_{i-1}.
- ▷ Velocity V_i is influenced by previous level $B_{i-1}$and velocity V_{i-1}.
- ▷ Battery meter M_i is only influenced by Battery level B_i.

Stationary Markov Processes as Transition Models

Remark 5.1.12. Given a stochastic process with state variables X_t and evidence variables E_t, then $\mathbb{P}(X_t|\mathbf{X}_{0:t})$ is a transition model and $\mathbb{P}(E_t|\mathbf{X}_{0:t},\mathbf{E}_{1:t-1})$ a sensor model in the sense of a model-based agent.

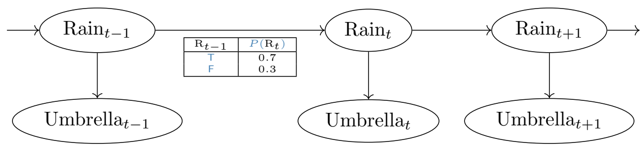
Note that we assume that the X_t do not depend on the E_t.

Also note that with the Markov property, the transition model simplifies to $\mathbb{P}(\mathbf{X}_t|\mathbf{X}_{t-n})$.

Problem: Even with the Markov property the transition model is infinite. ($t \in \mathbb{N}$)

Definition 5.1.13. A Markov chain is called stationary if the transition model is independent of time, i.e. $\mathbb{P}(X_t|X_{t-1})$ is the same for all t.

Example 5.1.14 (Umbrellas are stationary). $\mathbb{P}(\mathtt{Rain}_t|\mathtt{Rain}_{t-1})$ does not depend on t. (need only one table)



⚠ Don't confuse "stationary" (Markov processes) with "static" (environments).

We restrict ourselves to stationary Markov processes in AI-2.

Markov Sensor Models

Recap: The sensor model $\mathbb{P}(E_t|\mathbf{X}_{0:t},\mathbf{E}_{1:t-1})$ allows us (using Bayes rule et al) to update our belief state about X_t given the observations $\mathbf{E}_{0:t}$.

Problem: The evidence variables E_t could depend on any of the variables $\mathbf{X}_{0:t},\mathbf{E}_{1:t-1}$...

Definition 5.1.15. We say that a sensor model has the sensor Markov property, iff $\mathbb{P}(E_t|\mathbf{X}_{0:t},\mathbf{E}_{1:t-1}) = \mathbb{P}(E_t|X_t)$ – i.e., the sensor model depends only on the current state.

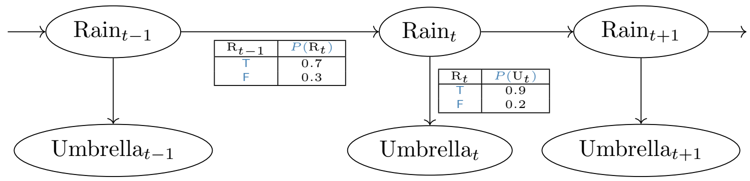
Assumptions on Sensor Models: We usually assume the sensor Markov property and make it stationary as well: $\mathbb{P}(E_t|X_t)$ is fixed for all t.

Definition 5.1.16 (Note).

- ▷ If a Markov chain X is stationary and discrete, we can represent the transition model as a matrix $\mathbf{T}_{ij}:=P(X_t=j|X_{t-1}=i)$.
- ▷ If a sensor model has the sensor Markov property, we can represent each observation $E_t=e_t$ at time t as the diagonal matrix $\mathbf{O}_t$ with $\mathbf{O}_{tii}:=P(E_t=e_t|X_t=i)$.
- ▷ A pair $\langle X,E\rangle$ where X is a (stationary) Markov chains, E_i only depends on X_i, and E has the sensor Markov property is called a (stationary) Hidden Markov Model (HMM). (X and E are single variables)

Umbrellas, the full Story

Example 5.1.17 (Umbrellas, Transition & Sensor Models).



This is a hidden Markov model

Observation 5.1.18. *If we know the initial prior probabilities* $\mathbb{P}(X_0)$ *(*$\widehat{=}$ *time* $t = 0$*), then we can compute the full joint probability distribution as*

$$\mathbb{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbb{P}(X_0) \cdot \prod_{i=1}^{t} \mathbb{P}(X_i | X_{i-1}) \cdot \mathbb{P}(E_i | X_i)$$

5.2 Inference: Filtering, Prediction, and Smoothing

Inference tasks

Definition 5.2.1. Given a Markov process with state variables X_t and evidence variables E_t, we are interested in the following Markov inference tasks:

- ▷ Filtering (or monitoring) $\mathbb{P}(X_t | E_{1:t}^{=e})$: Given the sequence of observations up until time t, compute the likely state of the world at *current* time t.
- ▷ Prediction (or state estimation) $\mathbb{P}(X_{t+k} | E_{1:t}^{=e})$ for $k > 0$: Given the sequence of observations up until time t, compute the likely *future* state of the world at time $t + k$.
- ▷ Smoothing (or hindsight) $\mathbb{P}(X_{t-k} | E_{1:t}^{=e})$ for $0 < k < t$: Given the sequence of observations up until time t, compute the likely *past* state of the world at time $t - k$.
- ▷ Most likely explanation $\underset{\mathbf{x}_{1:t}}{\operatorname{argmax}} (P(X_{1:t}^{=x} | E_{1:t}^{=e}))$: Given the sequence of observations up until time t, compute the most likely sequence of states that led to these observations.

  Note: The most likely sequence of states is *not* (necessarily) the sequence of most likely states ;-)

In this section, we assume X and E to represent *multiple* variables, where X jointly forms a Markov chain and the E jointly have the sensor Markov property.

In the case where X and E are stationary *single* variables, we have a stationary hidden Markov model and can use the matrix forms.

Filtering (Computing the Belief State given Evidence)

Note:

- ▷ Using the full joint probability distribution, we can compute any conditional probability we want, but not necessarily efficiently.
- ▷ We want to use filtering to update our "world model" $\mathbb{P}(X_t)$ based on a new observation $E_t = e_t$ and our *previous* world model $\mathbb{P}(X_{t-1})$.

⇒ We want a function $\mathbb{P}(X_t|E_{1:t}^{=e}) = F(e_t, \underbrace{\mathbb{P}(X_{t-1}|E_{1:t-1}^{=e})}_{F(e_{t-1},\ldots)})$

Spoiler:

$$F(e_t, \mathbb{P}(X_{t-1}|E_{1:t-1}^{=e})) = \alpha(\mathbf{O}_t \cdot \mathbf{T}^T \cdot \mathbb{P}(X_{t-1}|E_{1:t-1}^{=e}))$$

Filtering Derivation

$$\begin{aligned}
\mathbb{P}(X_t|E_{1:t}^{=e}) &= \mathbb{P}(X_t|E_t = e_t, E_{1:t-1}^{=e}) && \text{(dividing up evidence)}\\
&= \alpha(\mathbb{P}(E_t = e_t|X_t, E_{1:t-1}^{=e}) \cdot \mathbb{P}(X_t|E_{1:t-1}^{=e})) && \text{(using Bayes' rule)}\\
&= \alpha(\mathbb{P}(E_t = e_t|X_t) \cdot \mathbb{P}(X_t|E_{1:t-1}^{=e})) && \text{(sensor Markov property)}\\
&= \alpha(\mathbb{P}(E_t = e_t|X_t) \cdot (\sum_{x\in\mathbf{dom}(X)} \mathbb{P}(X_t|X_{t-1} = x, E_{1:t-1}^{=e}) \cdot P(X_{t-1} = x|E_{1:t-1}^{=e}))) && \text{(marginalization)}\\
&= \alpha(\underbrace{\mathbb{P}(E_t = e_t|X_t)}_{\text{sensor model}} \cdot (\sum_{x\in\mathbf{dom}(X)} \underbrace{\mathbb{P}(X_t|X_{t-1} = x)}_{\text{transition model}} \cdot \underbrace{P(X_{t-1} = x|E_{1:t-1}^{=e})}_{\text{recursive call}})) && \text{(conditional independence)}
\end{aligned}$$

Reminder: In a stationary HMM, we have the matrices $\mathbf{T}_{ij} = P(X_t = j|X_{t-1} = i)$ and $\mathbf{O}_{t\,ii} = P(E_t = e_t|X_t = i)$.

Then interpreting $\mathbb{P}(X_{t-1}|E_{1:t-1}^{=e})$ as a vector, the above corresponds exactly to the matrix multiplication $\alpha(\mathbf{O}_t \cdot \mathbf{T}^T \cdot \mathbb{P}(X_{t-1}|E_{1:t-1}^{=e}))$

Definition 5.2.2. We call the inner part of the above expression the forward algorithm, i.e. $\mathbb{P}(X_t|E_{1:t}^{=e}) = \alpha(\text{FORWARD}(e_t, \mathbb{P}(X_{t-1}|E_{1:t-1}^{=e}))) =: \mathbf{f}_{1:t}$.

Filtering the Umbrellas

Example 5.2.3. Let's assume:

- ▷ $\mathbb{P}(\mathtt{R}_0) = \langle 0.5, 0.5\rangle$, (Note that with growing t (and evidence), the impact of the prior at $t = 0$ vanishes anyway)
- ▷ $P(\mathtt{R}_{t+1}|\mathtt{R}_t) = 0.6$, $P(\neg\mathtt{R}_{t+1}|\neg\mathtt{R}_t) = 0.8$, $P(\mathtt{U}_t|\mathtt{R}_t) = 0.9$ and $P(\neg\mathtt{U}_t|\neg\mathtt{R}_t) = 0.85$

⇒ $\mathbf{T} = \begin{pmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{pmatrix}$

- ▷ The director carries an umbrella on days 1 and 2, and *not* on day 3.

⇒ $\mathbf{O}_1 = \mathbf{O}_2 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.1 \end{pmatrix}$ and $\mathbf{O}_3 = \begin{pmatrix} 0.15 & 0 \\ 0 & 0.85 \end{pmatrix}$.

Then:

- ▷ $\mathbf{f}_{1:1} := \mathbb{P}(\mathtt{R}_1|\mathtt{U}_1 = \mathsf{T}) = \alpha(\mathbb{P}(\mathtt{U}_1 = \mathsf{T}|\mathtt{R}_1) \cdot (\sum_{b\in\{\mathsf{T},\mathsf{F}\}} \mathbb{P}(\mathtt{R}_1|\mathtt{R}_0 = b) \cdot P(\mathtt{R}_0 = b)))$

  $= \alpha(\langle 0.9, 0.1\rangle \cdot (\langle 0.6, 0.4\rangle \cdot 0.5 + \langle 0.2, 0.8\rangle \cdot 0.5)) = \alpha(\langle 0.36, 0.06\rangle) = \langle 0.857, 0.143\rangle$

▷ Using matrices: $\alpha(\mathbf{O}_1 \cdot \mathbf{T}^T \cdot \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}) = \alpha(\begin{pmatrix} 0.9 & 0 \\ 0 & 0.1 \end{pmatrix} \cdot \begin{pmatrix} 0.6 & 0.2 \\ 0.4 & 0.8 \end{pmatrix} \cdot \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix})$

$=\alpha(\begin{pmatrix} 0.9 \cdot 0.6 & 0.9 \cdot 0.2 \\ 0.1 \cdot 0.4 & 0.1 \cdot 0.8 \end{pmatrix} \cdot \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}) = \alpha(\begin{pmatrix} 0.9 \cdot 0.6 \cdot 0.5 + 0.9 \cdot 0.2 \cdot 0.5 \\ 0.1 \cdot 0.4 \cdot 0.5 + 0.1 \cdot 0.8 \cdot 0.5 \end{pmatrix}) = \alpha(\begin{pmatrix} 0.36 \\ 0.06 \end{pmatrix})$

Filtering the Umbrellas (Continued)

Example 5.2.4. $\mathbf{f}_{1:1} := \mathbb{P}(\mathtt{R}_1|\mathtt{U}_1 = \mathsf{T}) = \langle 0.857, 0.143\rangle$

▷ $\mathbf{f}_{1:2} := \mathbb{P}(\mathtt{R}_2|\mathtt{U}_2 = \mathsf{T}, \mathtt{U}_1 = \mathsf{T}) = \alpha(\mathbf{O}_2 \cdot \mathbf{T}^T \cdot \mathbf{f}_{1:1}) = \alpha(\mathbb{P}(\mathtt{U}_2 = \mathsf{T}|\mathtt{R}_2) \cdot (\sum_{b\in\{\mathsf{T},\mathsf{F}\}} \mathbb{P}(\mathtt{R}_2|\mathtt{R}_1 = b) \cdot \mathbf{f}_{1:1}(b)))$

$=\alpha(\langle 0.9, 0.1\rangle \cdot (\langle 0.6, 0.4\rangle \cdot 0.857 + \langle 0.2, 0.8\rangle \cdot 0.143)) = \alpha(\langle 0.489, 0.046\rangle) = \langle 0.91, 0.09\rangle$

▷ $\mathbf{f}_{1:3} := \mathbb{P}(\mathtt{R}_3|\mathtt{U}_3 = \mathsf{F}, \mathtt{U}_2 = \mathsf{T}, \mathtt{U}_1 = \mathsf{T}) = \alpha(\mathbf{O}_3 \cdot \mathbf{T}^T \cdot \mathbf{f}_{1:2})$

$=\alpha(\mathbb{P}(\mathtt{U}_3 = \mathsf{F}|\mathtt{R}_3) \cdot (\sum_{b\in\{\mathsf{T},\mathsf{F}\}} \mathbb{P}(\mathtt{R}_3|\mathtt{R}_2 = b) \cdot \mathbf{f}_{1:2}(b)))$

$=\alpha(\langle 0.15, 0.85\rangle \cdot (\langle 0.6, 0.4\rangle \cdot 0.91 + \langle 0.2, 0.8\rangle \cdot 0.09)) = \alpha(\langle 0.085, 0.37\rangle) = \langle 0.187, 0.813\rangle$

Prediction in Markov Chains

Prediction: $\mathbb{P}(X_{t+k}|E_{1:t}^{=e})$ for $k > 0$.

Intuition: Prediction is filtering without new evidence – i.e. we can use filtering until t, and then continue as follows:

Lemma 5.2.5. *By the same reasoning as filtering:*

$$\mathbb{P}(X_{t+k+1}|E_{1:t}^{=e}) = \sum_{x\in\mathbf{dom}(X)} \underbrace{\mathbb{P}(X_{t+k+1}|X_{t+k} = x)}_{\text{transition model}} \cdot \underbrace{P(X_{t+k} = x|E_{1:t}^{=e})}_{\text{recursive call}} \underbrace{= \mathbf{T}^T \cdot \mathbb{P}(X_{t+k} = x|E_{1:t}^{=e})}_{\text{HMM}}$$

Observation 5.2.6. *As $k \to \infty$, $\mathbb{P}(X_{t+k}|E_{1:t}^{=e})$ converges towards a fixed point called the stationary distribution of the Markov chain. (which we can compute from the equation $S = \mathbf{T}^T \cdot S$)*

⇒ the impact of the evidence vanishes.

⇒ The stationary distribution only depends on the transition model.

⇒ There is a small window of time (depending on the transition model) where the evidence has enough impact to allow for prediction beyond *the mere stationary distribution, called the mixing time of the Markov chain.*

⇒ Predicting the future is difficult, and the further into the future, the more difficult it is (Who knew...)

Smoothing

Smoothing: $\mathbb{P}(X_{t-k}|E_{1:t}^{=e})$ for $k > 0$.

Intuition: Use filtering to compute $\mathbb{P}(X_t|E_{1:t-k}^{=e})$, then recurse *backwards* from t until $t - k$.

$$\begin{aligned}
\mathbb{P}(X_{t-k}|E_{1:t}^{=e}) &= \mathbb{P}(X_{t-k}|E_{t-(k-1):t}^{=e}, E_{1:t-k}^{=e}) && \text{(Divide the evidence)}\\
&= \alpha(\mathbb{P}(E_{t-(k-1):t}^{=e}|X_{t-k}, E_{1:t-k}^{=e}) \cdot \mathbb{P}(X_{t-k}|E_{1:t-k}^{=e})) && \text{(Bayes Rule)}\\
&= \alpha(\underbrace{\mathbb{P}(E_{t-(k-1):t}^{=e}|X_{t-k})}_{=:\mathbf{b}_{t-(k-1):t}} \cdot \underbrace{\mathbb{P}(X_{t-k}|E_{1:t-k}^{=e})}_{=\mathbf{f}_{1:t-k}}) && \text{(cond. independence)}\\
&= \alpha(\mathbf{f}_{1:t-k} \times \mathbf{b}_{t-(k-1):t})
\end{aligned}$$

(where $\times$ denotes component-wise multiplication)

Smoothing (continued)

Definition 5.2.7 (Backward message). $\mathbf{b}_{t-k:t} = \mathbb{P}(E_{t-k:t}^{=e}|X_{t-(k+1)})$

$$\begin{aligned}
&= \sum_{x\in\mathbf{dom}(X)} \mathbb{P}(E_{t-k:t}^{=e}|X_{t-k}=x, X_{t-(k+1)}) \cdot \mathbb{P}(X_{t-k}=x|X_{t-(k+1)})\\
&= \sum_{x\in\mathbf{dom}(X)} P(E_{t-k:t}^{=e}|X_{t-k}=x) \cdot \mathbb{P}(X_{t-k}=x|X_{t-(k+1)})\\
&= \sum_{x\in\mathbf{dom}(X)} P(E_{t-k}=e_{t-k}, E_{t-(k-1):t}^{=e}|X_{t-k}=x) \cdot \mathbb{P}(X_{t-k}=x|X_{t-(k+1)})\\
&= \sum_{x\in\mathbf{dom}(X)} \underbrace{P(E_{t-k}=e_{t-k}|X_{t-k}=x)}_{\text{sensor model}} \cdot \underbrace{P(E_{t-(k-1):t}^{=e}|X_{t-k}=x)}_{=\mathbf{b}_{t-(k-1):t}} \cdot \underbrace{\mathbb{P}(X_{t-k}=x|X_{t-(k+1)})}_{\text{transition model}}
\end{aligned}$$

Note: in a stationary hidden Markov model, we get the matrix formulation $\mathbf{b}_{t-k:t} = \mathbf{T}\cdot\mathbf{O}_{t-k}\cdot\mathbf{b}_{t-(k-1):t}$

Definition 5.2.8. We call the associated algorithm the backward algorithm, i.e. $\mathbb{P}(X_{t-k}|E_{1:t}^{=e}) = \alpha(\underbrace{\mathrm{FORWARD}(e_{t-k}, \mathbf{f}_{1:t-(k+1)})}_{\mathbf{f}_{1:t-k}} \times \underbrace{\mathrm{BACKWARD}(e_{t-(k-1)}, \mathbf{b}_{t-(k-2):t})}_{\mathbf{b}_{t-(k-1):t}})$.

As a starting point for the recursion, we let $\mathbf{b}_{t+1:t}$ the uniform vector with 1 in every component.

Smoothing example

Example 5.2.9 (Smoothing Umbrellas). Reminder: We assumed $\mathbb{P}(\mathtt{R}_0) = \langle 0.5, 0.5\rangle$, $P(\mathtt{R}_{t+1}|\mathtt{R}_t) = 0.6$, $P(\neg\mathtt{R}_{t+1}|\neg\mathtt{R}_t) = 0.8$, $P(\mathtt{U}_t|\mathtt{R}_t) = 0.9$, $P(\neg\mathtt{U}_t|\neg\mathtt{R}_t) = 0.85$

$\Rightarrow \mathbf{T} = \begin{pmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{pmatrix}$, $\mathbf{O}_1 = \mathbf{O}_2 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.1 \end{pmatrix}$ and $\mathbf{O}_3 = \begin{pmatrix} 0.15 & 0 \\ 0 & 0.85 \end{pmatrix}$. (The director carries an umbrella on days 1 and 2, and *not* on day 3)

$\mathbf{f}_{1:1} = \langle 0.857, 0.143\rangle$, $\mathbf{f}_{1:2} = \langle 0.91, 0.09\rangle$ and $\mathbf{f}_{1:3} = \langle 0.187, 0.813\rangle$

Let's compute

$$\mathbb{P}(\mathtt{R}_1|\mathtt{U}_1 = \mathsf{T}, \mathtt{U}_2 = \mathsf{T}, \mathtt{U}_3 = \mathsf{F}) = \alpha(\mathbf{f}_{1:1} \times \mathbf{b}_{2:3})$$

▷ We need to compute $\mathbf{b}_{2:3}$ and $\mathbf{b}_{3:3}$:

▷ $\mathbf{b}_{3:3} = \mathbf{T}\cdot\mathbf{O}_3\cdot\mathbf{b}_{4:3} = \begin{pmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{pmatrix} \cdot \begin{pmatrix} 0.15 & 0 \\ 0 & 0.85 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.43 \\ 0.71 \end{pmatrix}$

▷ $\mathbf{b}_{2:3} = \mathbf{T} \cdot \mathbf{O}_2 \cdot \mathbf{b}_{3:3} = \begin{pmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{pmatrix} \cdot \begin{pmatrix} 0.9 & 0 \\ 0 & 0.1 \end{pmatrix} \cdot \begin{pmatrix} 0.43 \\ 0.71 \end{pmatrix} = \begin{pmatrix} 0.261 \\ 0.134 \end{pmatrix}$

$\Rightarrow \alpha(\begin{pmatrix} 0.857 \\ 0.143 \end{pmatrix} \times \begin{pmatrix} 0.261 \\ 0.134 \end{pmatrix}) = \alpha(\begin{pmatrix} 0.224 \\ 0.02 \end{pmatrix}) = \begin{pmatrix} 0.918 \\ 0.082 \end{pmatrix}$

⇒ Given the evidence $U_2, \neg U_3$, the posterior probability for R_1 went up from 0.857 to 0.918!

Forward/Backward Algorithm for Smoothing

Definition 5.2.10. Forward backward algorithm: returns the sequence of posterior distributions $\mathbb{P}(X_1) \ldots \mathbb{P}(X_t)$ given evidence $e_1, \ldots, e_t$:

```
function FORWARD-BACKWARD(⟨e_1,...,e_t⟩, P(X_0))
    f := ⟨P(X_0)⟩
    b := ⟨1, 1, ...⟩
    S := ⟨P(X_0)⟩
    for i = 1,...,t do
        f_i := FORWARD(f_{i-1}, e_i)                /* filtering */
    for i = t,...,1 do
        S_i := α(f_i × b)                           /* smoothing */
        b := BACKWARD(b, e_i)
    return S
```

(Note the discrepancy wrt normalization between the derivation and the algorithm... why is this okay? ;))

Time complexity linear in t (polytree inference), Space complexity $\mathcal{O}(t \cdot |\mathbf{f}|)$.

Country dance algorithm

Idea: If $\mathbf{T}$ and $\mathbf{O}_i$ are invertible, we can avoid storing all forward messages in the smoothing algorithm by running filtering backwards:

$$\mathbf{f}_{1:i+1} = \alpha(\mathbf{O}_{i+1} \cdot \mathbf{T}^T \cdot \mathbf{f}_{1:i})$$

$$\Rightarrow \mathbf{f}_{1:i} = \alpha(\mathbf{T}^{T^{-1}} \cdot \mathbf{O}_{i+1}{}^{-1} \cdot \mathbf{f}_{1:i+1})$$

⇒ we can trade space complexity for time complexity:

▷ In the first for-loop, we only compute the final $\mathbf{f}_{1:t}$ (No need to store the intermediate results)

▷ In the second for-loop, we compute both $\mathbf{f}_{1:i}$ and $\mathbf{b}_{t-i:t}$ (Only one copy of $\mathbf{f}_{1:i}$, $\mathbf{b}_{t-i:t}$ is stored)

⇒ constant space.

But: Requires that both matrices are invertible, i.e. *every observation must be possible in every state.* (Possible hack: increase the probabilities of 0 to "negligibly small")

Most Likely Explanation

Smoothing allows us to compute the *sequence of most likely states* $X_1, \ldots, X_t$ given $E_{1:t}^{=e}$. What if we want the *most likely sequence* of states? i.e. $\max_{x_1,\ldots,x_t}(P(X_{1:t}^{=x}|E_{1:t}^{=e}))$?

Example 5.2.11. Given the sequence $\mathtt{U}_1, \mathtt{U}_2, \neg\mathtt{U}_3, \mathtt{U}_4, \mathtt{U}_5$, the most likely state for $\mathtt{R}_3$ is F, but the most likely sequence *might* be that it rained throughout...

Prominent Application: In speech recognition, we want to find the most likely word sequence, given what we have heard. (can be quite noisy)

Idea:

- ▷ For every $x_t \in \mathbf{dom}(X)$ and $0 \leq i \leq t$, recursively compute the most likely path $X_1, \ldots, X_i$ ending in $X_i = x_i$ given the observed evidence.
- ▷ remember the x_{i-1} that most likely leads to x_i.
- ▷ Among the resulting paths, pick the one to *the* $X_t = x_t$ with the most likely path,
- ▷ and then recurse backwards.

$\Rightarrow$ we want to know $\max_{x_1,\ldots,x_{t-1}} \mathbb{P}(X_{1:t-1}^{=x}, X_t|E_{1:t}^{=e})$, and then pick the x_t with the maximal value.

Most Likely Explanation (continued)

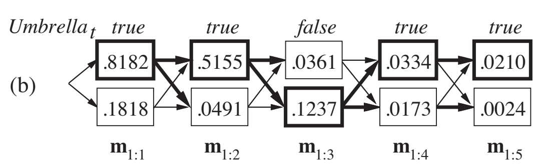
By the same reasoning as for filtering:

$$\max_{x_1,\ldots,x_{t-1}} \mathbb{P}(X_{1:t-1}^{=x}, X_t|E_{1:t}^{=e})$$

$$= \alpha(\underbrace{\mathbb{P}(E_t = e_t|X_t)}_{\text{sensor model}} \cdot \max_{x_{t-1}}(\underbrace{\mathbb{P}(X_t|X_{t-1} = x_{t-1})}_{\text{transition model}} \cdot \underbrace{\max_{x_1,\ldots,x_{t-2}}(P(X_{1:t-2}^{=x}, X_{t-1} = x_{t-1}|E_{1:t-1}^{=e}))}_{=:\mathbf{m}_{1:t-1}(x_{t-1})}))$$

$\mathbf{m}_{1:t}(i)$ gives the maximal probability that the most likely path up to t leads to state $X_t = i$. Note that we can leave out the α, since we're only interested in the maximum.

Example 5.2.12. For the sequence [T, T, F, T, T]:



bold arrows: best predecessor measured by "best preceding sequence probability × transition probability"

The Viterbi Algorithm

Definition 5.2.13. The Viterbi algorithm now proceeds as follows:

function $\textsc{Viterbi}(\langle e_1,\ldots,e_t\rangle, \mathbb{P}(X_0))$
- $m := \langle \mathbb{P}(X_0) \rangle$ /* $\mathbf{m}_{1:i}$ */
- $\mathrm{prev} := \langle\rangle$ /* *the most likely predecessor of each possible* x_i */
- **for** $i = 1,\ldots,t$ **do**
  - $m_i := \max\limits_{x_{i-1}} (\mathbb{P}(E_i = e_i|X_i) \cdot \mathbb{P}(X_i|X_{i-1} = x_{i-1}) \cdot m_{i-1}(x_{i-1}))$
  - $\mathrm{prev}_i := \operatorname{argmax}\limits_{x_{i-1}} (\mathbb{P}(E_i = e_i|X_i) \cdot \mathbb{P}(X_i|X_{i-1} = x_{i-1}) \cdot m_{i-1}(x_{i-1}))$
- $P := \langle 0, 0, \ldots, \max\limits_{(x \in \mathbf{dom}(X))} \mathrm{prev}_t(vx) \rangle$
- **for** $i = t-1,\ldots,1$ **do**
  - $P_i := mx_i(P_{i+1})$
- **return** P

Observation 5.2.14. *Viterbi has linear time complexity and linear space complexity (needs to keep the most likely sequence leading to each state).*

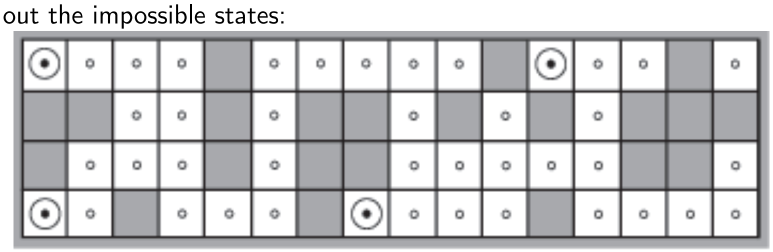
5.3 Hidden Markov Models – Extended Example

Example: Robot Localization using Common Sense

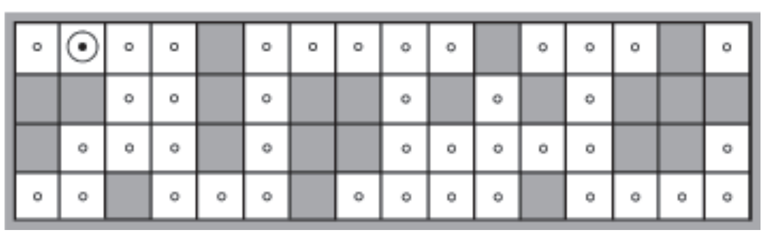
Example 5.3.1 (Robot Localization in a Maze). A robot has four sonar sensors that tell it about obstacles in four directions: N, S, W, E.

We write the result where the sensor that detects obstacles in the north, south, and east as N S E.

We filter out the impossible states:



a) Possible robot locations after $\mathrm{e}_1 = \mathrm{N\ S\ W}$



b) Possible robot locations after $\mathrm{e}_1 = \mathrm{N\ S\ W}$ and $\mathrm{e}_2 = \mathrm{N\ S}$

Remark 5.3.2. This only works for perfect sensors. (else no impossible states)

What if our sensors are imperfect?

HMM Example: Robot Localization (Modeling)

Example 5.3.3 (HMM-based Robot Localization). We have the following setup:

▷ A hidden Random variable X_t for robot location (domain: 42 empty squares)

▷ Let $N(i)$ be the set of neighboring fields of the field $X_i = x_i$

▷ The Transition matrix for the move action (**T** has $42^2 = 1764$ entries)

$$P(X_{t+1} = j | X_t = i) = \mathbf{T}_{ij} = \begin{cases} \frac{1}{|N(i)|} & \text{if } j \in N(i) \\ 0 & \text{else} \end{cases}$$

▷ We do not know where the robot starts: $P(X_0) = \frac{1}{n}$ (here $n = 42$)

▷ Evidence variable E_t: four bit presence/absence of obstacles in N, S, W, E. Let d_{it} be the number of wrong bits and ϵ the error rate of the sensor. Then

$$P(E_t = e_t | X_t = i) = \mathbf{O}_{tii} = (1-\epsilon)^{4-d_{it}} \cdot \epsilon^{d_{it}}$$

(We assume the sensors are independent)

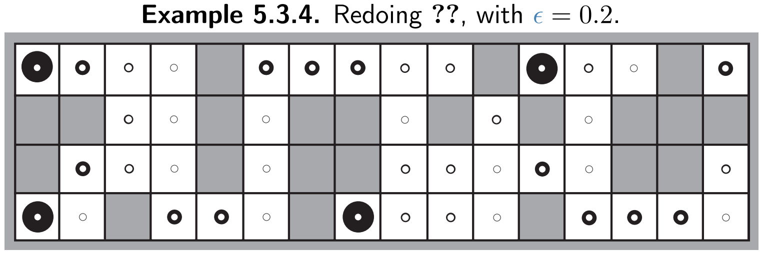
For example, the probability that the sensor on a square with obstacles in north and south would produce N S E is $(1-\epsilon)^3 \cdot \epsilon^1$.

We can now use filtering for localization, smoothing to determine e.g. the starting location, and the Viterbi algorithm to find out how the robot got to where it is now.

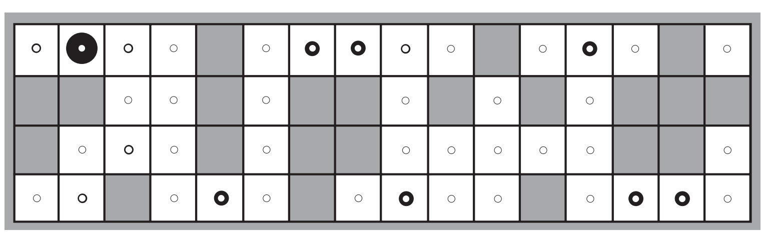
HMM Example: Robot Localization

We use HMM filtering equation $\mathbf{f}_{1:t+1} = \alpha \cdot \mathbf{O}_{t+1}\mathbf{T}^t\mathbf{f}_{1:t}$ to compute posterior distribution over locations. (i.e. robot localization)

Example 5.3.4. Redoing ??, with $\epsilon = 0.2$.



a) Posterior distribution over robot location after $E_1 = N\ S\ W$



b) Posterior distribution over robot location after $E_1 = N\ S\ W$ and $E_2 = N\ S$

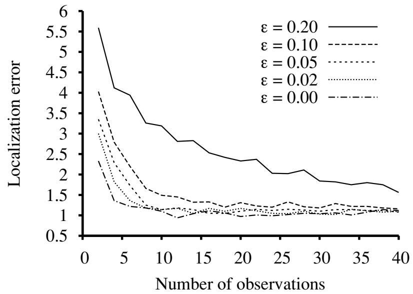
Still the same locations as in the "perfect sensing" case, but now other locations have non-zero probability.

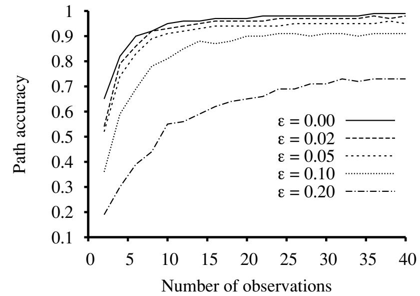
HMM Example: Further Inference Applications

Idea: We can use smoothing: $\mathbf{b}_{k+1:t} = \mathbf{T}\mathbf{O}_{k+1}\mathbf{b}_{k+2:t}$ to find out where it started and the

Viterbi algorithm to find the most likely path it took.

Example 5.3.5. Performance of HMM localization vs. observation length (various error rates ϵ)





Localization error (Manhattan distance from true location)

Viterbi path accuracy (fraction of correct states on Viterbi path)

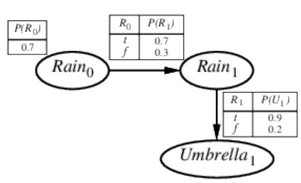
5.4 Dynamic Bayesian Networks

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/30355`.

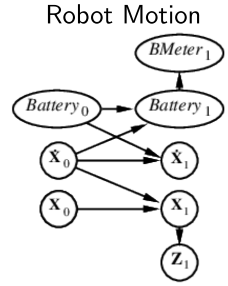
Dynamic Bayesian networks

- **Definition 5.4.1.** A Bayesian network $\mathcal{D}$ is called dynamic (a DBN), iff its random variables are indexed by a time structure. We assume that $\mathcal{D}$ is
  - time sliced, i.e. that the time slices $\mathcal{D}_t$ – the subgraphs of t-indexed random variables and the edges between them – are isomorphic.
  - a stationary Markov chain, i.e. that variables X_t can only have parents in $\mathcal{D}_t$ and $\mathcal{D}_{t-1}$.
- $\mathbf{X}_t$, $\mathbf{E}_t$ contain arbitrarily many variables in a replicated Bayesian network.
- **Example 5.4.2.**

Umbrellas



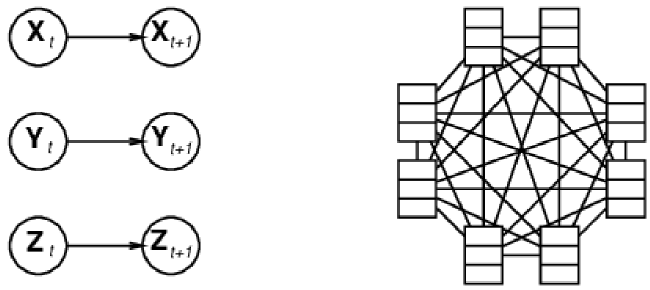
Robot Motion



DBNs vs. HMMs

- **Observation 5.4.3.**

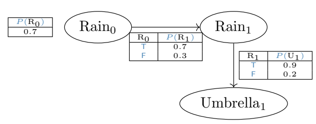
- ▷ *Every* HMM *is a single-variable* DBN. *(trivially)*
- ▷ *Every discrete* DBN *is an* HMM. *(combine variables into tuple)*
- ▷ DBNs *have sparse dependencies* ⇝ *exponentially fewer parameters;*

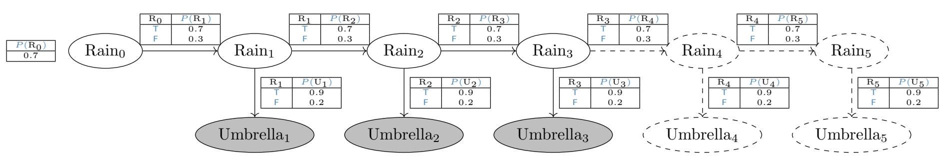


▷ **Example 5.4.4 (Sparse Dependencies).** With 20 Boolean state variables, three parents each, a DBN has $20 \cdot 2^3 = 160$ parameters, the corresponding HMM has $2^{20} \cdot 2^{20} \approx 10^{12}$.

Exact inference in DBNs

▷ **Definition 5.4.5 (Naive method).** Unroll the network and run any exact algorithm.





▷ **Problem:** Inference cost for each update grows with t.

▷ **Definition 5.4.6.** Rollup filtering: add slice $t+1$, "sum out" slice t using variable elimination.

▷ **Observation:** Largest factor is $\mathcal{O}(d^{n+1})$, update cost $\mathcal{O}(d^{n+2})$, where d is the maximal domain size.

▷ **Note:** Much better than the HMM update cost of $\mathcal{O}(d^{2n})$

Summary

▷ Temporal probability models use state and evidence variables replicated over time.

▷ Markov property and stationarity assumption, so we need both

- ▷ a transition model and $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$

  - ▷ a sensor model $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$.
- ▷ Tasks are filtering, prediction, smoothing, most likely sequence; (all done recursively with constant cost per time step)
- ▷ Hidden Markov models have a single discrete state variable; (used for speech recognition)
- ▷ DBNs subsume HMMs, exact update intractable.

Chapter 6

Making Simple Decisions Rationally

6.1 Introduction

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/30338`.

Overview

We now know how to update our world model, represented as (a set of) random variables, given observations. Now we need to *act*.

For that we need to answer two questions:

Questions:

▷ Given a world model and a set of *actions*, what will the likely consequences of each action be?

▷ How "good" are these consequences?

Idea:

▷ Represent actions as "special random variables":

Given disjoint actions $a_1, \ldots, a_n$, introduce a random variable A with domain $\{a_1, \ldots, a_n\}$. Then we can model/query $\mathbb{P}(X|A = a_i)$.

▷ Assign *numerical values* to the outcomes of actions (i.e. a function $u\colon \mathbf{dom}(X) \to \mathbb{R}$).

▷ Choose the action that maximizes the *expected value* of u

Definition 6.1.1. Decision theory investigates decision problems, i.e. how a model-based agent a deals with choosing among actions based on the desirability of their outcomes given by a real-valued utility function u on states $s \in S$: i.e. $u\colon S \to \mathbb{R}$.

Decision Theory

If our states are random variables, then we obtain a random variable for the utility function:

Observation: Let $X_i\colon \Omega \to D_i$ random variables on a probability model $\langle \Omega, P \rangle$ and $f\colon D_1 \times \ldots \times D_n \to E$. Then $F(x) := f(X_0(x), \ldots, X_n(x))$ is a random variable $\Omega \to E$.

Definition 6.1.2. Given a probability model $\langle \Omega, P \rangle$ and a random variable $X\colon \Omega \to D$ with $D \subseteq \mathbb{R}$,

then $E(X) := \sum_{x\in D} P(X = x) \cdot x$ is called the expected value (or expectation) of X.
(Assuming the sum/series is actually defined!)

Analogously, let $e_1, \ldots, e_n$ a sequence of events. Then the expected value of X given $e_1, \ldots, e_n$ is defined as $E(X|e_1, \ldots, e_n) := \sum_{x\in D} P(X = x|e_1, \ldots, e_n) \cdot x$.

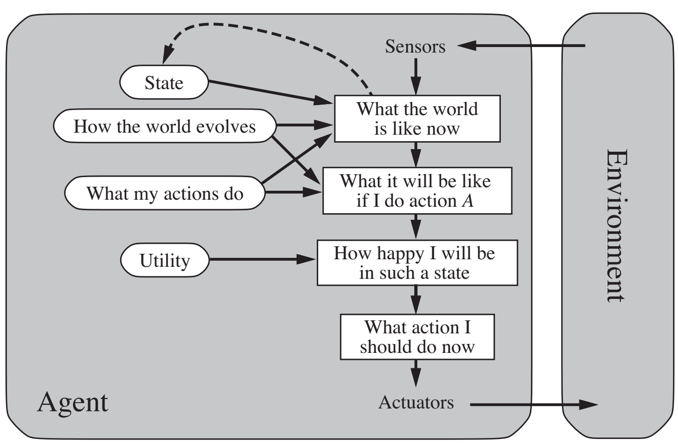
Putting things together:

Definition 6.1.3. Let $A\colon \Omega \to D$ a random variable (where D is a set of actions) $X_i\colon \Omega \to D_i$ random variables (the state), and $u\colon D_1 \times \ldots \times D_n \to \mathbb{R}$ a utility function. Then the expected utility of the action $a \in D$ is the expected value of u (interpreted as a random variable) given $A = a$; i.e.

$$\mathrm{EU}(a) := \sum_{\langle x_1,\ldots,x_n\rangle \in D_1\times\ldots\times D_n} P(X_1 = x_1, \ldots, X_n = x_n|A = a) \cdot u(x_1, \ldots, x_n)$$

Utility-based Agents

- **Definition 6.1.4.** A utility-based agent uses a world model along with a utility function that models its preferences among the states of that world. It chooses the action that leads to the best expected utility.
- **Agent Schema:**



Maximizing Expected Utility (Ideas)

Definition 6.1.5 (MEU principle for Rationality). We call an action rational if it maximizes expected (MEU). An utility-based agent is called rational, iff it always chooses a rational action.

Hooray: This solves all of AI. (in principle)

Problem: There is a long, long way towards an operationalization ;)

Note: An agent can be entirely rational (consistent with MEU) without ever representing or manipulating utilities and probabilities.

Example 6.1.6. A simple reflex agent for tic tac toe based on a perfect lookup table is rational if we take (the negative of) "winning/drawing in n steps" as the utility function.

Example 6.1.7 (AI1). Heuristics in tree search (greedy search, A^*) and game-play (minimax, alpha-beta pruning) maximize "expected" utility.

$\Rightarrow$ In fully observable, deterministic environments, "expected utility" reduces to a specific determined utility value:

$\mathrm{EU}(a) = U(T(S(s,e),a))$, where e the most recent percept, s the current state, S the sensor function and T the transition function.

Now let's figure out how to actually assign utilities!

6.2 Preferences and Utilities

Preferences in Deterministic Environments

Problem: How do we determine the utility of a state? (We cannot directly measure our satisfaction/happiness in a possibly future state...) (What unit would we even use?)

Example 6.2.1. I have to decide whether to go to class today (or sleep in). What is the utility of this lecture? (obviously 42)

Idea: We can let people/agents choose between two states (subjective preference) and derive a utility from these choices.

Example 6.2.2. *Give me your cell-phone or I will give you a bloody nose.* $\leadsto$
To make a decision in a deterministic environment, the agent must determine whether it prefers a state without phone to one with a bloody nose?

Definition 6.2.3. Given states A and B (we call them prizes) and agent can express preferences of the form

- $A \succ B$ A prefered over B
- $A \sim B$ indifference between A and B
- $A \succeq B$ B not prefered over A

i.e. Given a set $\mathcal{S}$ (of states), we define binary relations $\succ$ and $\sim$ on $\mathcal{S}$.

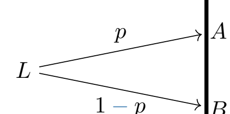
Preferences in Non-Deterministic Environments

Problem: In nondeterministic environments we do not have full information about the states we choose between.

Example 6.2.4 (Airline Food). *Do you want chicken or pasta*(but we cannot see through the tin foil)

Definition 6.2.5.

Let $\mathcal{S}$ a set of states. We call a random variable X with domain $D \subseteq \mathcal{S}$ a lottery and write $[p_1,A_1;\ldots;p_n,A_n]$, where $p_i = P(X = A_i)$.



Idea: A lottery represents the result of a nondeterministic action that can have outcomes A_i with prior probability p_i. For the binary case, we use $[p,A;1-p,B]$. We can then extend preferences to include lotteries, as a measure of how *strongly* we prefer one prize over another.

Convention: We assume $\mathcal{S}$ to be *closed under lotteries*, i.e. lotteries themselves are also states.

That allows us to consider lotteries such as $[p,A;1-p,[q,B;1-q,C]]$.

Rational Preferences

Note: Preferences of a rational agent must obey certain constraints – An agent with *rational* preferences can be described as an MEU-agent.

Definition 6.2.6. We call a set $\succ$ of preferences rational, iff the following constraints hold:

Orderability	$A\succ B \vee B\succ A \vee A\sim B$
Transitivity	$A\succ B \wedge B\succ C \Rightarrow A\succ C$
Continuity	$A\succ B\succ C \Rightarrow (\exists p.[p,A;1-p,C]\sim B)$
Substitutability	$A\sim B \Rightarrow [p,A;1-p,C]\sim[p,B;1-p,C]$
Monotonicity	$A\succ B \Rightarrow (p>q) \Leftrightarrow [p,A;1-p,B]\succ[q,A;1-q,B]$
Decomposability	$[p,A;1-p,[q,B;1-q,C]]\sim[p,A\,;\,((1-p)q),B\,;\,((1-p)(1-q)),C]$

From a set of rational preferences, we can obtain a meaningful utility function.

The rationality constraints can be understood as follows:

Orderability: $A\succ B \vee B\succ A \vee A\sim B$ Given any two prizes or lotteries, a rational agent must either prefer one to the other or else rate the two as equally preferable. That is, the agent cannot avoid deciding. Refusing to bet is like refusing to allow time to pass.

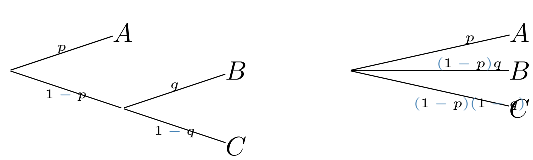
Transitivity: $A\succ B \wedge B\succ C \Rightarrow A\succ C$

Continuity: $A\succ B\succ C \Rightarrow (\exists p.[p,A;1-p,C]\sim B)$ If some lottery B is between A and C in preference, then there is some probability p for which the rational agent will be indifferent between getting B for sure and the lottery that yields A with probability p and C with probability $1-p$.

Substitutability: $A\sim B \Rightarrow [p,A;1-p,C]\sim[p,B;1-p,C]$ If an agent is indifferent between two lotteries A and B, then the agent is indifferent between two more complex lotteries that are the same except that B is substituted for A in one of them. This holds regardless of the probabilities and the other outcome(s) in the lotteries.

Monotonicity: $A\succ B \Rightarrow (p>q) \Leftrightarrow [p,A;1-p,B]\succ[q,A;1-q,B]$ Suppose two lotteries have the same two possible outcomes, A and B. If an agent prefers A to B, then the agent must prefer the lottery that has a higher probability for A (and vice versa).

Decomposability: $[p,A;1-p,[q,B;1-q,C]]\sim[p,A;((1-p)q),B;((1-p)(1-q)),C]$ Compound lotteries can be reduced to simpler ones using the laws of probability. This has been called the "no fun in gambling" rule because it says that two consecutive lotteries can be compressed into a single equivalent lottery: the following two are equivalent:

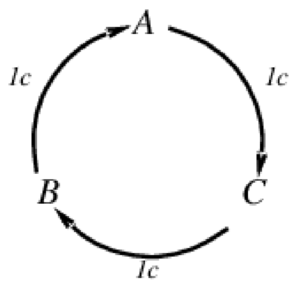


Rational preferences contd.

▷ Violating the rationality constraints from ?? leads to self-evident irrationality.

▷ **Example 6.2.7.** An agent with intransitive preferences can be induced to give away all its money:

- If $B \succ C$, then an agent who has C would pay (say) 1 cent to get B
- If $A \succ B$, then an agent who has B would pay (say) 1 cent to get A
- If $C \succ A$, then an agent who has A would pay (say) 1 cent to get C



6.3 Utilities and Money

Video Nuggets covering this section can be found at `https://fau.tv/clip/id/30341` and `https://fau.tv/clip/id/30342`.

Ramseys Theorem and Value Functions

- **Theorem 6.3.1.** *(Ramsey, 1931; von Neumann and Morgenstern, 1944)*
  Given a rational set of preferences there exists a real valued function U such that $U(A) \geq U(B)$, iff $A \succeq B$ and $U([p_1,S_1\,;\ldots;p_n,S_n]) = \sum_i p_i U(S_i)$
- This is an existence theorem, uniqueness not guaranteed.
- **Note:** Agent behavior is *invariant* w.r.t. positive linear transformations, i.e. an agent with utility function $U'(x) = k_1 U(x) + k_2$ where $k_1 > 0$ behaves exactly like one with U.
- **Observation:** With deterministic prizes only (no lottery choices), only a total ordering on prizes can be determined.
- **Definition 6.3.2.** We call a total ordering on states a value function or ordinal utility function.

Maximizing Expected Utility (Definitions)

- We first formalize the notion of expectation of a random variable.
- **Definition 6.3.3.** Given a probability model $\langle\Omega, P\rangle$ and a random variable $X\colon \Omega \to D$ with $D \subseteq \mathbb{R}$, then $E(X) := \sum_{x \in D} P(X = x) \cdot x$ is called the expected value (or expectation) of X.
- **Idea:** Apply this idea to get the expected utility of an action, this is stochastic:

- ▷ In partially observable environments, we do not know the current state.
- ▷ In nondeterministic environments, we cannot be sure of the result of an action.

▷ **Definition 6.3.4.** Let $\mathcal{A}$ be an agent with a set Ω of states and a utility function $U\colon \Omega \rightarrow \mathbb{R}_0^+$, then for each action a, we define a random variable R_a whose values are the results of performing a in the current state.

▷ **Definition 6.3.5.** The expected utility $\mathrm{EU}(a|\mathbf{e})$ of an action a (given evidence $\mathbf{e}$) is

$$\mathrm{EU}(a|\mathbf{e}) := \sum_{s \in \Omega} P(R_a = s|a, \mathbf{e}) \cdot U(s)$$

Utilities

▷ **Intuition:** Utilities map states to real numbers.

▷ **Question:** Which numbers exactly?

▷ **Definition 6.3.6 (Standard approach to assessment of human utilities).** Compare a given state A to a standard lottery L_p that has

- ▷ "best possible prize" $u_\top$ with probability p
- ▷ "worst possible catastrophe" $u_\bot$ with probability $1-p$

adjust lottery probability p until $A \sim L_p$. Then $U(A) = p$.

▷ **Example 6.3.7.** Choose $u_\top \mathrel{\widehat{=}}$ current state, $u_\bot \mathrel{\widehat{=}}$ instant death

pay \$30 $\sim L$ —0.999999→ continue as before
pay \$30 $\sim L$ —0.000001→ instant death

Measuring Utility

▷ **Definition 6.3.8.** Normalized utilities: $u_\top = 1$, $u_\bot = 0$.

▷ **Definition 6.3.9.** Micromorts: one millionth chance of instant death.

▷ Micromorts are useful for Russian roulette, paying to reduce product risks, etc.

▷ **Problem:** What is the value of a micromort?

▷ **Ask them directly:** What would you pay to avoid playing Russian roulette with a million-barrelled revolver? (very large numbers)

▷ **But their behavior suggests a lower price:**

- ▷ Driving in a car for 370km incurs a risk of one micromort;
- ▷ Over the life of your car – say, $150,000\text{km}$ that's 400 micromorts.
- ▷ People appear to be willing to pay about $10,000$ more for a safer car that halves the risk of death. ($\rightsquigarrow$ 25 per micromort)

▷ This figure has been confirmed across many individuals and risk types.

▷ Of course, this argument holds only for small risks. Most people won't agree to kill themselves for 25M.

▷ **Definition 6.3.10.** QALYs: quality adjusted life years

▷ **Application:** QALYs are useful for medical decisions involving substantial risk.

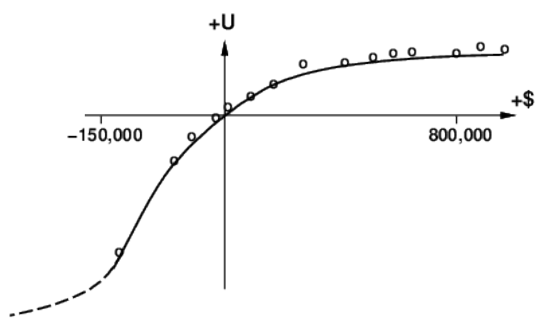
Money vs. Utility

▷ Money does *not* behave as a utility function should.

▷ Given a lottery L with expected monetary value $\text{EMV}(L)$, usually $U(L) < U(\text{EMV}(L))$, i.e., people are risk averse.

▷ **Utility curve:** For what probability p am I indifferent between a prize x and a lottery $[p,M\$;1-p,0\$]$ for large numbers M?

▷ Typical empirical data, extrapolated with risk prone behavior for debitors:



▷ **Empirically:** Comes close to the logarithm on the positive numbers.

6.4 Multi-Attribute Utility

Video Nuggets covering this section can be found at https://fau.tv/clip/id/30343 and https://fau.tv/clip/id/30344.

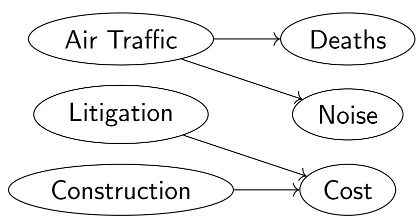
In this section we will make the ideas introduced above more practical. The discussion above conceived utility functions as functions on atomic states, which were good enough for introducing the theory. But when we build decision models for utility-based agent we want to characterize states by attributes that are already random variables in the Bayesian network we use to represent the belief state. For factored states, the utility function can be expressed as a multivariate function on attribute values.

Utility Functions on Attributes

- **Recap:** So far we understand how to obtain utility functions $u\colon S \to \mathbb{R}$ on states $s \in S$ from (rational) preferences.
- **But** in a partially observable, stochastic environment, we cannot know the current state. (utilities/preferences useless?)
- **Idea:** Base utilities/preferences on random variables that we can model.
- **Definition 6.4.1.** Let $X_1, \ldots, X_n$ be random variables with domains $D_1, \ldots, D_n$. Then we call a function $u\colon D_1 \times \ldots \times D_n \to \mathbb{R}$ a (multi-attribute) utility function on attributes $X_1, \ldots, X_n$.
- **Intuition:** Given a probabilistic belief state that includes random variables $X_1, \ldots, X_n$, and a utility function on attributes $X_1, \ldots, X_n$, we can still maximize expected utility! (MEU principle)
- **Preview:** Understand multi attribute utility functions and use Bayesian networks as representations of belief states.

Multi-Attribute Utility: Example

- **Example 6.4.2 (Assessing an Airport Site).**

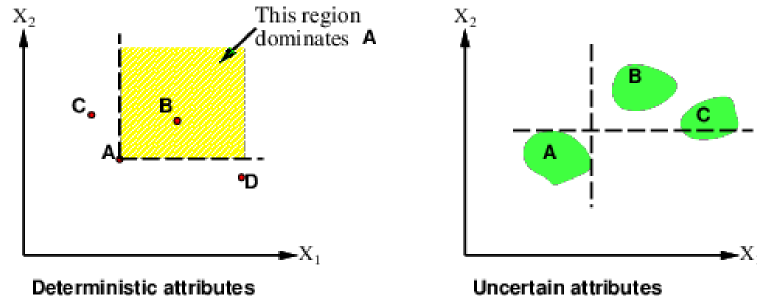


- Attributes: Deaths, Noise, Cost.
- **Question**: What is $U(\text{Deaths}, \text{Noise}, \text{Cost})$ for a projected airport?

- How can complex utility function be assessed from preference behaviour?
- **Idea 1:** Identify conditions under which decisions can be made without complete identification of $U(X_1, \ldots, X_n)$.
- **Idea 2:** Identify various types of *independence* in preferences and derive consequent canonical forms for $U(X_1, \ldots, X_n)$.

Strict Dominance

- Typically define attributes such that U is monotone in each argument. (wlog. growing)
- **Definition 6.4.3.** Choice B strictly dominates choice A iff $X_i(B) \geq X_i(A)$ for all i (and hence $U(B) \geq U(A)$)



▷ **Observation:** Strict dominance seldom holds in practice (life is difficult) but is useful for narrowing down the field of contenders.

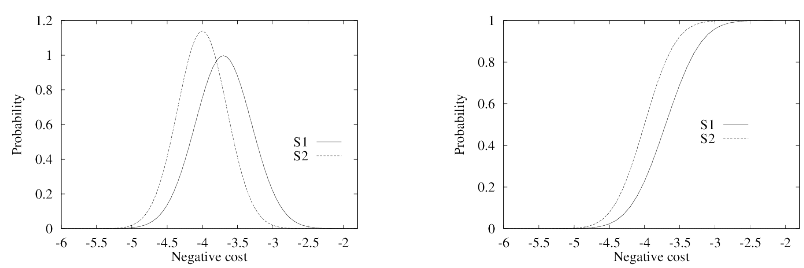
▷ For uncertain attributes strict dominance is even more unlikely.

Stochastic Dominance

▷ **Definition 6.4.4.** A distribution p_2 stochastically dominates distribution p_1 iff the cummulative distribution of p_2 strictly dominates that for p_1 for all t, i.e.

$$\int_t^{-\infty} p_1(x)dx \leq \int_t^{-\infty} p_2(x)dx$$

▷ **Example 6.4.5.** Even if the distributions (left) overlap considerably the cummulative distribution (right) strictly dominates.



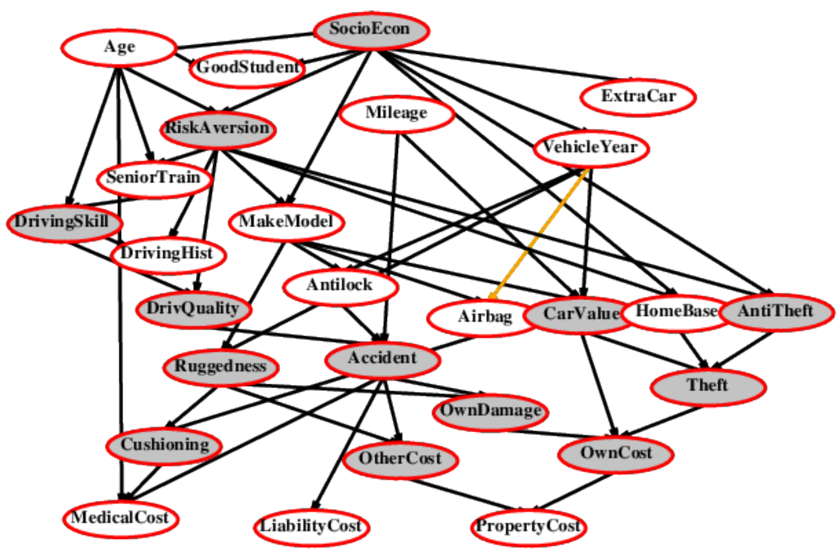
Stochastic dominance contd.

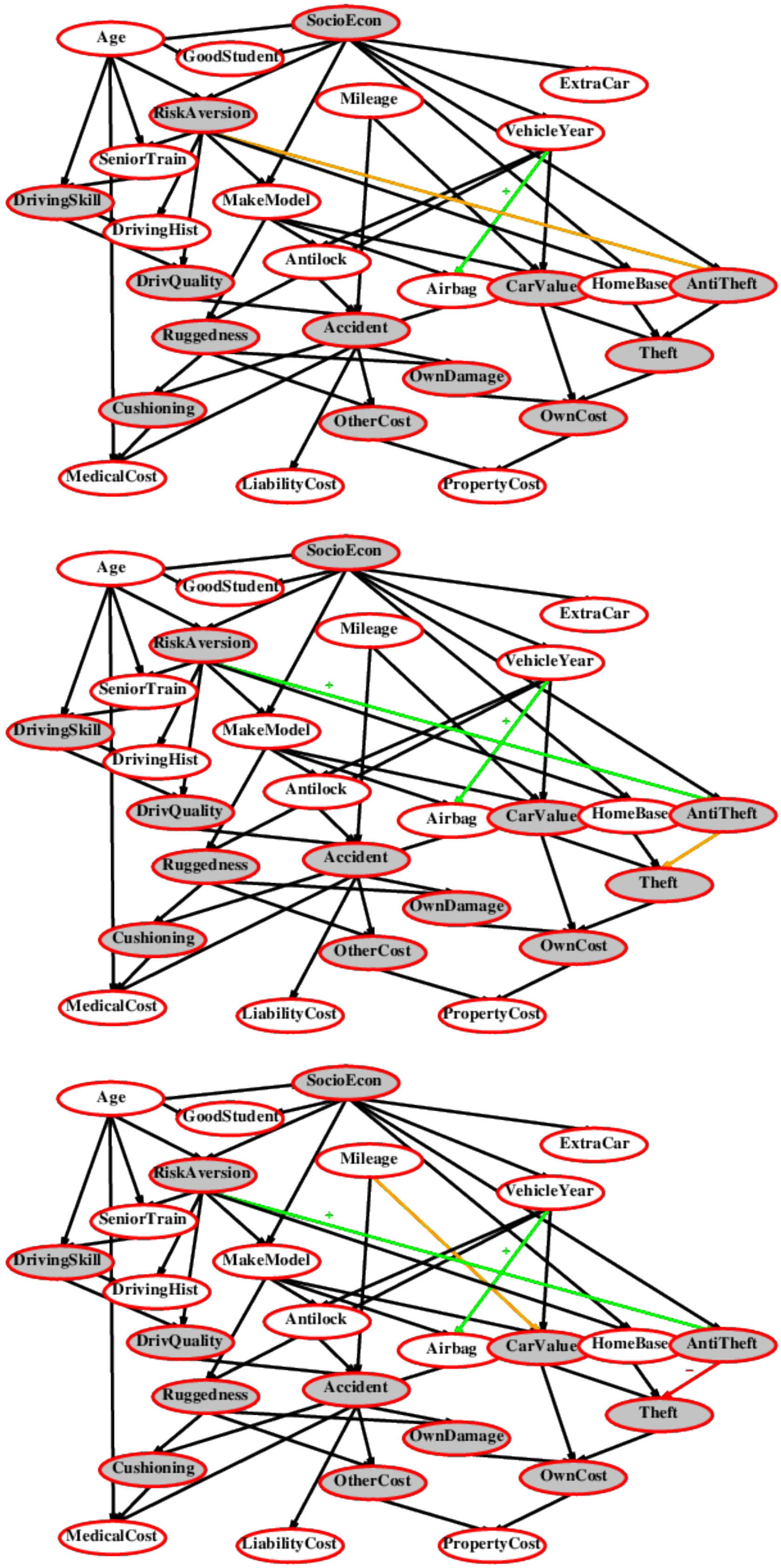
▷ **Observation 6.4.6.** *If U is monotone in x, then A_1 with outcome distribution p_1 stochastically dominates A_2 with outcome distribution p_2:*

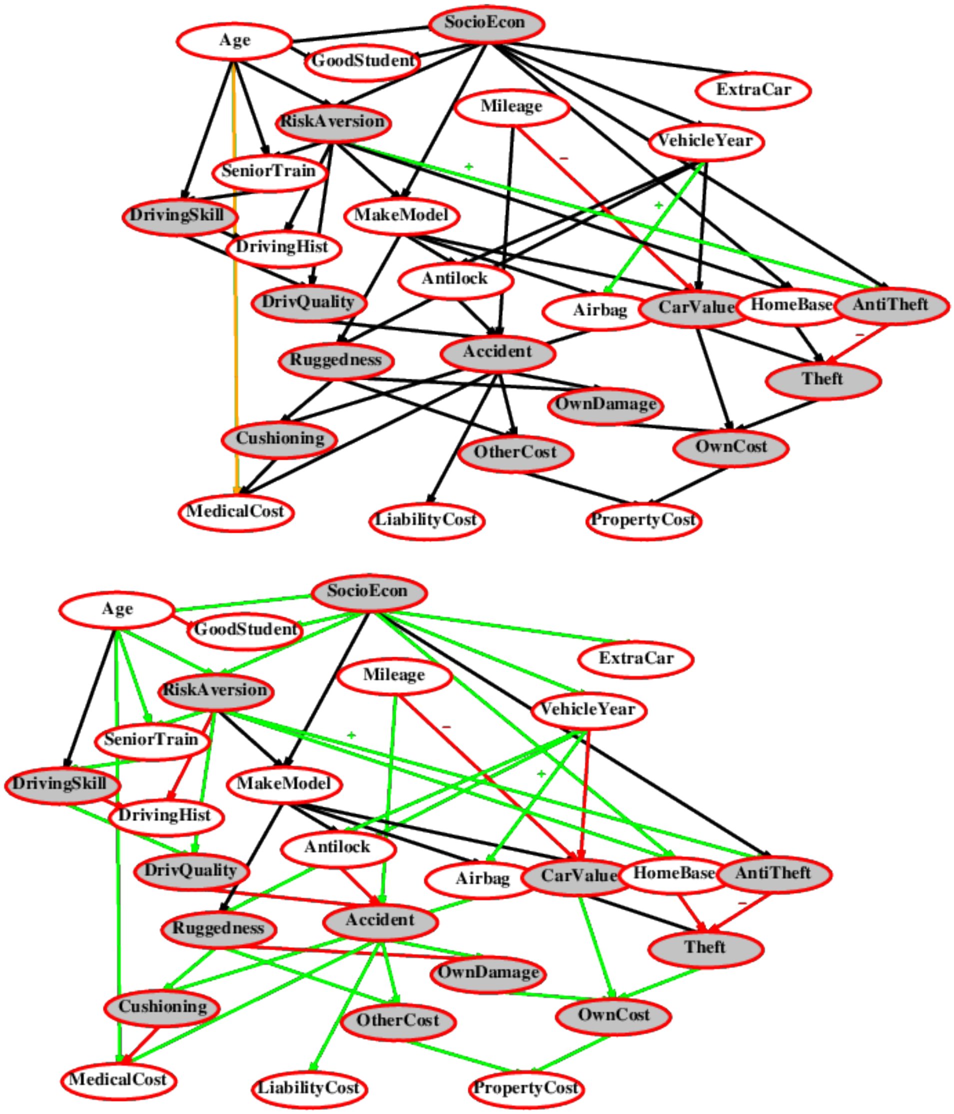
$$\int_{\infty}^{-\infty} p_1(x)U(x)dx \geq \int_{\infty}^{-\infty} p_2(x)U(x)dx$$

- ▷ Multi-attribute case: stochastic dominance on all attributes $\rightsquigarrow$ optimal.
- ▷ **Observation:** Stochastic dominance can often be determined without exact distributions using *qualitative* reasoning.
- ▷ **Example 6.4.7 (Construction cost increases with distance).** If airport location S_1 is closer to the city than $S_2 \rightsquigarrow S_1$ stochastically dominates S_2 on cost.q
- ▷ **Example 6.4.8.** Injury increases with collision speed.
- ▷ **Idea:** Annotate Bayesian networks with stochastic dominance information.
- ▷ **Definition 6.4.9.** $X \overset{+}{\to} Y$ (X positively influences Y) means that $\mathbf{P}(Y|X_1,\mathbf{z})$ stochastically dominates $\mathbf{P}(Y|X_2,\mathbf{z})$ for every value $\mathbf{z}$ of Y's other parents $\mathbf{Z}$ and all X_1 and X_2 with $X_1 \geq X_2$.

Label the arcs + or – for influence in a Bayesian Network




SocioEcon
Age
GoodStudent
ExtraCar
Mileage
RiskAversion
VehicleYear
SeniorTrain
DrivingSkill
MakeModel
DrivingHist
Antilock
DrivQuality
Airbag
CarValue
HomeBase
AntiTheft
Ruggedness
Accident
Theft
OwnDamage
Cushioning
OwnCost
OtherCost
MedicalCost
LiabilityCost
PropertyCost



We have seen how we can do inference with attribute-based utility functions, let us consider the computational implications. We observe that we have just replaced one evil – exponentially many states (in terms of the attributes) – by another – exponentially many parameters of the utility functions.

Wo we do what we always do in AI-2: we look for structure in the domain, do more theory to be able to turn such structures into computationally improved representations.

Preference Structure and Multi-Attribute Utility

▷ **Observation 6.4.10.** *With n attributes with d values each $\leadsto$ need d^n parameters for the utility function $U(X_1, \ldots, X_n)$.* *(worst case)*

▷ **Assumption:** Preferences of real agents have much more structure.

▷ **Approach:** Identify regularities and prove representation theorems based on these:

$$U(X_1, \ldots, X_n) = F(f_1(X_1), \ldots, f_n(f_n)X_n)$$

where F is simple, e.g. addition.

▷ Note the similarity to Bayesian networks that decompose the full joint probability distribution.

Preference structure: Deterministic

▷ **Recall:** In deterministic environments an agent has a value function.

▷ **Definition 6.4.11.** X_1 and X_2 preferentially independent of X_3 iff preference between $\langle x_1, x_2, z\rangle$ and $\langle x'_1, x'_2, z\rangle$ does not depend on z.

▷ **Example 6.4.12.** E.g., $\langle \mathrm{Noise}, \mathrm{Cost}, \mathrm{Safety}\rangle$: are preferentially independent
⟨20,000 suffer, 4.6 G$, 0.06 deaths/mpm⟩ vs.⟨70,000 suffer, 4.2 G$, 0.06 deaths/mpm⟩

▷ **Theorem 6.4.13 (Leontief, 1947).** *If every pair of attributes is preferentially independent of its complement, then every subset of attributes is preferentially independent of its complement: mutual preferential independence.*

▷ **Theorem 6.4.14 (Debreu, 1960).** *Mutual preferential independence implies that there is an additive value function: $V(S) = \sum_i V_i(X_i(S))$, where V_i is a value function referencing just one variable X_i.*

▷ Hence assess n single-attribute functions. (often a good approximation)

▷ **Example 6.4.15.** The value function for the airport decision might be

$$V(noise, cost, deaths) = -noise \cdot 10^4 - cost - deaths \cdot 10^{12}$$

Preference structure: Stochastic

▷ Need to consider preferences over lotteries and real utitlity functions (not just value functions)

▷ **Definition 6.4.16.** $\mathbf{X}$ is utility independent of $\mathbf{Y}$ iff preferences over lotteries in $\mathbf{X}$ do not depend on particular values in $\mathbf{Y}$.

▷ **Definition 6.4.17.** A set $\mathbf{X}$ is mutually utility independent (MUI), iff each subset is utility independent of its complement.

▷ **Example 6.4.18.** Arguably, the attributes of Example 6.4.2 are MUI.

▷ **Theorem 6.4.19.** *For MUI sets of attributes, there is a multiplicative utility function: [Kee74]*

▷ **Definition 6.4.20.** We "define" a multiplicative utility function by example: For three attributes we have:

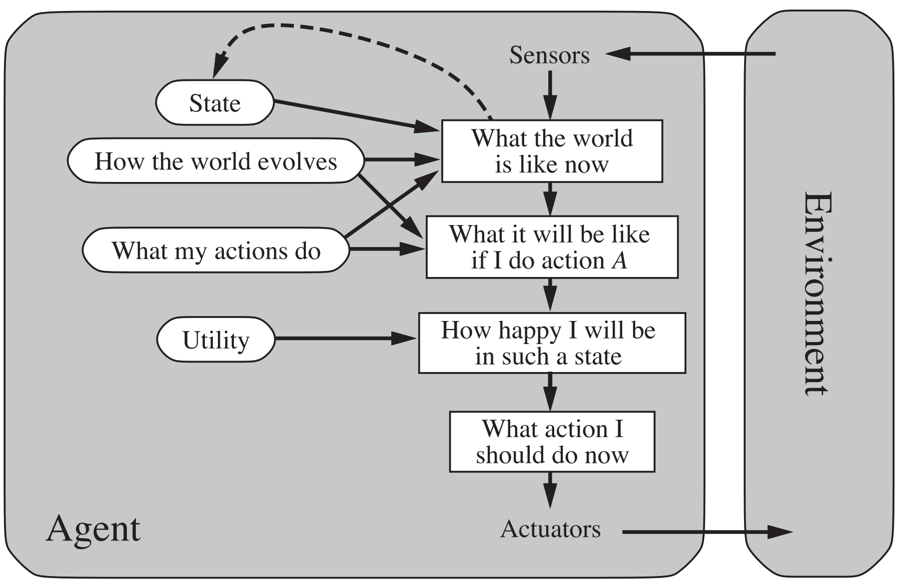
$$U = k_1U_1 + k_2U_2 + k_3U_3 + k_1k_2U_1U_2 + k_2k_3U_2U_3 + k_3k_1U_3U_1 + k_1k_2k_3U_1U_2U_3$$

▷ **System Support:** Routine procedures and software packages for generating preference tests to identify various canonical families of utility functions.

6.5 Decision Networks

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/30345`.
Now that we understand multi-attribute utilitysutility function, we can complete our design of a utility-based agent, which we now recapitulate as a refresher.

Utility-Based Agents (Recap)



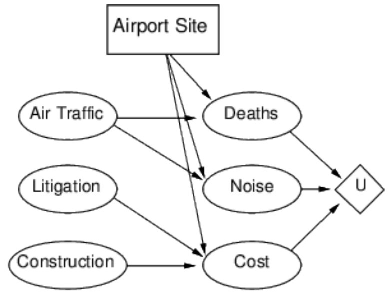
▷

As we already use Bayesian networks for the belief state of an utility-based agent, integrating utilities and possible actions into the network suggests itself naturally. This leads to the notion of a decision network.

Decision networks

▷ **Definition 6.5.1.** A decision network is a Bayesian network with added action nodes and utility nodes (also called value node) that enable decision making.

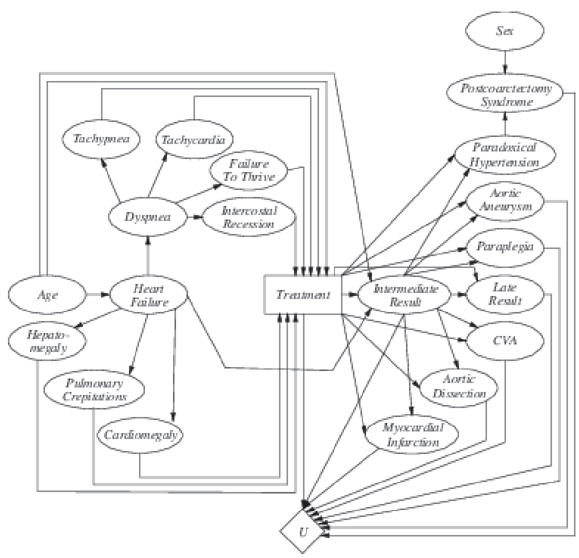
▷ **Example 6.5.2 (Choosing an Airport Site).**



▷ **Algorithm:** For each value of action node
compute expected value of utility node given action, evidence
Return MEU action (via argmax)

Decision Networks: Example

▷ **Example 6.5.3 (A Decision-Network for Aortic Coarctation).** from [Luc96]



Knowledge Eng. for Decision-Theoretic Expert Systems

▷ **Question:** How do you create a model like the one from Example 6.5.3?

▷ **Answer:** By a systematic process of the form: (after [Luc96])

1. **Create a causal model**: a graph with nodes for symptoms, disorders, treatments, outcomes, and their influences (edges).

2. **Simplify to a qualitative decision model**: remove random variables not involved in treatment decisions.
3. **Assign probabilities**: (⇝ Bayesian network)
   e.g. from patient databases, literature studies, or the expert's subjective assessments
4. **Assign utilities**. (e.g. in QALYs or micromorts)
5. **Verify and refine the model** wrt. a gold standard given by experts
   e.g. refine by "running the model backwards" and compare with the literature.
6. **Perform sensitivity analysis**: (important step in practice)
   - ▷ is the optimal treatment decision robust against small changes in the parameters? (if yes ⇝ great! if not, collect better data)

6.6 The Value of Information

Video Nuggets covering this section can be found at `https://fau.tv/clip/id/30346` and `https://fau.tv/clip/id/30347`.
So far we have tacitly been concentrating on actions that directly affect the environment. We will now come to a type of action we have hypothesized in the beginning of the course, but have completely ignored up to now: information gathering actions.

What if we do not have all information we need?

- ▷ **It is Well-Known:** One of the most important parts of decision making is knowing what questions to ask.
- ▷ **Example 6.6.1 (Medical Diagnosis).**
  - ▷ We do not expect a doctor to already know the results of the diagnostic tests when the patient comes in.
  - ▷ Tests are often expensive, and sometimes hazardous. (directly or by delaying treatment)
  - ▷ **Therefore**: Only test, if
    - ▷ knowing the results lead to a significantly better treatment plan,
    - ▷ information from test results is not drowned out by a-priori likelihood.
- ▷ **Definition 6.6.2.** Information value theory enables the agent to make decisions on information gathering rationally.
- ▷ **Intuition:** Simple form of sequential decision making. (action only impacts belief state).
- ▷ **Intuition:** With the new information, we can base the action choice to the *actual* information, rather than the average.

Value of Information by Example

- ▷ **Idea:** Compute value of acquiring each possible piece of evidence.

▷ **We will see:** This can be done directly from a decision network.

▷ **Example 6.6.3 (Buying Oil Drilling Rights).** There are n blocks of rights, exactly one has oil, worth k, in particular

- ▷ Prior probabilities $1/n$ each, mutually exclusive.
- ▷ Current price of each block is k/n.
- ▷ "Consultant" offers accurate survey of block 3. What's a fair price?

▷ **Solution:** Compute expected value of information $\widehat{=}$ expected value of best action given the information minus expected value of best action without information.

▷ **Example 6.6.4 (Oil Drilling Rights contd.).**

- ▷ Survey may say *oil in block 3 with probability* $1/n$ $\leadsto$ buy block 3 for k/n make profit of $(k - k/n)$.
- ▷ Survey may say *no oil in block 3 with probability* $(n-1)/n$ $\leadsto$ buy another block, make profit of $k/(n-1) - k/n$.
- ▷ Expected profit is $\frac{1}{n} \cdot \frac{(n-1)k}{n} + \frac{n-1}{n} \cdot \frac{k}{n(n-1)} = \frac{k}{n}$.
- ▷ So, we should pay up to k/n for the information. (as much as block 3 is worth)

General formula (VPI)

▷ Given current evidence E, possible actions $a \in A$ with outcomes in S_a, and current best action α

$$\mathrm{EU}(\alpha|E) = \max_{a \in A} \left(\sum_{s \in S_a} U(s) \cdot P(s|E, a)\right)$$

▷ Suppose we knew $F = f$ (new evidence), then we would choose α_f s.t.

$$\mathrm{EU}(\alpha_f|E, F = f) = \max_{a \in A} \left(\sum_{s \in S_a} U(s) \cdot P(s|E, a, F = f)\right)$$

here, F is a random variable with domain D whose value is *currently* unknown.

▷ **Idea:** So we must compute the expected gain over all possible values $f \in D$.

▷ **Definition 6.6.5.** Let F be a random variable with domain D, then the value of perfect information (VPI) on F given evidence E is defined as

$$\mathrm{VPI}_E(F) := \left(\sum_{f \in D} P(F = f|E) \cdot \mathrm{EU}(\alpha_f|E, F = f)\right) - \mathrm{EU}(\alpha|E)$$

where $\alpha_f = \underset{a \in A}{\mathrm{argmax}}\, \mathrm{EU}(a|E, F = f)$ and A the set of possible actions.

Properties of VPI

▷ **Observation 6.6.6 (VPI is Non-negative).**
$\mathrm{VPI}_E(F) \geq 0$ *for all* j *and* E *(in expectation, not post hoc)*

▷ **Observation 6.6.7 (VPI is Non-additive).**
$\mathrm{VPI}_E(F,G) \neq \mathrm{VPI}_E(F) + \mathrm{VPI}_E(G)$ *(consider, e.g., obtaining F twice)*

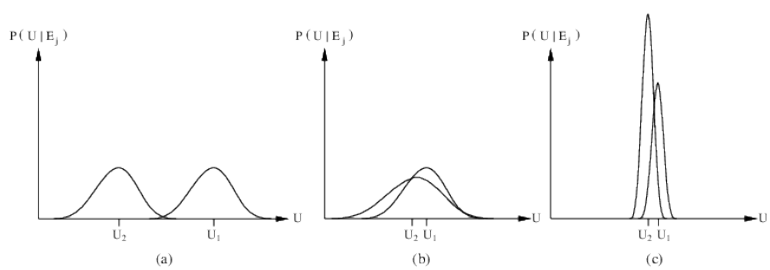
▷ **Observation 6.6.8 (VPI is Order-independent).**

$$\mathrm{VPI}_E(F,G) = \mathrm{VPI}_E(F) + \mathrm{VPI}_{E,F}(G) = \mathrm{VPI}_E(G) + \mathrm{VPI}_{E,G}(F)$$

▷ **Note:** When more than one piece of evidence can be gathered, maximizing VPI for each to select one is not always optimal
⇝ evidence-gathering becomes a sequential decision problem.

Qualitative behavior of VPI

▷ **Question:** Say we have three distributions for $P(U|E_j)$



Qualitatively: What is the value of information (VPI) in these three cases?

▷ **Answers:** reserved for the plenary sessions ⇝ be there!

We will now use information value theory to specialize our utility-based agent from above.

A simple Information-Gathering Agent

▷ **Definition 6.6.9.** A simple information gathering agent. (gathers info before acting)

function Information−Gathering−Agent (percept) **returns** an action
  persistent: D, a decision network
  integrate percept into D
  $j := \underset{k}{\mathrm{argmax}}\ \mathrm{VPI}_E(E_k)/Cost(E_k)$
  if $\mathrm{VPI}_E(E_j) > Cost(E_j)$ **return** Request(E_j)
  else return the best action from D

The next percept after Request(E_j) provides a value for E_j.

▷ **Problem:** The information gathering implemented here is myopic, i.e. calculating VPI as if only a single evidence variable will be acquired. (cf. greedy search)

▷ But it works relatively well in practice. (e.g. outperforms humans for selecting diagnostic tests)

Chapter 7

Making Complex Decisions

A Video Nugget covering the introduction to this chapter can be found at https://fau.tv/clip/id/30356.

We will now pick up the thread from chapter 6 but using temporal models instead of simply probabilistic ones. We will first look at a sequential decision theory in the special case, where the environment is stochastic, but fully observable (Markov decision processes) and then lift that to obtain POMDPs and present an agent design based on that.

Outline

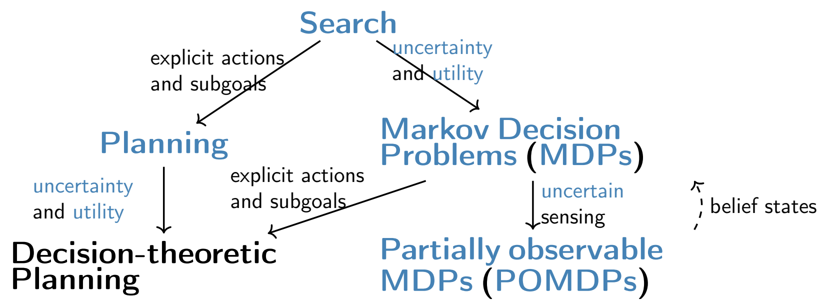
- ▷ Markov decision processes (MDPs) for sequential environments.
- ▷ Value/policy iteration for computing utilities in MDPs.
- ▷ Partially observable MDP (POMDPs).
- ▷ Decision theoretic agents for POMDPs.

7.1 Sequential Decision Problems

A Video Nugget covering this section can be found at https://fau.tv/clip/id/30357.

Sequential Decision Problems

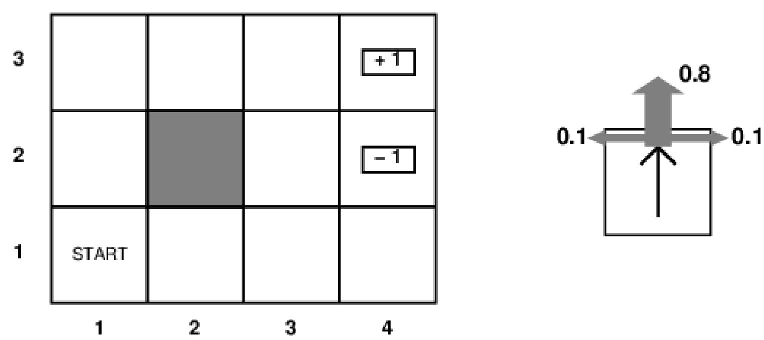
- ▷ **Definition 7.1.1.** In sequential decision problems, the agent's utility depends on a sequence of decisions (or their result states).
- ▷ **Definition 7.1.2.** Utility functions on action sequences are often expressed in terms of immediate rewards that are incurred upon reaching a (single) state.
- ▷ **Methods:** depend on the environment:
  - ▷ If it is fully observable ⇝ Markov decision process (MDPs)
  - ▷ else ⇝ partially observable MDP (POMDP).
- ▷ Sequential decision problems incorporate utilities, uncertainty, and sensing.
- ▷ **Preview:** Search problems and planning tasks are special cases.



We will fortify our intuition by an example. It is specifically chosen to be very simple, but to exhibit all the peculiarities of Markov decision problems, which we will generalize from this example.

Markov Decision Problem: Running Example

▷ **Example 7.1.3 (Running Example: The 4x3 World).** A (fully observable) 4×3 environment with non-deterministic actions:



- ▹ States $s \in \mathcal{S}$, actions $a \in \mathrm{Act}(s)$.
- ▹ Transition model: $P(s'|s, a) \,\widehat{=}$ probability that a in s leads to s'.
- ▹ reward function:

$$R(s) := \begin{cases} -0.04 & \text{if } \text{(small penalty) for nonterminal states} \\ \pm 1 & \text{if for terminal states} \end{cases}$$

Perhaps what is more interesting than the components of an MDP is that is *not* a component: a belief and/or sensor model. Recall that MDPs are for fully observable environments.

Markov Decision Process

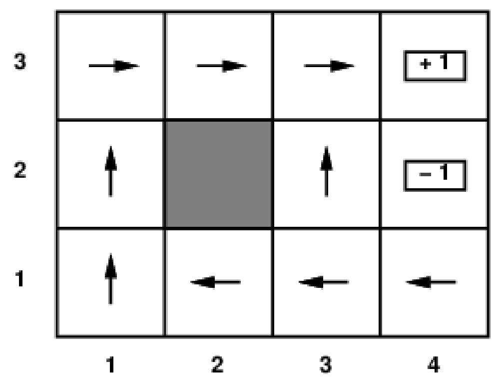
▷ **Motivation:** We are interested in sequential decision problems in a fully observable, stochastic environment with Markovian transition models and additive reward functions.

▷ **Definition 7.1.4.** A Markov decision process (MDP) $\langle \mathcal{S}, \mathrm{Act}, \mathcal{T}, s_0, R \rangle$ consists of

- a set of $\mathcal{S}$ of states (with initial state $s_0 \in \mathcal{S}$),
- sets $\mathrm{Act}(s)$ of actions for each state s.
- a transition model $\mathcal{T}(s, a) = s'$ with $P(s'|s, a)$, and
- a reward function $R\colon \mathcal{S} \to \mathbb{R}$ we call $R(s)$ a reward.

Solving MDPs

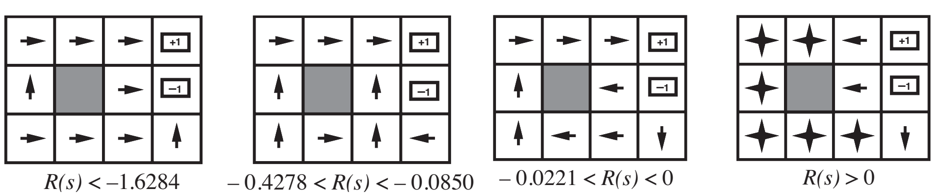
- **Recall:** In search problems, the aim is to find an optimal sequence of actions.
- In MDPs, the aim is to find an optimal policy $\pi(s)$ i.e., best action for every possible state s. (because can't predict where one will end up)
- **Definition 7.1.5.** In an MDP, a policy is a mapping from states to actions. An optimal policy maximizes (say) the expected sum of rewards. (MEU)
- **Example 7.1.6.** Optimal policy when state penalty $R(s)$ is 0.04:



Note: When you run against a wall, you stay in your square.

Risk and Reward

- **Example 7.1.7.** Optimal policy depends on the reward function $R(s)$.



- **Question:** Explain what you see in a qualitative manner!
- **Answer:** reserved for the plenary sessions ⇝ be there!

7.2 Utilities over Time

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/30358`.

In this section we address the problem that even if the transition models are stationary, the utilities may not be. In fact we generally have to take the utilities of state sequences into account in sequential decision problems. If we can derive a notion of the utility of a (single) state from that, we may be able to reuse the machinery we introduced above, so that is exactly what we will attempt.

Utility of state sequences

- ▷ **Recall:** We cannot observe/assess utility functions, only preferences ⇝ induce utility functions from rational preferences
- ▷ **Problem:** In MDPs we need to understand preferences between sequences of states.
- ▷ **Definition 7.2.1.** We call preferences on reward sequences stationary, iff

$$[r, r_0, r_1, r_2, \ldots] \succ [r, r'_0, r'_1, r'_2, \ldots] \Leftrightarrow [r_0, r_1, r_2, \ldots] \succ [r'_0, r'_1, r'_2, \ldots]$$

- ▷ **Theorem 7.2.2.** *For stationary preferences, there are only two ways to combine rewards over time.*
  - ▷ *additive rewards:* $U([s_0, s_1, s_2, \ldots]) = R(s_0) + R(s_1) + R(s_2) + \cdots$
  - ▷ *discounted rewards:* $U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$ *where* γ *is called discount factor.*

Utilities of State Sequences

- ▷ **Problem:** Infinite lifetimes ⇝ additive utilities become infinite.
- ▷ **Possible Solutions:**
  1. Finite horizon: terminate utility computation at a fixed time T

     $$U([s_0, \ldots, s_\infty]) = R(s_0) + \cdots + R(s_T)$$

     ⇝ nonstationary policy: $\pi(s)$ depends on time left.
  2. If there are absorbing states: for any policy π agent eventually "dies" with probability 1 ⇝ expected utility of every state is finite.
  3. Discounting: assuming $\gamma < 1$, $R(s) \leq R_{\max}$,

     $$U([s_0, \ldots, s_\infty]) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq \sum_{t=0}^{\infty} \gamma^t R_{\max} = R_{\max}/(1-\gamma)$$

     Smaller γ ⇝ shorter horizon.
- ▷ **Idea:** Maximize system gain $\widehat{=}$ average reward per time step.

▷ **Theorem 7.2.3.** *The optimal policy has constant gain after initial transient.*

▷ **Example 7.2.4.** Taxi driver's daily scheme cruising for passengers.

Utility of States

▷ **Intuition:** Utility of a state $\widehat{=}$ *expected (discounted) sum of rewards (until termination) assuming optimal actions*.

▷ **Definition 7.2.5.** Given a policy π, let s_t be the state the agent reaches at time t starting at state s_0. Then the expected utility obtained by executing π starting in s is given by

$$U^\pi(s) := E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t)\right]$$

we define $\pi^*_s := \underset{\pi}{\operatorname{argmax}}\, U^\pi(s)$.

▷ **Observation 7.2.6.** *π^*_s is independent of the state s.*

▷ *Proof sketch:* If π^*_a and π^*_b reach point c, then there is no reason to disagree – or with π^*_c

▷ **Definition 7.2.7.** We call $\pi^* := \pi^*_s$ for some s the optimal policy.

▷ ⚠ Observation 7.2.6 does not hold for finite horizon policies.

▷ **Definition 7.2.8.** The utility $U(s)$ of a state s is $U^{\pi^*}(s)$.

Utility of States (continued)

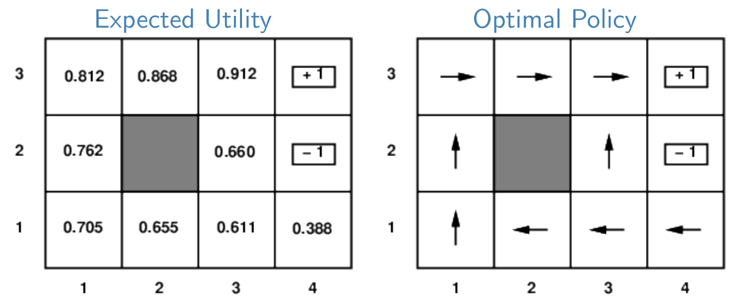
▷ **Remark:** $R(s) \widehat{=}$ "short-term reward", whereas $U \widehat{=}$ "long-term reward".

▷ Given the utilities of the states, choosing the best action is just MEU:

- ▷ maximize the expected utility of the immediate successor states

$$\pi^*(s) = \underset{a \in A(s)}{\operatorname{argmax}} \left(\sum_{s'} P(s'|s,a) \cdot U(s')\right)$$

▷ **Example 7.2.9 (Running Example Continued).**



▷ **Question:** Why do we go left in $(3,1)$ and not up? (follow the utility)

7.3 Value/Policy Iteration

A Video Nugget covering this section can be found at `https://fau.tv/clip/id/30359`.

Dynamic programming: the Bellman equation

▷ **Problem:** We have defined $U(s)$ via the optimal policy: $U(s) := U^{\pi^*}(s)$, but how to compute it without knowing π^*?

▷ **Observation:** A simple relationship among utilities of neighboring states:

expected sum of rewards = current reward + γ · exp. reward sum after best action

▷ **Theorem 7.3.1 (Bellman equation (1957)).**

$$U(s) = R(s) + \gamma \cdot \max_{a \in A(s)} \sum_{s'} U(s') \cdot P(s'|s,a)$$

We call this equation the Bellman equation

▷ **Example 7.3.2.** $U(1,1) = -0.04$

$+\ \gamma\ \max\{0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1),$	*up*
$0.9U(1,1) + 0.1U(1,2)$	*left*
$0.9U(1,1) + 0.1U(2,1)$	*down*
$0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)\}$	*right*

▷ **Problem:** One equation/state $\rightsquigarrow$ n nonlinear (max isn't) equations in n unknowns.
$\rightsquigarrow$ cannot use linear algebra techniques for solving them.

Value Iteration Algorithm

▷ **Idea:** We use a simple iteration scheme to find a fixpoint:

1. start with arbitrary utility values,

2. update to make them locally consistent with the Bellman equation,
3. everywhere locally consistent $\rightsquigarrow$ global optimality.

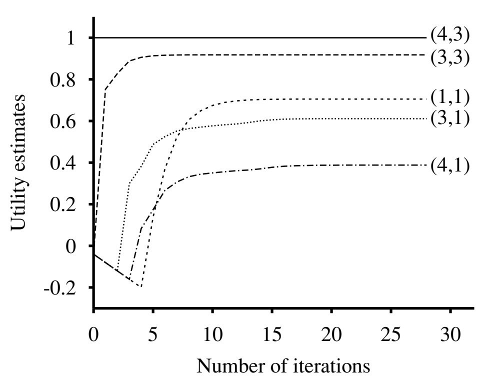
▷ **Definition 7.3.3.** The value iteration algorithm for utilitysutility function is given by

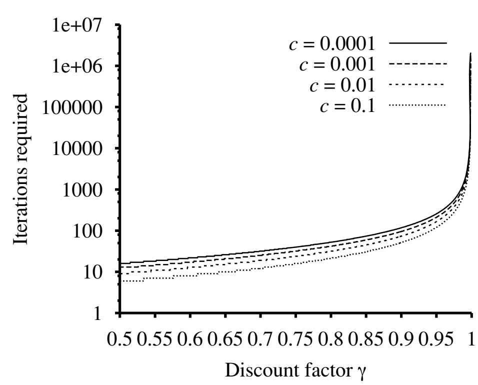
function VALUE−ITERATION (mdp,ϵ) **returns** a utility fn.
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s'|s,a)$,
      rewards $R(s)$, and discount γ
    ϵ, the maximum error allowed **in** the utility of any state
  local variables: U, U', vectors of utilities **for** states **in** S, initially zero
      δ, the maximum change **in** the utility of any state **in** an iteration
  repeat
    $U := U'$; $\delta := 0$
    for each state s **in** S **do**
      $U'[s] := R(s) + \gamma \cdot \max_{a \in A(s)} \left(\sum_{s'} U[s'] \cdot P(s'|s,a)\right)$
      if $|U'[s] - U[s]| > \delta$ **then** $\delta := |U'[s] - U[s]|$
    until $\delta < \epsilon(1-\gamma)/\gamma$
    return U

▷ **Remark:** Retrieve the optimal policy with $\pi[s] := \operatorname{argmax}_{a \in A(s)} \left(\sum_{s'} U[s'] \cdot P(s'|s,a)\right)$

Value Iteration Algorithm (Example)

▷ **Example 7.3.4 (Iteration on 4x3).**





Convergence

▷ **Definition 7.3.5.** The maximum norm $\|U\| = \max_s |U(s)|$, so $\|U - V\|$ = maximum difference between U and V.

▷ Let U^t and U^{t+1} be successive approximations to the true utility U.

▷ **Theorem 7.3.6.** *For any two approximations U^t and V^t*

$$\left\|U^{t+1} - V^{t+1}\right\| \leq \gamma \left\|U^t - V^t\right\|$$

I.e., any distinct approximations must get closer to each other so, in particular, any approximation must get closer to the true U and value iteration converges to a unique, stable, optimal solution.

▷ **Theorem 7.3.7.** *If* $\left\|U^{t+1} - U^t\right\| < \epsilon$, *then* $\left\|U^{t+1} - U\right\| < 2\epsilon\gamma/1 - \gamma$
I.e., once the change in U^t becomes small, we are almost done.

▷ **Remark:** MEU policy using U^t may be optimal long before convergence of values.

So we see that iteration with Bellman updates will always converge towards the utility of a state, even without knowing the optimal policy. That gives us a first way of dealing with sequential decision problems: we compute utility functions based on states and then use the standard MEU machinery. We have seen above that optimal policies and state utilities are essentially interchangeable: we can compute one from the other. This leads to another approach to computing state utilities: policy iteration, which we will discuss now.

Policy Iteration

▷ **Recap:** Value iteration computes utilities ⇝ optimal policy by MEU.

▷ This even works if the utility estimate is inaccurate. (⇜ policy loss small)

▷ **Idea:** Search for optimal policy and utility values simultaneously [How60]: Iterate

- ▷ policy evaluation: given policy π_i, calculate $U_i = U^{\pi_i}$, the utility of each state were π_i to be executed.
- ▷ policy improvement: calculate a new MEU policy π_{i+1} using 1 lookahead

Terminate if policy improvement yields no change in computed utilities.

▷ **Observation 7.3.8.** *Upon termination U_i is a fixpoint of Bellman update ⇝ Solution to Bellman equation ⇝ π_i is an optimal policy.*

▷ **Observation 7.3.9.** *Policy improvement improves policy and policy space is finite ⇝ termination.*

Policy Iteration Algorithm

▷ **Definition 7.3.10.** The policy iteration algorithm is given by the following pseudocode:

function POLICY−ITERATION(mdp) **returns** a policy
  inputs: mdp, and MDP with states S, actions $A(s)$, transition model $P(s'|s,a)$
  local variables: U a vector of utilities **for** states **in** S, initially zero
          π a policy indexed by state, initially random,
  repeat
    U := POLICY−EVALUATION(π,U,mdp)
    unchanged? := **true**
    foreach state s **in** X **do**
      if $\max_{a\in A(s)} \left(\sum_{s'} P(s'|s,a)\cdot U(s')\right) > \sum_{s'} P(s'|s,\pi[s'])\cdot U(s')$ **then do**
        $\pi[s] := \operatorname{argmax}_{b\in A(s)} \left(\sum_{s'} P(s'|s,b)\cdot U(s')\right)$
        unchanged? := **false**
  until unchanged?

return π

Policy Evaluation

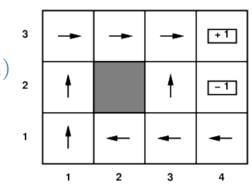
▷ **Problem:** How to implement the POLICY—EVALUATION algorithm?

▷ **Solution:** To compute utilities given a fixed π: For all s we have

$$U(s) = R(s) + \gamma(\sum_{s'} U(s') \cdot P(s'|s, \pi(s)))$$

▷ **Example 7.3.11 (Simplified Bellman Equations for π).**

$$\begin{aligned} U_i(1,1) &= -0.04 + 0.8U_i(1,2) + 0.1U_i(1,1) + 0.1U_i(2,1) \\ U_i(1,2) &= -0.04 + 0.8U_i(1,3) + 0.1U_i(1,2) \\ &\vdots \end{aligned}$$



▷ **Observation 7.3.12.** *n simultaneous linear equations in n unknowns, solve in $\mathcal{O}(n^3)$ with standard linear algebra methods.*

Modified Policy Iteration

▷ Policy iteration often converges in few iterations, but each is expensive.

▷ **Idea:** Use a few steps of value iteration (but with π fixed)
starting from thevalue function produced the last time
to produce an approximate value determination step.

▷ Often converges much faster than pure VI or PI.

▷ Leads to much more general algorithms where Bellman value updates and Howard policy updates can be performed locally in any order.

▷ **Remark:** Reinforcement learning algorithms operate by performing such updates based on the observed transitions made in an initially unknown environment.

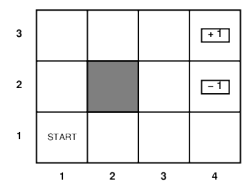
7.4 Partially Observable MDPs

We will now lift the last restriction we made in the decision problems for our agents: in the definition of Markov decision processes we assumed that the environment was fully observable. As we have seen Observation 7.2.6 this entails that the optimal policy only depends on the current state. **A Video Nugget** covering this section can be found at `https://fau.tv/clip/id/30360`.

Partial Observability

- **Definition 7.4.1.** A partially observable MDP (a POMDP for short) is a MDP together with an observation model O that has the sensor Markov property and is stationary: $O(s,e) = P(e|s)$.
- **Example 7.4.2 (Noisy 4x3 World).**

  Add a partial and/or noisy sensor.
  e.g. count number of adjacent walls ($1 \leq w \leq 2$)
  with 0.1 error (noise)
  If sensor reports 1, we are in $(3,?)$ (probably)



- **Problem:** Agent does not know which state it is in $\leadsto$ makes no sense to talk about policy $\pi(s)$!
- **Theorem 7.4.3 (Astrom 1965).** *The optimal policy in a POMDP is a function $\pi(b)$ where b is the belief state (probability distribution over states).*
- **Idea:** Convert a POMDP into an MDP in belief state space, where $\mathcal{T}(b,a,b')$ is the probability that the new belief state is b' given that the current belief state is b and the agent does a. I.e., essentially a filtering update step.

POMDP: Filtering at the Belief State Level

- **Recap:** Filtering updates the belief state for new evidence.
- For POMDPs, we also need to consider actions. (but the effect is the same)
- If b is the previous belief state and agent does action a and then perceives e, then the new belief state is

$$b'(s') = \alpha \cdot P(e|s') \cdot (\sum_s P(s'|s,a) \cdot b(s))$$

  We write $b' = \mathrm{FORWARD}(b,a,e)$ in analogy to recursive state estimation.
- **Fundamental Insight for POMDPs:** The optimal action only depends on the agent's current belief state. (good, it does not know the state!)
- **Consequence:** The optimal policy can be written as a function $\pi^*(b)$ from belief states to actions.
- **Definition 7.4.4.** The POMDP decision cycle is to iterate over
  1. Given the current belief state b, execute the action $a = \pi^*(b)$
  2. Receive percept e.
  3. Set the current belief state to $\mathrm{FORWARD}(b,a,e)$ and repeat.
- **Intuition:** POMDP decision cycle is search in belief state space.

Partial Observability contd.

- **Recap:** The POMDP decision cycle is search in belief state space.
- **Observation 7.4.5.** *Actions change the belief state, not just the (physical) state.*
- **Thus** POMDP solutions automatically include information gathering behavior.
- **Problem:** The belief state is continuous: If there are n states, b is an n-dimensional real-valued vector.
- **Example 7.4.6.** The belief state of the 4x3 world is a 11 dimensional continuous space. (11 states)
- **Theorem 7.4.7.** *Solving POMDPs is very hard!* (*actually,* **PSPACE** *hard*)
- **In particular,** none of the algorithms we have learned applies. (discreteness assumption)
- The real world is a POMDP (with initially unknown transition model T and sensor model O)

Reducing POMDPs to Belief-State MDPs

- **Idea:** Calculating the probability that an agent in belief state b reaches belief state b' after executing action a.
  - if we knew the action and the subsequent percept, then $b' = \mathrm{FORWARD}(b, a, e)$. (deterministic update to the belief state)
  - but we don't, so b' depends on e. (let's calculate $P(e|a,b)$)
- **Idea:** To compute $P(e|a,b)$ — the probability that e is perceived after executing a in belief state b — sum up over all actual states the agent might reach:

$$\begin{aligned} P(e|a,b) &= \sum_{s'} P(e|a,s',b) \cdot P(s'|a,b) \\ &= \sum_{s'} P(e|s') \cdot P(s'|a,b) \\ &= \sum_{s'} P(e|s') \cdot (\sum_{s} P(s'|s,a), b(s)) \end{aligned}$$

Write the probability of reaching b' from b, given action a, as $P(b'|b,a)$, then

$$\begin{aligned} P(b'|b,a) &= P(b'|a,b) = \sum_{e} P(b'|e,a,b) \cdot P(e|a,b) \\ &= \sum_{e} P(b'|e,a,b) \cdot (\sum_{s'} P(e|s') \cdot (\sum_{s} P(s'|s,a), b(s))) \end{aligned}$$

where $P(b'|e,a,b)$ is 1 if $b' = \mathrm{FORWARD}(b,a,e)$ and 0 otherwise.

▷ **Observation:** This equation defines a transition model for belief state space!

▷ **Idea:** We can also define a reward function for belief states:

$$\rho(b) := \sum_s b(s) \cdot R(s)$$

i.e., the expected reward for the actual states the agent might be in.

▷ Together, $P(b'|b,a)$ and $\rho(b)$ define an (observable) MDP on the space of belief states.

▷ **Theorem 7.4.8.** *An optimal policy $\pi^*(b)$ for this MDP, is also an optimal policy for the original POMDP.*

▷ **Upshot:** Solving a POMDP on a physical state space can be reduced to solving an MDP on the corresponding belief state space.

▷ **Remember:** The belief state is always observable to the agent, by definition.

Ideas towards Value-Iteration on POMDPs

▷ **Recap:** The value iteration algorithm from ?? computes one utility value per state.

▷ **Problem:** We have infinitely many belief states ⇝ be more creative!

▷ **Observation:** Consider an optimal policy π^*

- ▷ applied in a specific belief state b: π^* generates an action,
- ▷ for each subsequent percept, the belief state is updated and a new action is generated ...

For this specific b: $\pi^* \hat{=}$ a conditional plan!

▷ **Idea:** Think about conditional plans and how the expected utility of executing a fixed conditional plan varies with the initial belief state. (instead of optimal policies)

Expected Utilities of Conditional Plans on Belief States

▷ **Observation 1:** Let p be a conditional plan and $\alpha_p(s)$ the utility of executing p in state s.

- ▷ the expected utility of p in belief state b is $\sum_s b(s) \cdot \alpha_p(s) \hat{=} b{\cdot}\alpha_p$ as vectors.
- ▷ the expected utility of a fixed conditional plan varies linearly with b
- ▷ ⇝ it corresponds to a hyperplane in belief state space.

▷ **Observation 2:** Let π^* be the optimal policy. At any given belief state b,

- ▷ π^* will choose to execute the conditional plan with highest expected utility
- ▷ the expected utility of b under the π^* is the utility of that plan:

$$U(b) = U^{\pi^*}(b) = \max_b (b{\cdot}\alpha_p)$$

- If the optimal policy π^* chooses to execute p starting at b, then it is reasonable to expect that it might choose to execute p in belief states that are very close to b;
- if we bound the depth of the conditional plans, then there are only finitely many such plans
- the continuous space of belief states will generally be divided into regions, each corresponding to a particular conditional plan that is optimal in that region.

▷ **Observation 3 (conbined):** The utility function $U(b)$ on belief states, being the maximum of a collection of hyperplanes, is defined piecewise linear and convex.

A simple Illustrating Example

▷ **Example 7.4.9.** A world with states 0 and 1, where $R(0) = 0$ and $R(1) = 1$ and two actions:

- "Stay" stays put with probability 0.9
- "Go" switches to the other state with probability 0.9.
- The sensor reports the correct state with probability 0.6.

Obviously, the agent should "Stay" when it thinks it's in state 1 and "Go" when it thinks it's in state 0.

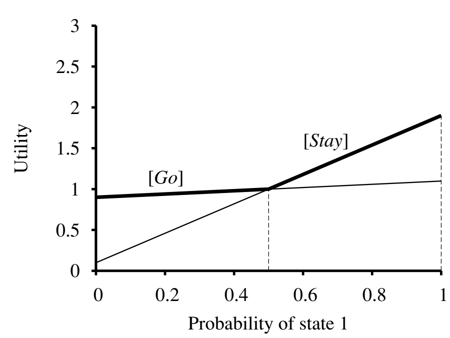
▷ The belief state has dimension 1. (the two probabilities sum up to 1)

▷ Consider the one-step plans $[Stay]$ and $[Go]$ and their (discounted) rewards:

$$\begin{aligned}
\alpha_{([Stay])}(0) &= R(0) + \gamma(0.9r(0) + 0.1r(1)) = 0.1 \\
\alpha_{([stay])}(1) &= r(1) + \gamma(0.9r(1) + 0.1r(0)) = 1.9 \\
\alpha_{([go])}(0) &= r(0) + \gamma(0.9r(1) + 0.1r(0)) = 0.9 \\
\alpha_{([go])}(1) &= r(1) + \gamma(0.9r(0) + 0.1r(1)) = 1.1
\end{aligned}$$

for now we will assume the discount factor $\gamma = 1$.

▷ Let us visualize the hyperplanes $b{\cdot}\alpha_{([Stay])}$ and $b{\cdot}\alpha_{([Go])}$.



- The maximum represents the represents the utility function for the finite-horizon problem that allows just one action
- in each "piece" the optimal action is the first action of the corresponding plan.

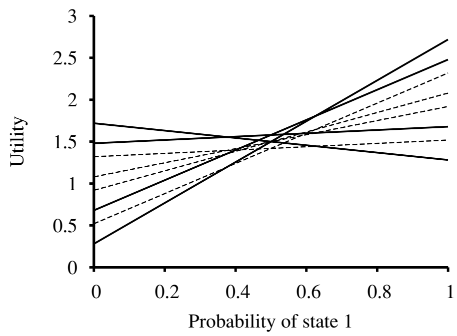
- ▷ Here the optimal one-step policy is to "Stay" when $b(1) > 0.5$ and "Go" otherwise.

▷ compute the utilities for conditional plans of depth 2 by considering

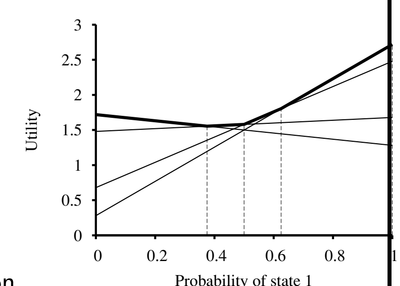
- ▷ each possible first action,
- ▷ each possible subsequent percept, and then
- ▷ each way of choosing a depth-1 plan to execute for each percept:

There are eight of depth 2:

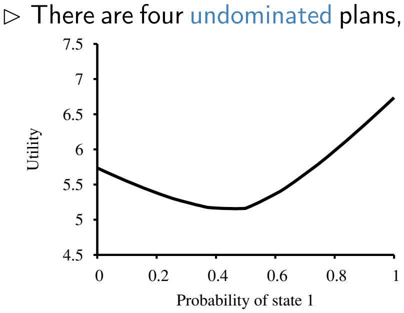
$$[Stay, \text{if } P = 0 \text{ then } Stay \text{ else } Stay \text{ fi}], [Stay, \text{if } P = 0 \text{ then } Stay \text{ else } Go \text{ fi}], \ldots$$



Four of them (dashed lines) are suboptimal for the whole belief space
We call them dominated (they can be ignored)



▷ There are four undominated plans, each optimal in their region



▷ **Idea:** Repeat for depth 3 and so on.

▷ **Theorem 7.4.10 (POMDP Plan Utility).** *Let p be a depth-d conditional plan whose initial action is a and whose depth-$d-1$-subplan for percept e is $p.e$, then*

$$\alpha_p(s) = R(s) + \gamma(\sum_{s'} P(s'|s,a)(\sum_{e} P(e|s') \cdot \alpha_{p.e}(s')))$$

▷ This recursion naturally gives us a value iteration algorithm,

A Value Iteration Algorithm for POMDPs

▷ **Definition 7.4.11.** The POMDP value iteration algorithm for POMDPs is given by

function POMDP–VALUE–ITERATION($pomdp$, ϵ) **returns** a utility **function**
inputs: $pomdp$, a POMDP with states S, actions $A(s)$, transition model $P(s'|s,a)$,
sensor model $P(e|s)$, rewards $R(s)$, discount γ
ϵ the maximum error allowed **in** the utility of any state
local variables: U , U', sets of plans p with associated utility vectors α_p
U' := a set containing just the empty plan $[]$, with $\alpha_{([])}(s) = R(s)$
repeat
$U := U'$
U' := the set of all plans consisting of an action and, **for** each possible next percept,
a plan **in** U with utility vectors computed via the POMDP Plan Utility Theorem
U' := REMOVE–DOMPLANS(U')
until MAX–DIFF(U,U') $< \epsilon(1-\gamma)/\gamma$
return U

Where REMOVE–DOMPLANS and MAX–DIFF are implemented as linear programs.

▷ **Observations:** The complexity depends primarily on the generated plans:

- ▷ Given $\#(A)$ actions and $\#(E)$ possible observations, there are are $\mathcal{O}(\#(A)^{\#(E)^{d-1}})$ distinct depth-d plans.
- ▷ Even for the example with $d = 8$, we have 2255 (144 undominated)
- ▷ The elimination of dominated plans is essential for reducing this doubly exponential growth (but they are already constructed)

▷ Hopelessly inefficient in practice – even the 3x4 POMDP is too hard!

7.5 Online Agents with POMDPs

In the last section we have seen that even though we can in principle compute utilities of states – and thus use the MEU principle – to make decisions in sequential decision problems, all methods based on the “lifting idea” are hopelessly inefficient.

This section describes a different, approximate method for solving POMDPs, one based on look-ahead search. **A Video Nugget** covering this section can be found at `https://fau.tv/clip/id/30361`.

DDN: Decision Networks for POMDPs

▷ **Idea:** Let's try to use the computationally efficient representations (dynamic Bayesian networks and decision networks) for POMDPs.

▷ **Definition 7.5.1.** A dynamic decision network (DDN) is a graph-based representation of a POMDP, where

- ▷ Transition and sensor model are represented as a DBN.
- ▷ Action nodes and utility nodes are added as in decision networks.

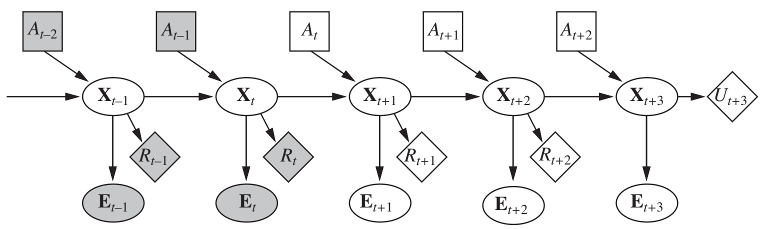
▷ In a DDN, a filtering algorithm is used to incorporate each new percept and action and to update the belief state representation.

▷ Decisions are made in DDN by projecting forward possible action sequences and choosing the best one.

▷ DDNs – like the DBNs they are based on – are factored representations
⇝ typically exponential complexity advantages!

Structure of DDNs for POMDPs

▷ **DDN for POMDPs:** The generic structure of a dymamic decision network at time t is



- ▷ POMDP state S_t becomes a set of random variables X_t
- ▷ there may be multiple evidence variables E_t
- ▷ Action at time t denoted by A_t. agent must choose a value for A_t.
- ▷ Transition model: $P(\mathrm{X}_{t+1}|\mathrm{X}_t, A_t)$; sensor model: $P(\mathrm{E}_t|\mathrm{X}_t)$.
- ▷ Reward functions R_t and utility U_t of state S_t.
- ▷ Variables with known values are gray, rewards for $t = 0, \ldots, t+2$, but utility for $t+3$($\widehat{=}$ discounted sum of rest)

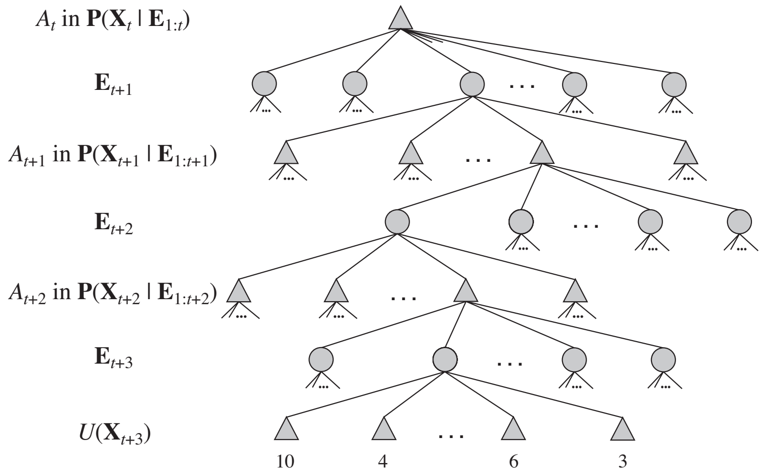
▷ **Problem:** How do we compute with that?

▷ **Answer:** All POMDP algorithms can be adapted to DDNs! (only need CPTs)

Lookahead: Searching over the Possible Action Sequences

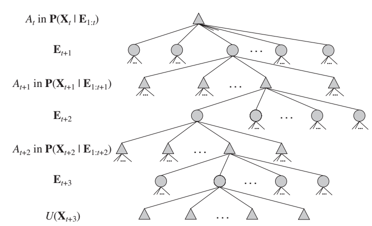
▷ **Idea:** Search over the tree of possible action sequences (like in game-play)

▷ Part of the lookahead solution of the DDN above (three steps lookahead)



- circle $\widehat{=}$ chance nodes (the environment decides)
- triangle $\widehat{=}$ belief state (each action decision is taken there)

Designing Online Agents for POMDPs



- **Note:** belief state update is deterministic irrespective of the action outcome $\leadsto$ no chance nodes for action outcomes
- Belief state at triangle computed by filtering with actions/percepts leading to it
  - for decision A_{t+i} will use percepts $\mathbf{E}_{t+1:t+i}$ (even if values at time t unknown)
  - thus a POMDP agent automatically takes into account the value of information and executes information gathering actions where appropriate.
- **Observation:** Time complexity for exhaustive search up to depth d is $\mathcal{O}(|A|^d \cdot |\mathbf{E}|^d)$($|A| \widehat{=}$ number of actions, $|\mathbf{E}| \widehat{=}$ number of percepts)
- **Upshot:** Much better than POMDP value iteration with $\mathcal{O}(\#(A)^{\#(E)^{d-1}})$.
- **Empirically:** For problems in which the discount factor γ is not too close to 1, a shallow search is often good enough to give near-optimal decisions.

Summary

- ▷ Decision theoretic agents for sequential environments
- ▷ Building on temporal, probabilistic models/inference (dynamic Bayesian networks)
- ▷ MDPs for fully observable case.
- ▷ Value/Policy Iteration for MDPs $\rightsquigarrow$ optimal policies.
- ▷ POMDPs for partially observable case.
- ▷ POMDPs$\widehat{=}$ MDP on belief state space.
- ▷ The world is a POMDP with (initially) unknown transition and sensor models.

Bibliography

[DF31] B. De Finetti. "Sul significato soggettivo della probabilita". In: *Fundamenta Mathematicae* 17 (1931), pp. 298–329.

[Glo] *Grundlagen der Logik in der Informatik*. Course notes at `https://www8.cs.fau.de/_media/ws16:gloin:skript.pdf`. URL: `https://www8.cs.fau.de/_media/ws16:gloin:skript.pdf` (visited on 10/13/2017).

[How60] R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[Kee74] R. L. Keeney. "Multiplicative utility functions". In: *Operations Research* 22 (1974), pp. 22–34.

[Koh08] Michael Kohlhase. "Using LaTeX as a Semantic Markup Format". In: *Mathematics in Computer Science* 2.2 (2008), pp. 279–304. URL: `https://kwarc.info/kohlhase/papers/mcs08-stex.pdf`.

[Luc96] Peter Lucas. "Knowledge Acquisition for Decision-theoretic Expert Systems". In: *AISB Quarterly* 94 (1996), pp. 23–33. URL: `https://www.researchgate.net/publication/2460438_Knowledge_Acquisition_for_Decision-theoretic_Expert_Systems`.

[Pra+94] Malcolm Pradhan et al. "Knowledge Engineering for Large Belief Networks". In: *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*. UAI'94. Seattle, WA: Morgan Kaufmann Publishers Inc., 1994, pp. 484–490. ISBN: 1-55860-332-8. URL: `http://dl.acm.org/citation.cfm?id=2074394.2074456`.

[RN03] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 2nd ed. Pearso n Education, 2003. ISBN: 0137903952.

[RN09] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd. Prentice Hall Press, 2009. ISBN: 0136042597, 9780136042594.

[RN95] Stuart J. Russell and Peter Norvig. *Artificial Intelligence — A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 1995.

[sTeX] *sTeX: A semantic Extension of TeX/LaTeX*. URL: `https://github.com/sLaTeX/sTeX` (visited on 05/11/2020).

[WHI] *Human intelligence — Wikipedia The Free Encyclopedia*. URL: `https://en.wikipedia.org/w/index.php?title=Human_intelligence` (visited on 04/09/2018).