Instructor's Solution Manual

Artificial Intelligence A Modern Approach

Fourth Edition

Stuart J. Russell and Peter Norvig

with contributions from

Nalin Chhibber, Ernest Davis, Nicholas J. Hay, Jared Moore, Alex Rudnick, Mehran Sahami, Xiaocheng Mesut Yang, and Albert Yu

This solution manual is intended for the instructor of a class. Students should use the online site for exercises at `aimacode.github.io/aima-exercises`. That site is open for anyone to use. It offers solutions for some but not all of the exercises; an instructor can check there to see which ones have solutions. The exercises are online rather than in the textbook itself because (a) the textbook is long enough as is, and (b) we wanted to be able to update the exercises frequently.

EXERCISES 1

INTRODUCTION

Note that for many of the questions in this chapter, we give references where answers can be found rather than writing them out—the full answers would be far too long.

1.1 What Is AI?

Exercise 1.1.#DEFA

Define in your own words: (a) intelligence, (b) artificial intelligence, (c) agent, (d) rationality, (e) logical reasoning.

a. Dictionary definitions of **intelligence** talk about "the capacity to acquire and apply knowledge" or "the faculty of thought and reason" or "the ability to comprehend and profit from experience." These are all reasonable answers, but if we want something quantifiable we would use something like "the ability to act successfully across a wide range of objectives in complex environments."

b. We define **artificial intelligence** as the study and construction of agent programs that perform well in a given class of environments, for a given agent architecture; they *do the right thing*. An important part of that is dealing with the uncertainty of what the current state is, what the outcome of possible actions might be, and what is it that we really desire.

c. We define an **agent** as an entity that takes action in response to percepts from an environment.

d. We define **rationality** as the property of a system which does the "right thing" given what it knows. See Section 2.2 for a more complete discussion. The basic concept is *perfect* rationality; Section **??** describes the impossibility of achieving perfect rationality and proposes an alternative definition.

e. We define **logical reasoning** as the a process of deriving new sentences from old, such that the new sentences are necessarily true if the old ones are true. (Notice that does not refer to any specific syntax or formal language, but it does require a well-defined notion of truth.)

Exercise 1.1.#TURI

Read Turing's original paper on AI (Turing, 1950). In the paper, he discusses several objections to his proposed enterprise and his test for intelligence. Which objections still carry

weight? Are his refutations valid? Can you think of new objections arising from developments since he wrote the paper? In the paper, he predicts that, by the year 2000, a computer will have a 30% chance of passing a five-minute Turing Test with an unskilled interrogator. What chance do you think a computer would have today? In another 25 years?

See the solution for exercise 26.1 for some discussion of potential objections.

The probability of fooling an interrogator depends on just how unskilled the interrogator is. A few entrants in the Loebner prize competitions have fooled judges, although if you look at the transcripts, it looks like the judges were having fun rather than taking their job seriously. There certainly have been examples of a chatbot or other online agent fooling humans. For example, see the description of the Julia chatbot at `www.lazytd.com/lti/julia/`. We'd say the chance today is something like 10%, with the variation depending more on the skill of the interrogator rather than the program. In 25 years, we expect that the entertainment industry (movies, video games, commercials) will have made sufficient investments in artificial actors to create very credible impersonators.

Note that governments and international organizations are seriously considering rules that require AI systems to be identified as such. In California, it is already illegal for machines to impersonate humans in certain circumstances.

Exercise 1.1.#REFL

Are reflex actions (such as flinching from a hot stove) rational? Are they intelligent?

Yes, they are rational, because slower, deliberative actions would tend to result in more damage to the hand. If "intelligent" means "applying knowledge" or "using thought and reasoning" then it does not require intelligence to make a reflex action.

Exercise 1.1.#SYAI

To what extent are the following computer systems instances of artificial intelligence:

- Supermarket bar code scanners.
- Web search engines.
- Voice-activated telephone menus.
- Spelling and grammar correction features in word processing programs.
- Internet routing algorithms that respond dynamically to the state of the network.

- Although bar code scanning is in a sense computer vision, these are not AI systems. The problem of reading a bar code is an extremely limited and artificial form of visual interpretation, and it has been carefully designed to be as simple as possible, given the hardware.
- In many respects. The problem of determining the relevance of a web page to a query is a problem in natural language understanding, and the techniques are related to those

we will discuss in Chapters 23 and 24. Search engines also use clustering techniques analogous to those we discuss in Chapter 20. Likewise, other functionalities provided by a search engines use intelligent techniques; for instance, the spelling corrector uses a form of data mining based on observing users' corrections of their own spelling errors. On the other hand, the problem of indexing billions of web pages in a way that allows retrieval in seconds is a problem in database design, not in artificial intelligence.

- To a limited extent. Such menus tends to use vocabularies which are very limited – e.g. the digits, "Yes", and "No" — and within the designers' control, which greatly simplifies the problem. On the other hand, the programs must deal with an uncontrolled space of all kinds of voices and accents. Modern digital assistants like Siri and the Google Assistant make more use of artificial intelligence techniques, but still have a limited repetoire.
- Slightly at most. The spelling correction feature here is done by string comparison to a fixed dictionary. The grammar correction is more sophisticated as it need to use a set of rather complex rules reflecting the structure of natural language, but still this is a very limited and fixed task.

  The spelling correctors in search engines would be considered much more nearly instances of AI than the Word spelling corrector are, first, because the task is much more dynamic – search engine spelling correctors deal very effectively with proper names, which are detected dynamically from user queries – and, second, because of the technique used – data mining from user queries vs. string matching.
- This is borderline. There is something to be said for viewing these as intelligent agents working in cyberspace. The task is sophisticated, the information available is partial, the techniques are heuristic (not guaranteed optimal), and the state of the world is dynamic. All of these are characteristic of intelligent activities. On the other hand, the task is very far from those normally carried out in human cognition. In recent years there have been suggestions to base more core algorithmic work on machine learning.

Exercise 1.1.#COGN

Many of the computational models of cognitive activities that have been proposed involve quite complex mathematical operations, such as convolving an image with a Gaussian or finding a minimum of the entropy function. Most humans (and certainly all animals) never learn this kind of mathematics at all, almost no one learns it before college, and almost no one can compute the convolution of a function with a Gaussian in their head. What sense does it make to say that the "vision system" is doing this kind of mathematics, whereas the actual person has no idea how to do it?

Presumably the brain has evolved so as to carry out this operations on visual images, but the mechanism is only accessible for one particular purpose in this particular cognitive task of image processing. Until about two centuries ago there was no advantage in people (or animals) being able to compute the convolution of a Gaussian for any other purpose.

The really interesting question here is what we mean by saying that the "actual person" can do something. The person can see, but he cannot compute the convolution of a Gaussian;

but computing that convolution is *part* of seeing. This is beyond the scope of this solution manual.

Exercise 1.1.#EVOR

Why would evolution tend to result in systems that act rationally? What goals are such systems designed to achieve?

The notion of acting rationally *presupposes* an objective, whether explicit or implicit. We understand evolution as a process that operates in the physical world, where there are no inherent objectives. So the question is really asking whether evolution tends to produce systems whose behavior can be interpreted consistently as rational according to some objective.

It is tempting to say that evolution tends to produce organisms that act rationally in the pursuit of reproduction. This is not completely wrong but the true picture is much more complex because of the question of what "system" refers to—it could be organisms (humans, rats, bacteria), superorganisms (ant and termite colonies, human tribes, corals), and even individual genes and groups of genes within the genome. Selection and mutation processes operate at all these levels. By definition, the systems that exist are those whose progenitors have reproduced successfully. If we consider an ant colony, for example, there are many individual organisms (e.g., worker ants) that do not reproduce at all, so it is not completely accurate to say that evolution produces organisms whose objective is to reproduce.

Exercise 1.1.#AISC

Is AI a science, or is it engineering? Or neither or both? Explain.

This question is intended to be about the essential nature of the AI problem and what is required to solve it, but could also be interpreted as a sociological question about the current practice of AI research.

A *science* is a field of study that leads to the acquisition of empirical knowledge by the scientific method, which involves falsifiable hypotheses about what is. A pure *engineering* field can be thought of as taking a fixed base of empirical knowledge and using it to solve problems of interest to society. Of course, engineers do bits of science—e.g., they measure the properties of building materials—and scientists do bits of engineering to create new devices and so on.

The "human" side of AI is clearly an empirical science—called cognitive science these days—because it involves psychological experiments designed out to find out how human cognition actually works. What about the the "rational" side? If we view it as studying the abstract relationship among an arbitrary task environment, a computing device, and the program for that computing device that yields the best performance in the task environment, then the rational side of AI is really mathematics and engineering; it does not require any empirical knowledge about the *actual* world—and the *actual* task environment—that we inhabit; that a given program will do well in a given environment is a *theorem*. (The same is true of pure decision theory.) In practice, however, we are interested in task environments that do approximate the actual world, so even the rational side of AI involves finding out what the actual

world is like. For example, in studying rational agents that communicate, we are interested in task environments that contain humans, so we have to find out what human language is like. In studying perception, we tend to focus on sensors such as cameras that extract useful information from the actual world. (In a world without light, cameras wouldn't be much use.) Moreover, to design vision algorithms that are good at extracting information from camera images, we need to understand the actual world that generates those images. Obtaining the required understanding of scene characteristics, object types, surface markings, and so on is a quite different kind of science from ordinary physics, chemistry, biology, and so on, but it is still science.

In summary, AI is definitely engineering but it would not be especially useful to us if it were not also an empirical science concerned with those aspects of the real world that affect the design of intelligent systems for that world.

Exercise 1.1.#INTA

"Surely computers cannot be intelligent—they can do only what their programmers tell them." Is the latter statement true, and does it imply the former?

This depends on your definition of "intelligent" and "tell." In one sense computers only do what the programmers command them to do, but in another sense what the programmers consciously tells the computer to do often has very little to do with what the computer actually does. Anyone who has written a program with an ornery bug knows this, as does anyone who has written a successful machine learning program. So in one sense Samuel "told" the computer "learn to play checkers better than I do, and then play that way," but in another sense he told the computer "follow this learning algorithm" and it learned to play. So we're left in the situation where you may or may not consider learning to play checkers to be a sign of intelligence (or you may think that learning to play in the right way requires intelligence, but not in this way), and you may think the intelligence resides in the programmer or in the computer.

Exercise 1.1.#INTB

"Surely animals cannot be intelligent—they can do only what their genes tell them." Is the latter statement true, and does it imply the former?

The point of this exercise is to notice the parallel with the previous one. Whatever you decided about whether computers could be intelligent in 1.11, you are committed to making the same conclusion about animals (including humans), *unless* your reasons for deciding whether something is intelligent take into account the mechanism (programming via genes versus programming via a human programmer). Note that Searle makes this appeal to mechanism in his Chinese Room argument (see Chapter 27).

Exercise 1.1.#INTC

"Surely animals, humans, and computers cannot be intelligent—they can do only what their constituent atoms are told to do by the laws of physics." Is the latter statement true, and does it imply the former?

Again, your definition of "intelligent" drives your answer to this question.

1.2 The Foundations of Artificial Intelligence

Exercise 1.2.#NTRC

There are well-known classes of problems that are intractably difficult for computers, and other classes that are provably undecidable. Does this mean that AI is impossible?

No. It means that AI systems should avoid trying to solve intractable problems. Usually, this means they can only approximate optimal behavior. Notice that humans don't solve NP-complete problems either. Sometimes they are good at solving specific instances with a lot of structure, perhaps with the aid of background knowledge. AI systems should attempt to do the same.

Exercise 1.2.#SLUG

The neural structure of the sea slug *Aplysia* has been widely studied (first by Nobel Laureate Eric Kandel) because it has only about 20,000 neurons, most of them large and easily manipulated. Assuming that the cycle time for an *Aplysia* neuron is roughly the same as for a human neuron, how does the computational power, in terms of memory updates per second, compare with the personal computer described in Figure 1.2?

Depending on what you want to count, the computer has a thousand to a million times more storage, and a thousand times more operations per second.

Exercise 1.2.#INTR

How could introspection—reporting on one's inner thoughts—be inaccurate? Could I be wrong about what I'm thinking? Discuss.

Just as you are unaware of all the steps that go into making your heart beat, you are also unaware of most of what happens in your thoughts. You do have a conscious awareness of some of your thought processes, but the majority remains opaque to your consciousness. The field of psychoanalysis is based on the idea that one needs trained professional help to analyze one's own thoughts. Neuroscience has also shown that we are unaware of much of the activity in our brains.

1.3 The History of Artificial Intelligence

Exercise 1.3.#IQEV

Suppose we extend Evans's ANALOGY program (Evans, 1968) so that it can score 200 on a standard IQ test. Would we then have a program more intelligent than a human? Explain.

No. IQ test scores correlate well with certain other measures, such as success in college, ability to make good decisions in complex, real-world situations, ability to learn new skills and subjects quickly, and so on, but *only* if they're measuring fairly normal humans. The IQ test doesn't measure everything. A program that is specialized only for IQ tests (and specialized further only for the analogy part) would very likely perform poorly on other measures of intelligence. Consider the following analogy: if a human runs the 100m in 10 seconds, we might describe him or her as *very athletic* and expect competent performance in other areas such as walking, jumping, hurdling, and perhaps throwing balls; but we would not describe a Boeing 747 as *very athletic* because it can cover 100m in 0.4 seconds, nor would we expect it to be good at hurdling and throwing balls.

Even for humans, IQ tests are controversial because of their theoretical presuppositions about innate ability (distinct from training effects) and the generalizability of results. See *The Mismeasure of Man* (Stephen Jay Gould, 1981) or *Multiple Intelligences: the Theory in Practice* (Howard Gardner, 1993) for more on IQ tests, what they measure, and what other aspects there are to "intelligence."

Exercise 1.3.#PRMO

Some authors have claimed that perception and motor skills are the most important part of intelligence, and that "higher level" capacities are necessarily parasitic—simple add-ons to these underlying facilities. Certainly, most of evolution and a large part of the brain have been devoted to perception and motor skills, whereas AI has found tasks such as game playing and logical inference to be easier, in many ways, than perceiving and acting in the real world. Do you think that AI's traditional focus on higher-level cognitive abilities is misplaced?

Certainly perception and motor skills are important, and it is a good thing that the fields of vision and robotics exist (whether or not you want to consider them part of "core" AI). But given a percept, an agent still has the task of "deciding" (either by deliberation or by reaction) which action to take. This is just as true in the real world as in artificial micro-worlds such as chess-playing. So computing the appropriate action will remain a crucial part of AI, regardless of the perceptual and motor system to which the agent program is "attached." On the other hand, it is true that a concentration on micro-worlds has led AI away from the really interesting environments such as those encountered by self-driving cars.

Exercise 1.3.#WINT

Several "AI winters," or rapid collapses in levels of economic and academic activity (and media interest) associated with AI, have occurred. Describe the causes of each collapse and of the boom in interest that preceded it.

In addition to the information in the chapter, ? (?, ?), ? (?), and ? (?) provide ample starting material for the aspiring historian of AI. One can identify at least three AI winters (although the phrase was not applied to the first one, because the original phrase **nuclear winter** did not emerge until the early 1980s).

a. As noted in the chapter, research funding dried up in the early 1970s in both the US and UK. The ostensible reason was failure to make progress on the rather lavish promises of the 1960s, particularly in the areas of neural networks and machine translation. In 1970, the US Congress curtailed most AI funding from ARPA, and in 1973 the Lighthill report in the UK ended funding for all but a few researchers. Lighthill referred particularly to the difficulties of overcoming the combinatorial explosion.

b. In the late 1980s, the expert systems boom ended, due largely to the difficulty and expense of building and maintaining expert systems for complex applications, the lack of a valid uncertainty calculus in these systems, and the lack of interoperability between AI software and hardware and existing data and computation infrastructure in industry.

c. In the early 2000s, the end of the dot-com boom also ended an upsurge of interest in the use of AI systems in the burgeoning online ecosystem. AI systems had been used for such tasks as information extraction from web pages to support shopping engines and price comparisons; various kinds of search engines; planning algorithms for achieving complex goals requiring several steps and combining information from multiple web pages; and converting human-readable web pages into machine-readable database tuples to allow global information aggregation, as in citation databases constructed from online pdf files.

It is also interesting to explore the extent to which the winters were due to over-optimistic and exaggerated claims by AI researchers or to over-enthusiasm and over-interpretation of the significance of early results by funders and investors.

Exercise 1.3.#DLAI

The resurgence of interest in AI in the 2010s is often attributed to deep learning. Explain what deep learning is, how it relates to AI as a whole, and where the core technical ideas actually originated.

Deep learning is covered in Section 1.3.8, where it is defined as "machine learning using multiple layers of simple, adjustable computing elements." Thus, it is a particular branch of machine learning, which is itself a subfield of AI. Since many AI systems do not use learning at all, and there are many effective machine learning techniques that are unrelated to deep learning, the view (often expressed in popular articles on AI) that deep learning has "replaced" AI is wrong for multiple reasons.

Some of the key technical ideas are the following (see also Chapter 21):

- *Networks of simple, adjustable computing elements*: ? (?), drawing on ? (?, ?).
- *Backpropagation*, i.e., a localized way of computing gradients of functions expressed by networks of computing elements, based on the chain rule of calculus: ? (?, ?). For neural network learning specifically, ? (?) preceded by more than a decade the much better-known work on ? (?).
- *Convolutional networks* with many copies of small subnetworks, all sharing the same patterns of weights: This is usually attributed to work in the 1990s on handwritten digit recognition by ? (?) at AT&T Bell Labs. ? (?) acknowledge the influence of the **neocognitron** model (?), which in turn was inspired by the neuroscience work of ? (?, ?).
- *Stochastic gradient descent* to facilitate learning in deep networks: as described in the historical notes section of Chapter 19, ? (?) explored stochastic approzimations to gradient methods, including convergence properties; the first application to neural networks was by ? (?) and independently by ? (?) in their ADALINE networks.

1.4 The State of the Art

Exercise 1.4.#SOTA

Examine the AI literature to discover whether the following tasks can currently be solved by computers:

a. Playing a decent game of table tennis (Ping-Pong).
b. Driving in the center of Cairo, Egypt.
c. Driving in Victorville, California.
d. Buying a week's worth of groceries at the market.
e. Buying a week's worth of groceries on the Web.
f. Playing a decent game of bridge at a competitive level.
g. Discovering and proving new mathematical theorems.
h. Writing an intentionally funny story.
i. Giving competent legal advice in a specialized area of law.
j. Translating spoken English into spoken Swedish in real time.
k. Performing a complex surgical operation.

For the currently infeasible tasks, try to find out what the difficulties are and predict when, if ever, they will be overcome.

a. (ping-pong) A reasonable level of proficiency was achieved by Andersson's robot (Andersson, 1988).
b. (driving in Cairo) No. Although there has been a lot of progress in automated driving, they operate in restricted domains: on the highway, in gated communities, or in well-mapped cities with limited traffic problems. Driving in downtown Cairo is too unpredictable for any of these to work.

c. (driving in Victorville, California) Yes, to some extent, as demonstrated in DARPA's Urban Challenge. Some of the vehicles managed to negotiate streets, intersections, well-behaved traffic, and well-behaved pedestrians in good visual conditions.

d. (shopping at the market) No. No robot can currently put together the tasks of moving in a crowded environment, using vision to identify a wide variety of objects, and grasping the objects (including squishable vegetables) without damaging them. The component pieces are nearly able to handle the individual tasks, but it would take a major integration effort to put it all together.

e. (shopping on the web) Yes. Software robots are capable of handling such tasks, particularly if the design of the web grocery shopping site does not change radically over time.

f. (bridge) Yes. Programs such as GIB now play at a solid level.

g. (theorem proving) Yes. For example, the proof of Robbins algebra.

h. (funny story) No. While some computer-generated prose and poetry is hysterically funny, this is invariably unintentional, except in the case of programs that echo back prose that they have memorized.

i. (legal advice) Yes, in some cases. AI has a long history of research into applications of automated legal reasoning. Two outstanding examples are the Prolog-based expert systems used in the UK to guide members of the public in dealing with the intricacies of the social security and nationality laws. The social security system is said to have saved the UK government approximately $150 million in its first year of operation. However, extension into more complex areas such as contract law awaits a satisfactory encoding of the vast web of common-sense knowledge pertaining to commercial transactions and agreement and business practices.

j. (translation) Yes. Although translation is not perfect, it is serviceable, and is used by travellers every day.

k. (surgery) Yes. Robots are increasingly being used for surgery, although always under the command of a doctor. Robotic skills demonstrated at superhuman levels include drilling holes in bone to insert artificial joints, suturing, and knot-tying. They are not yet capable of planning and carrying out a complex operation autonomously from start to finish.

Exercise 1.4.#CONT

Various subfields of AI have held contests by defining a standard task and inviting researchers to do their best. Examples include the ImageNet competition for computer vision, the DARPA Grand Challenge for robotic cars, the International Planning Competition, the Robocup robotic soccer league, the TREC information retrieval event, and contests in machine translation, speech recognition, and other fields. Investigate one of these contests, and describe the progress made over the years. To what degree have the contests advanced the state of the art in AI? Do what degree do they hurt the field by drawing energy away from new ideas?

The progress made in these contests is a matter of fact, but the impact of that progress is a matter of opinion. Some examples:

- **DARPA Grand Challenge for Robotic Cars**: In 2004 the Grand Challenge was a 240 km race through the Mojave Desert. It clearly stressed the state of the art of autonomous driving, and in fact no competitor finished the race. The best team, CMU, completed only 12 of the 240 km. In 2005 the race featured a 212km course with fewer curves and wider roads than the 2004 race. Five teams finished, with Stanford finishing first, edging out two CMU entries. This was hailed as a great achievement for robotics and for the Challenge format. In 2007 the Urban Challenge put cars in a city setting, where they had to obey traffic laws and avoid other cars. This time CMU edged out Stanford. The competition appears to have been a good testing ground to put theory into practice, something that the failures of 2004 showed was needed. But it is important that the competition was done at just the right time, when there was theoretical work to consolidate, as demonstrated by the earlier work by Dickmanns (whose VaMP car drove autonomously for 158km in 1995) and by Pomerleau (whose Navlab car drove 5000km across the USA, also in 1995, with the steering controlled autonomously for 98% of the trip, although the brakes and accelerator were controlled by a human driver).
- **International Planning Competition**: In 1998, five planners competed: Blackbox, HSP, IPP, SGP, and STAN. The result page (`ftp://ftp.cs.yale.edu/pub/mcdermott/aipscomp-results.html`) stated "all of these planners performed very well, compared to the state of the art a few years ago." Most plans found were 30 or 40 steps, with some over 100 steps. In 2008, the competition had expanded quite a bit: there were more tracks (satisficing vs. optimizing; sequential vs. temporal; static vs. learning). There were about 25 planners, including submissions from the 1998 groups (or their descendants) and new groups. Solutions found were much longer than in 1998. In sum, the field has progressed quite a bit in participation, in breadth, and in power of the planners. In the 1990s it was possible to publish a Planning paper that discussed only a theoretical approach; now it is necessary to show quantitative evidence of the efficacy of an approach. The field is stronger and more mature now, and it seems that the planning competition deserves some of the credit. However, some researchers feel that too much emphasis is placed on the particular classes of problems that appear in the competitions, and not enough on real-world applications.
- **Robocup Robotic Soccer**: This competition has proved extremely popular, attracting 300–500 teams from 40–50 countries since the mid-2000s (up from 38 teams from 11 countries in 1997). The "standard platform" league, originally based on the Sony AIBO four-legged robot, switched to the humanoid Aldebaran Nao robot in 2009. There are also small and mid-size leagues in which teams are free to design their own robots provided they satisfy certain physical constraints. The competition has served to increase interest and participation in robotics and has led to some advances in both robotics and AI (particularly related to learning in team games). Victory on competitions ofen depends less on AI than on specific tricks for improving ball-handling and shooting. The long-term goal is to defeat a human World-Cup-winning team by 2050, although the precise details of ensuring safety of human participants remain to be worked out.
- **TREC Information Retrieval Conference**: This is one of the oldest competitions,

started in 1992. The competitions have served to bring together a community of researchers, have led to a large literature of publications, and have seen progress in participation and in quality of results over the years. In the early years, TREC served its purpose as a place to do evaluations of retrieval algorithms on text collections that were large for the time. However, starting around 2000 TREC became less relevant as the advent of the World Wide Web created a corpus that was available to anyone and was much larger than anything TREC had created, and the development of commercial search engines surpassed academic research.

- **ImageNet**: The ImageNet database is a hand-labelled collection of over 14 million images in over 20,000 categories (?). The ImageNet competition (more properly, the ImageNet Large Scale Visual Recognition Challenge or ILSVRC) is based on a subset of 1,000 categories, 90 of which are breeds of dog. The availability of such large training sets is often asserted to be a contributing factor in the emergence of deep learning. In the 2012 competition, the decisive win by AlexNet (?) set off a frenzy of research that has reduced error rates on ILSVRC well below the human level of about 5%. Research by (?), however, suggests that for many types of deep learning the progress may be illusory: in many cases, the object is "recognized" through the color and spatial distribution of background pixels such as the grass on which a dog is sitting.

Overall, we see that whatever you measure is bound to increase over time. For most of these competitions, the measurement was a useful one, and the state of the art has progressed. In the case of ICAPS, some planning researchers worry that too much attention has been lavished on the competition itself. In some cases, progress has left the competition behind, as in TREC, where the resources available to commercial search engines outpaced those available to academic researchers. In this case the TREC competition was useful—it helped train many of the people who ended up in commercial search engines—and in no way drew energy away from new ideas. In computer vision, as in planning, improved performance on competition benchmarks has become almost essential for a new idea to be published, which many argue is a serious obstacle to research and may lead to an entire field become stuck in a dead-end approach.

Exercise 1.4.#DROB

Investigate the state of the art for domestic robots: what can be done (with what assumptions and restrictions on the environment) and what problems remain unsolved? Where is research most needed?

Creating a fully general domestic robot that will work "out of the box" in any household remains a distant goal. At the time of writing (mid-2021), robot demonstrations for a number of specific tasks have been given. Some examples:

- Folding laundry (?), `https://youtu.be/gy5g33S0Gzo`. The robot assumes that a pile contains only rectangular pieces of cloth and requires a uniform green background. Subsequent work allowed a PR2 to complete almost the entire laundry cycle with a fairly wide assortment of laundry (?), `https://www.nsf.gov/discoveries/disc_summ.jsp?c` However, a demonstration exhibit at the Victoria & Albert Museum in London showed

the fragility of the solution: the commercial Baxter robot broke down too often for the exhibit to function for more than a fraction of the time.

- Loading and unloading a dishwasher: the subject of academic research-lab demos since the early 2000s (CMU's HERB robot, for example), this one is creeping closer to commercial realization, with companies such as Samsung showing carefully edited demos of robotic products that are not yet available. As with other such demos, the technology is not ready to take on an arbitrary kitchen and dishwasher. Also, many robots are far from waterproof.
- Cooking: ? (?) describe an integrated system intended to download recipes from the Internet, turn them into executable plans, and carry out those plans in a real kitchen. Only quite preliminary results were achieved: `https://www.csail.mit.edu/node/6480`. As with washing up, cooking is very messy and current general-purpose robots need to be carefully wrapped in plastic protective gear. There are also commercial "robot cooks" that claim to cook dishes in the home. Typically these are special-purpose systems that require specially prepared and sized ingredients and are generally not robust to failure (see, e.g., `https://www.youtube.com/watch?v=M8r0gmQXm1Y` and `https://www.youtube.com/watch?v=GyEHRXA_aA4`). Again, we are far from having a robot that go into anyone's kitchen with a bag of ingedients from the supermarket and make dinner.

Successful teleoperation demos (where a human controls a robot to carry out tasks such as opening a beer bottle) suggest that the problems do not lie primarily with robotic hardware (except for the need to design robots that are immune to water, grease, and solid ingredients. Perception has advanced considerably, but it is still difficult for a robot to analyze a pile of ingredients in a bowl and judge where the surface is and how well mixed the ingredients are. Spatial reasoning involving continuous, irregular shapes and liquid, semiliquid, and powdered ingredients or complex objects made from cloth, leather, etc., is quite weak. Also lacking is the ability to plan and manage domestic activities on a continuous basis, with constant interruptions from humans who rearrange the world.

Exercise 1.4.#JRNL

The vast majority of academic publications and media articles report on successes of AI. Examine recent articles describing failures of AI. (? (?) and ? (?) are good examples, but feel free to find your own.) How serious are the problems identified? Are they examples of overclaiming of or fundamental problems with the technical approach?

? (?) define *overinterpretation* as occurring when a machine learning algorithm finds predictive regularities in parts of the input data that are semantically irrelevant. Those regularities reflect particular properties of the training and test data, such as when all photos of Norfolk Terriers are taken with the dog sitting in various poses a particular crimson carpet. In that case, simply recognizing the presence of a few pixels of crimson suffices to identify the breed of dog. They show that this problem arises quite frequently with modern deep learning systems applied to standard data sets such as ImageNet and CIFAR-10. For some categories, such as "airplane" in CIFAR-10, a single pixel suffices to classify the object. In

almost all cases, the trained network labels images with high confidence using only a very small subset of fairly randomly distributed pixels that, to a human, is unintelligible. This seems to reveal a fundamental problem both with the technology (the networks appear to have too much capacity and no notion of what they are looking for) and the standard train/test methodology (which fails to detect the non-robustness of the learned classifiers). Put another way, the network does not know that "Norfolk Terrier" is a breed of dog rather than a carpet color ("Crimson Carpet"), so the task solved by the machine learning algorithm is not the one solved by a human.

? (?) describe a similar issue: for any given training set, there are typically vast numbers of learned network configurations that give equally good (or even perfect) performance on held-out data. They call this *underspecification* and note that it leads to poor performance after deployment in a wide range of real-world applications including computer vision, medical imaging, natural language processing, clinical risk prediction based on electronic health records, and medical genomics. They conclude that machine learning pipelines must be constrained by strong semantic priors if learning is to be effective and reliable (?, see also).

For a paper highly critical of the use of machine learning to diagnose COVID-19 from chest X-rays, see ? (?). The authors selected 62 of the most promising and carefully documented studies from a total of 2,212 published in 2020, but found that "none of the models identified are of potential clinical use due to methodological flaws and/or underlying biases." The problems identified were primarily failings on the part of the studies' authors, who may not have been aware of the stringent criteria for real-world deployment of high-stakes diagnostic tools. However, even a methodologically perfect study using deep learning might well have failed for reasons given in the preceding two paragraphs.

1.5 Risks and Benefits of AI

Exercise 1.5.#PRIN

Find and analyze at least three sets of proposed principles for the governance of AI. What do the sets of principles have in common? How do they differ? How implementable are these principles?

There are several hundred published sets of principles: some of the best-known are the OECD, Beijing, and Asilomar principles. ? (?, ?), and ? (?) provide useful analytical surveys and summaries.

Exercise 1.5.#EURG

Study the 2021 EU "Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act)" (or its final version, if available). Which provisions appear to have the most direct and tangible impact on what kinds of AI systems can be deployed?

If students have never seen legislation before, this will be an eye-opening exercise. Most

of the document stipulates practices in testing, documentation, etc. Only a few of the clauses have tangible impact—most obviously, those concerning facial recognition, impersonation of humans, and deepfakes. In these three areas, the rules are quite strict compared to those in force in most other parts of the world.

Exercise 1.5.#LATM

Summarize the pros and cons of allowing the development, deployment, and use of lethal autonomous weapons.

A good place to start is the Congressional Research Service report "International Discussions Concerning Lethal Autonomous Weapon Systems," dated October 15, 2020. Unfortunately the report leaves out the principal argument against lethal autonomous weapons: they will become cheap, scalable weapons of mass destruction that will proliferate easily and are likely to be used against civilian populations.

Exercise 1.5.#BIAS

Many researchers have pointed to the possibility that machine learning algorithms will produce classifiers that display racial, gender, or other forms of bias. How does this bias arise? Is it possible to constrain machine learning algorithms to produce rigorously fair predictions?

Bias in ML occurs largely because of two things: first, the bias that exists in data generated by a biased society, and second, specifying an incorrect objective for machine learning algorithms——namely, maximizing agreement with the training data. This is not the real objective: in fact, we care about fairness too.

It's possible to define various measures of fairness and to design algorithms that respect them. See ? (?) for a survey. Unfortunately, it is not possible to simultaneously satisfy all "reasonable" definitions of fairness, and there may be some sacrifice in accuracy.

Exercise 1.5.#SIFI

Examine at least three science-fiction movies in which AI systems threaten to (or actually do) "take control." In the movie, does the takeover stem from "spooky emergent consciousness" or from a poorly defined objective? If the latter, what was it?

Here are three examples:

a. *Colossus—The Forbin Project*: A US defense computer tasked with ensuring peace and security encounters a Soviet compueter with the same goal. They exchange information and decide that the goal is best achieved by jointly controlling all nuclear weapons and using threats of destruction to force humans to drop all aggressive war plans. Here, the problem is clearly a poorly defined objective.

b. *Ex Machina*: Ava, a very human-like android, brilliantly engineers her escape from a remote facility by outwitting her human captors and allies. Although it is never stated

explicitly in the film, her objective seems to be in places populated by large numbers of people; she is portrayed as having this objective as a result of a process of consciousness emerging from the sum total of human interactions recorded in a global social media platform.

c. *Transcendence*: Will Caster, a Berkeley AI professor, is gunned down by anti-AI terrorists. Before he dies, his brain is uploaded to a quantum supercomputer. The machine becomes conscious and begins to quickly outrun the human race, threatening to take over the world. Here, the risk comes from the possibility that the machine's conscious goals differ from those of its human progenitor.

EXERCISES 2

INTELLIGENT AGENTS

2.1 Agents and Environments

Exercise 2.1.#DFAG

Define in your own words the following terms: agent, environment, sensor, actuator, percept, agent function, agent program.

The following are just some of the many possible definitions that can be written:

Mobile agent

- *Agent*: an entity that perceives and acts; or, one that *can be viewed* as perceiving and acting. Essentially any object qualifies; the key point is the way the object implements an agent function. (Note: some authors restrict the term to *programs* that operate *on behalf of* a human, or to programs that can cause some or all of their code to run on other machines on a network, as in **mobile agents**.)
- *Environment*: the world in which the agent is situated, on which the agent's actuators produce effects and from which the agent's sensors receive percepts.
- *Sensor*: part of the agent whose state is affected by the environment and whose state can enter into the agent's computations directly as a value.
- *Actuator*: part of the agent whose state can be set by the agent's computations and affects the environment.
- *Percept*: the observed value of the sensor state (usually, all the sensors taken together) at a given time.
- *Agent function*: given an agent viewed as a fixed physical object, the agent function that specifies the agent's action in response to every possible percept sequence.
- *Agent program*: that program which, combined with a machine architecture, produces an agent function. In our simple designs, the program takes a new percept on each invocation and returns an action. Note that the agent program is generally not simply "implementing" the agent function exactly, because the program may take more than one time step to return an action.

Exercise 2.1.#AGEX

For each of the following agents, specify the sensors, actuators, and environment: microwave oven, chess program, autonomous supply delivery plane.

The following are just some of the many possible definitions that can be written:

- *Microwave oven*: Here it is important to view the oven as the agent, not the human who is using it; this means the buttons are sensors, not actuators! It might seem that there are no decisions to make but the internal controller logic can be quite complex, particularly if there are additional sensors. For safety, no power should be supplied if the door is open!
  - *Sensors*: Button state, door-open sensor. Some microwaves have humidity sensors, temperature sensors, or acoustic sensors (for popcorn).
  - *Actuators*: Turn on/turn off/adjust microwave power; sound completion alarm; indicate illegal button state.
  - *Environment*: The contents of the oven and the state of the door (open/closed).
- *Chess program*:
  - *Sensors*: A typical program accepts keyboard, mouse, or touchscreen input indicating the opponent's intended move, resignation, hint request, etc.
  - *Actuators*: Display a legal move, or the board resulting from the move.
  - *Environment*: At a minimum, the environment includes a representation of the board state. There are some subtle issues related to whether this is identical to the program's own internal representation or the displayed board that is accessible to the opponent. A more sophisticated program might include the human opponent as part of the environment, and try to build a model of that human's playing style and ability in order to improve its chances of winning or make the game more entertaining/instructive.
- *Autonomous supply delivery plane*: The answers here are quite similar to those for the autonomous taxi in Section 2.3. Details can be found in numerous online sources describing current autonomous delivery systems.
  - *Sensors*: GPS, air speed (e.g., Pitot tube), altimeter (actually an air pressure sensor, so reported altitude depends on meteorological conditions), sensors reporting state of each actuator, 3-axis accelerometer, possibly a camera.
  - *Actuators*: power to propeller, ailerons, elevators, rudder; drop payload.
  - *Environment*: geographical area of flight, atmospheric conditions.

Exercise 2.1.#AGFN

This exercise explores the differences between agent functions and agent programs.

a. Can there be more than one agent program that implements a given agent function? Give an example, or show why one is not possible.

b. Are there agent functions that cannot be implemented by any agent program?

c. Given a fixed machine architecture, does each agent program implement exactly one agent function?

d. Given an architecture with n bits of storage, how many different possible agent programs are there?

e. Suppose we keep the agent program fixed but speed up the machine by a factor of two. Does that change the agent function?

Although these questions are very simple, they hint at some very fundamental issues. Our answers are for the simple agent designs for *static* environments where nothing happens while the agent is deliberating; the issues get even more interesting for dynamic environments.

a. Yes; take any agent program and insert null statements that do not affect the output.

b. Yes; the agent function might specify that the agent print $true$ when the percept is a Turing machine program that halts, and $false$ otherwise. (Note: in dynamic environments, for machines of less than infinite speed, the rational agent function may not be implementable; e.g., the agent function that always plays a winning move, if any, in a game of chess.)

c. Yes; the agent's behavior is fixed by the architecture and program.

d. There are 2^n agent programs, although many of these will not run at all. (Note: Any given program can devote at most n bits to storage, so its internal state can distinguish among only 2^n past histories. Because the agent function specifies actions based on percept histories, there will be many agent functions that cannot be implemented because of lack of memory in the machine.)

e. It depends on the program and the environment. If the environment is dynamic, speeding up the machine may mean choosing different (perhaps better) actions and/or acting sooner. If the environment is static and the program pays no attention to the passage of elapsed time, the agent function is unchanged.

2.2 Good Behavior: The Concept of Rationality

Exercise 2.2.#PRMT

Suppose that the performance measure is concerned with just the first T time steps of the environment and ignores everything thereafter. Show that a rational agent's action may depend not just on the state of the environment but also on the time step it has reached.

This question tests the student's understanding of environments, rational actions, and performance measures. Any sequential environment in which rewards may take time to arrive will work, because then we can arrange for the reward to be "over the horizon." Suppose that in any state there are two action choices, a and b, and consider two cases: the agent is in state s at time T or at time $T-1$. In state s, action a reaches state s' with reward 0, while action b reaches state s again with reward 1; in s' either action gains reward 10. At time $T-1$, it's rational to do a in s, with expected total reward 10 before time is up; but at time T, it's rational to do b with total expected reward 1 because the reward of 10 cannot be obtained before time is up.

Students may also provide common-sense examples from real life: investments whose payoff occurs after the end of life, exams where it doesn't make sense to start the high-value question with too little time left to get the answer, and so on.

The environment state can include a clock, of course; this doesn't change the gist of the answer—now the action will depend on the clock as well as on the non-clock part of the state—but it does mean that the agent can never be in the same state twice.

Exercise 2.2.#VACR

Let us examine the rationality of various vacuum-cleaner agent functions.

a. Show that the simple vacuum-cleaner agent function described in Figure 2.3 is indeed rational under the assumptions listed on page 40.

b. Describe a rational agent function for the case in which each movement costs one point. Does the corresponding agent program require internal state?

c. Discuss possible agent designs for the cases in which clean squares can become dirty and the geography of the environment is unknown. Does it make sense for the agent to learn from its experience in these cases? If so, what should it learn? If not, why not?

Notice that for our simple environmental assumptions we need not worry about quantitative uncertainty.

a. It suffices to show that for all possible actual environments (i.e., all dirt distributions and initial locations), this agent cleans the squares at least as fast as any other agent. This is trivially true when there is no dirt. When there is dirt in the initial location and none in the other location, the world is clean after one step; no agent can do better. When there is no dirt in the initial location but dirt in the other, the world is clean after two steps; no agent can do better. When there is dirt in both locations, the world is clean after three steps; no agent can do better. (Note: in general, the condition stated in the first sentence of this answer is much stricter than necessary for an agent to be rational.)

b. The agent in (a) keeps moving backwards and forwards even after the world is clean. It is better to do $NoOp$ once the world is clean (the chapter says this). Now, since the agent's percept doesn't say whether the other square is clean, it would seem that the agent must have some memory to say whether the other square has already been cleaned. To make this argument rigorous is more difficult—for example, could the agent arrange things so that it would only be in a clean left square when the right square was already clean? As a general strategy, an agent *can* use the environment itself as a form of **external memory**—a common technique for humans who use things like appointment calendars and knots in handkerchiefs. In this particular case, however, that is not possible. Consider the reflex actions for $[A, Clean]$ and $[B, Clean]$. If either of these is $NoOp$, then the agent will fail in the case where that is the initial percept but the other square is dirty; hence, neither can be $NoOp$ and therefore the simple reflex agent is doomed to keep moving. In general, the problem with reflex agents is that they have to do the same thing in situations that look the same, even when the situations are actually quite different. In the vacuum world this is a big liability, because every interior square (except home) looks either like a square with dirt or a square without dirt.

External memory

c. If we consider asymptotically long lifetimes, then it is clear that learning a map (in some form) confers an advantage because it means that the agent can avoid bumping into walls. It can also learn where dirt is most likely to accumulate and can devise an optimal inspection strategy. The precise details of the exploration method needed to construct a complete map appear in Chapter 4; methods for deriving an optimal

inspection/cleanup strategy are in Chapter 22.

Exercise 2.2.#KPER

Describe three different task environments in which the performance measure is easy to specify completely and correctly, and three in which it is not.

Obviously there are many possible answers here. For the "easy" case, it makes sense to consider artificially defined task environments: electronic calculators (answers correct to the required number of significant digits, elapsed time), chess programs (win/draw/loss subject to time constraints), spider solitaire (number of suits completed, number of moves, elapsed time). But even in these environments, one must be careful, because such performance measures ignore *all other considerations*. For example, a general-purpose intelligent agent that has the ability to display messages or access the internet might improve its chess performance by acquiring additional hardware. Marvin Minsky pointed out that a machine designed to calculate as many digits of π as possible could take over the world to do so.

For the "hard" case, almost any task involving humans will do: tutor, automated taxi, digital personal assistant, military strategist, and so on. Although Section 2.3 lists the elements of the taxi's performance measure (reaching destination, fuel consumption, wear and tear, trip time, traffic laws, politness to other drivers, safety, passenger comfort, profit), the appropriate tradeoffs among these are far from clear. For example, if the taxi is 200 yards from the destination but stuck in a traffic jam that will take 20 minutes to clear, should it suggest to the passengers that they walk the remaining 200 yards? And how does this decision depend on the weather and general safety and legality of this premature dropoff location?

In addition to difficult tradeoffs, there will typically be attributes that are omitted from the performance measure that do in fact matter and can be affected by the agent. Obviously it doesn't make sense to ask students to list the attributes that are omitted, because they could simply include them! Instead, one might ask which attributes are "obvious" and which are "non-obvious," i.e., likely to be omitted on a first or second pass.

Exercise 2.2.#EVSA

Write an essay on the relationship between evolution and one or more of autonomy, intelligence, rationality, and learning.

Exercise 2.2.#AGTF

For each of the following assertions, say whether it is true or false and support your answer with examples or counterexamples where appropriate.

a. An agent that senses only partial information about the state cannot be perfectly rational.

b. There exist task environments in which no pure reflex agent can behave rationally.

c. There exists a task environment in which every agent is rational.

d. The input to an agent program is the same as the input to the agent function.

e. Every agent function is implementable by some program/machine combination.

f. Suppose an agent selects its action uniformly at random from the set of possible actions. There exists a deterministic task environment in which this agent is rational.

g. It is possible for a given agent to be perfectly rational in two distinct task environments.

h. Every agent is rational in an unobservable environment.

i. A perfectly rational poker-playing agent never loses.

j. For a simple reflex agent in a partially observable environment, a randomized policy can outperform any deterministic policy.

k. There is a model-based reflex agent that can remember all of its percepts.

l. Suppose agent A1 is rational and agent A2 is irrational. There exists a task environment where A2's actual score will be greater than A1's actual score.

a. *An agent that senses only partial information about the state cannot be perfectly rational.*
False. Perfect rationality refers to the ability to make good decisions given the sensor information received. Moreover, in *some* environments, an agent can deliberately ignore part of its sensor information and still be perfectly rational. For example, a physical chess agent can ignore flecks of dust on the chessboard and pretty much anything that isn't on the chessboard or the clock.

b. *There exist task environments in which no pure reflex agent can behave rationally.*
True. A pure reflex agent ignores previous percepts, so cannot obtain an optimal state estimate in a partially observable environment. For example, correspondence chess is played by sending moves; if the other player's move is the current percept, a reflex agent could not keep track of the board state and would have to respond to, say, "a4" in the same way regardless of the position in which it was played.

c. *There exists a task environment in which every agent is rational.*
True. For example, in an environment with a single state, such that all actions have the same reward, it doesn't matter which action is taken. More generally, any environment that is reward-invariant under permutation of the actions will satisfy this property.

d. *The input to an agent program is the same as the input to the agent function.*
False. The agent function, notionally speaking, takes as input the entire percept sequence up to that point, whereas the agent program takes the current percept only.

e. *Every agent function is implementable by some program/machine combination.*
False. For example, the environment may contain Turing machines and input tapes and the agent's job is to solve the halting problem; there is an agent *function* that specifies the right answers, but no agent program can implement it. Another example would be an agent function that requires solving intractable problem instances of arbitrary size in constant time.

f. *Suppose an agent selects its action uniformly at random from the set of possible actions. There exists a deterministic task environment in which this agent is rational.*
True. This is a special case of (c); if it doesn't matter which action you take, selecting

randomly is rational.

g. *It is possible for a given agent to be perfectly rational in two distinct task environments.*
True. For example, we can arbitrarily modify the parts of the environment that are unreachable by any optimal policy as long as they stay unreachable.

h. *Every agent is rational in an unobservable environment.*
False. Some actions are stupid—and the agent may know this if it has a model of the environment—even if one cannot perceive the environment state.

i. *A perfectly rational poker-playing agent never loses.*
False. Unless it draws the perfect hand, the agent can always lose if an opponent has better cards. This can happen for game after game. The correct statement is that the agent's expected winnings are nonnegative.

j. *For a simple reflex agent in a partially observable environment, a randomized policy can outperform any deterministic policy.*
True. Some vacuum robots use this idea to get themselves "unstuck" from corners. A partially observable environment can have hidden state that makes any particular deterministically chosen action fail when executed in response to a given percept, whereas a randomized policy may eventually chose the right action (if there is one). As an example, consider a reflex agent that needs the PIN to ulock a phone. A randomized agent will eventually emit the correct sequence. whereas a deterministic agent can only emit the same sequence over and over.

k. *There is a model-based reflex agent that can remember all of its percepts.*
True. A model-based agent updates an internal state representation—its "memory"—for each new percept. Memorizing the percepts is possible (assuming unbounded memory, which is a reasonable caveat). This may not be the most perspicuous representation of the current state of the world, but any other such representation of the current state could be computed from it on demand, for example by running any other model-based reflex agent's state estimation algorithm on the memorized percept sequence.

l. *Suppose agent A1 is rational and agent A2 is irrational. There exists a task environment where A2's actual score will be greater than A1's actual score.*
True. Rational decisions are defined by expected outcomes, not actual outcomes. A1 might just be unlucky.

2.3 The Nature of Environments

Exercise 2.3.#PEAS

For each of the following activities, give a PEAS description of the task environment and characterize it in terms of the properties listed in Section 2.3.2.

- Playing soccer.
- Exploring the subsurface oceans of Titan.
- Shopping for used AI books on the Internet.
- Playing a tennis match.

- Practicing tennis against a wall.
- Performing a high jump.
- Knitting a sweater.
- Bidding on an item at an auction.

Many of these can actually be argued either way, depending on the level of detail and abstraction.

A. Partially observable, stochastic, sequential, dynamic, continuous, multi-agent.
B. Partially observable, stochastic, sequential, dynamic, continuous, single agent (unless there are alien life forms that are usefully modeled as agents).
C. Partially observable, deterministic, sequential, static, discrete, single agent. This can be multi-agent and dynamic if we buy books via auction, or dynamic if we purchase on a long enough scale that book offers change.
D. Fully observable, stochastic, episodic (every point is separate), dynamic, continuous, multi-agent.
E. Fully observable, stochastic, episodic, dynamic, continuous, single agent.
F. Fully observable, stochastic, sequential, static, continuous, single agent.
G. Fully observable, deterministic, sequential, static, continuous, single agent.
H. Fully observable, strategic, sequential, static, discrete, multi-agent.

Exercise 2.3.#VCES

Implement a performance-measuring environment simulator for the vacuum-cleaner world depicted in Figure 2.2 and specified on page 40. Your implementation should be modular so that the sensors, actuators, and environment characteristics (size, shape, dirt placement, etc.) can be changed easily. (*Note:* for some choices of programming language and operating system there are already implementations in the online code repository.)

The code repository implements a vacuum-cleaner environment. Students can easily extend it to generate different shaped rooms, obstacles, dirt generation processes, sensor suites, and so on.

Exercise 2.3.#ENVP

For each of the following task environment properties, rank the example task environments from most to least according to how well the environment satisfies the property. Lay out any assumptions you make to reach your conclusions.

a. *Fully Observable*: driving; document classification; tutoring a high-school student in calculus; skin cancer diagnosis from images.

b. *Continuous*: driving; spoken conversation; written conversation; climate engineering by stratospheric aerosol injection.

c. *Stochastic*: driving; sudoku; poker; soccer.

d. *Static*: chat room; checkers; tax planning; tennis.

a. *Fully Observable*: document classification > skin cancer diagnosis from images > driving > tutoring a high-school student in calculus.
Document classification is a fairly canonical example of a (non-sequential) observable problem, because the correct classification depends almost entirely on the visible text of the document itself. There might be a slight influence from "provenance" information (date, authorship, etc.) that may not be directly observable. Skin cancer diagnosis can sometimes be done well from an image of the lesion, bot other factors such as patient age, changes in the lesion over time, medical history, and family history can be important. Driving is often considered to be observable because we imagine that we are making decisions based on what we see, but (1) velocity and turn signal status of other vehicles can be judged only from multiple image frames, and (2) assessing the intended actions of other vehicles may require accumulating information over an extended period—e.g., to determine if a vehicle is stopped or broken down, driving slowly or looking for an address or a parking spot, turning left or has forgotten to turn off the turn signal. Other vehicles, hedges, fog, and so on can obscure visual access to important aspects of the driving environment. Tutoring is almost completely unobservable: what matters is the student's level of understanding, learning style, basic math skills, etc. Clues must be gathered over days, weeks, and months.

b. *Continuous*: climate engineering > driving > spoken conversation > written conversation.
Climate engineering by aerosol injection is quintessentially continuous: the engineer must decide how much to inject, where, and when, and all of these are continuous quantities. The control actions of driving are mostly continuous (steering, acceleration/braking) but there are discrete elements (turn signal, headlights). More importantly, the problem is usually handled using discrete high-level actions (change lanes left, take exit, etc.) that have implementations as continuous control problems. This kind of discrete/continuous hierarchy is very common; playing chess in the physical world is a perfect example. Spoken conversation is closer to chess than driving: roughly speaking, we choose the discrete words to say and delegate the saying to continuous motor control routines. Prosody (volume, pitch, and speed variation) is, however, an important continuous element in how we speak that is largely absent from written communication.

c. *Stochastic*: poker > soccer > driving > sudoku.
In poker, nearly everything is determined by the fall of the cards, which is entirely stochastic from the viewpoint of the players. Both soccer and driving contain elements that are fairly deterministic, such as the flight of the ball and the response of the engine, and elements that are stochastic, such as tire punctures and the outcomes of tackles. Yet typically one can make reasonably reliable driving plans over many minutes, whereas it

is essentially impossible to predict the state of a soccer game one minute into the future. Sudoku, of course, is entirely deterministic.

d. *Static*: tax planning > checkers > chat room > tennis.
While no human activity is completely static, given the finite length of our lifetimes, tax planning comes close—the typical "deadline" to get it done is often weeks or months, and the relevant aspects of the environment (life/death, number of offspring, tax law) may change even more slowly. In checkers, the world state doesn't change until someone moves, but the clock ticks so the problem is semi-dynamic. In the chat room, long delays in replying are unacceptable, so it is a fairly real-time environment, but not nearly as real-time as tennis, where a delay of a split second often makes the difference between winning and losing a point.

2.4 The Structure of Agents

Exercise 2.4.#SIRA

Implement a simple reflex agent for the vacuum environment in Exercise VACUUM-START-EXERCISE. Run the environment with this agent for all possible initial dirt configurations and agent locations. Record the performance score for each configuration and the overall average score.

Pseudocode for a reflex agent is given in Figure 2.8. For states 1, 3, 5, 7 in Figure 4.9, the performance measures are 1996, 1999, 1998, 2000 respectively. The average is 1998.25.

Exercise 2.4.#VCPE

Consider a modified version of the vacuum environment in Exercise VACUUM-START-EXERCISE, in which the agent is penalized one point for each movement.

a. Can a simple reflex agent be perfectly rational for this environment? Explain.

b. What about a reflex agent with state? Design such an agent.

c. How do your answers to **a** and **b** change if the agent's percepts give it the clean/dirty status of every square in the environment?

a. No. Consider the reflex actions for $[A, Clean]$ and $[B, Clean]$. If either of these is $NoOp$, then the agent will fail in the case where that is the initial percept but the other square is dirty; hence, neither can be $NoOp$ and therefore the simple reflex agent is doomed to keep moving, which loses points unnecessarily. In general, the problem with reflex agents is that they have to do the same thing in situations that *look the same*, even when the situations are *actually quite different*. In the vacuum world this is a big liability, because every interior square (except home) looks either like a square with dirt or a square without dirt.

b. Yes, a reflex agent with state can be perfectly rational. It simply needs to remember whether it has visited the other square. If so, it can do nothing after cleaning the current

square (if needed). If not, it then goes to the other square and cleans it if needed.

c. In this case, a simple reflex agent can be perfectly rational. The agent can consist of a table with eight entries, indexed by percept, that specifies an action to take for each possible state. After the agent acts, the world is updated and the next percept will tell the agent what to do next. For larger environments, constructing a table is infeasible. Instead, the agent could run one of the optimal search algorithms in Chapter 3 and execute the first step of the solution sequence. Again, no internal state is *required*, but it would help to be able to store the solution sequence instead of recomputing it for each new percept.

Exercise 2.4.#VACU

Consider a modified version of the vacuum environment in Exercise VACUUM-START-EXERCISE, in which the geography of the environment—its extent, boundaries, and obstacles—is unknown, as is the initial dirt configuration. (The agent can go Up and $Down$ as well as $Left$ and $Right$.)

a. Can a simple reflex agent be perfectly rational for this environment? Explain.

b. Can a simple reflex agent with a *randomized* agent function outperform a simple reflex agent? Design such an agent and measure its performance on several environments.

c. Can you design an environment in which your randomized agent will perform poorly? Show your results.

d. Can a reflex agent with state outperform a simple reflex agent? Design such an agent and measure its performance on several environments. Can you design a rational agent of this type?

a. Because the agent does not know the geography and perceives only location and local dirt, and cannot remember what just happened, it will get stuck forever against a wall when it tries to move in a direction that is blocked—that is, unless it randomizes.

b. One possible design cleans up dirt and otherwise moves in a randomly chosen direction. This is fairly close to what the early-model Roomba$^{\text{TM}}$ vacuum cleaner does, although the Roomba has a bump sensor and randomizes only when it hits an obstacle. It works reasonably well but it can take a long time to cover all the squares at least once. Many modern robot cleaners build a map and derive an efficient cleaning pattern.

c. An example is shown in Figure S2.1. Students may also wish to measure clean-up time for linear or square environments of different sizes, and compare those to the efficient online search algorithms described in Chapter 4. It's worth noting that in the infinite-time limit on a bidirectional graph, an agent that moves randomly will spend time in each square proportional to its degree in the graph (the number of neighbors). Still, it can take a long time to get from one part of the graph to another. If the agent is in a long corridor, its expected travel after n time steps is $O(\sqrt{n})$ squares.

d. A reflex agent with state can build a map (see Chapter 4 for details). An online depth-first exploration will reach every state in time linear in the size of the environment; therefore, the agent can do much better than the simple reflex agent.

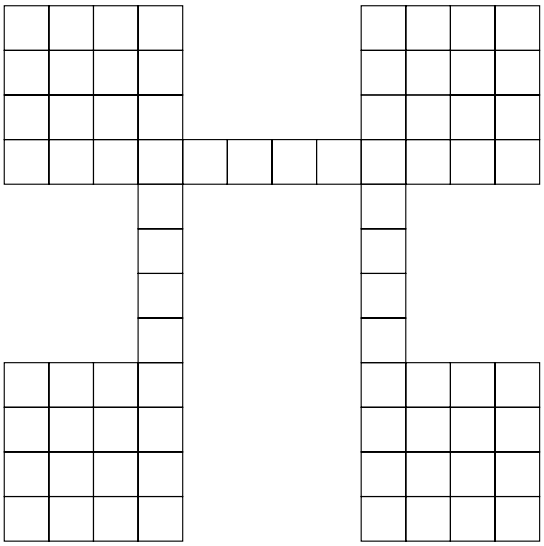

Figure S2.1 An environment in which random motion will take a long time to cover all the squares.

The question of rational behavior in unknown environments is a complex one but it is worth encouraging students to think about it. We need to have some notion of the prior probability distribution over the class of environments; call this the initial **belief state**. Any action yields a new percept that can be used to update this distribution, moving the agent to a new belief state. Once the environment is completely explored, the belief state collapses to a single possible environment. Therefore, the problem of optimal exploration can be viewed as a search for an optimal strategy in the space of possible belief states. This is a well-defined, if horrendously intractable, problem. Chapter 17 discusses some cases (called **bandit problems**) where optimal exploration is possible. Another concrete example of exploration is the Minesweeper computer game (see Exercise 7.22). For very small Minesweeper environments, optimal exploration is feasible although the belief state update is nontrivial to explain.

Exercise 2.4.#VACB

Repeat Exercise VACUUM-UNKNOWN-GEOG-EXERCISE for the case in which the location sensor is replaced with a "bump" sensor that detects the agent's attempts to move into an obstacle or to cross the boundaries of the environment. Suppose the bump sensor stops working; how should the agent behave?

The problem appears at first to be very similar; the main difference is that instead of using the location percept to build the map, the agent has to "invent" its own locations (which, after all, are just nodes in a data structure representing the state space graph). When a bump is detected, the agent assumes it remains in the same location and can add a wall to its map. For grid environments, the agent can keep track of its (x, y) location and so can tell when it has

returned to an old state. In the general case, however, there is no simple way to tell if a state is new or old.

Exercise 2.4.#VFIN

The vacuum environments in the preceding exercises have all been deterministic. Discuss possible agent programs for each of the following stochastic versions:

a. Murphy's law: twenty-five percent of the time, the $Suck$ action fails to clean the floor if it is dirty and deposits dirt onto the floor if the floor is clean. How is your agent program affected if the dirt sensor gives the wrong answer 10% of the time?

b. Small children: At each time step, each clean square has a 10% chance of becoming dirty. Can you come up with a rational agent design for this case?

a. For a reflex agent, this presents no *additional* challenge, because the agent will continue to $Suck$ as long as the current location remains dirty. For an agent that constructs a sequential plan, every $Suck$ action would need to be replaced by "$Suck$ until clean." If the dirt sensor can be wrong on each step, then the agent might want to wait for a few steps to get a more reliable measurement before deciding whether to $Suck$ or move on to a new square. Obviously, there is a trade-off because waiting too long means that dirt remains on the floor (incurring a penalty), but acting immediately risks either dirtying a clean square or ignoring a dirty square (if the sensor is wrong). A rational agent must also continue touring and checking the squares in case it missed one on a previous tour (because of bad sensor readings). It is not immediately obvious how the waiting time at each square should change with each new tour. These issues can be clarified by experimentation, which may suggest a general trend that can be verified mathematically. This problem is a partially observable Markov decision process—see Chapter 17. Such problems are hard in general, but some special cases may yield to careful analysis.

b. In this case, the agent must keep touring the squares indefinitely. The probability that a square is dirty increases monotonically with the time since it was last cleaned, so the rational strategy is, roughly speaking, to repeatedly execute the shortest possible tour of all squares. (We say "roughly speaking" because there are complications caused by the fact that the shortest tour may visit some squares twice, depending on the geography.) This problem is also a partially observable Markov decision process.

Exercise 2.4.#DFRA

Define in your own words the following terms: rationality, autonomy, reflex agent, model-based agent, goal-based agent, utility-based agent, learning agent.

The following are just some of the many possible definitions that can be written:

- *Rationality*: a property of agents that choose actions that maximize their expected utility, given the percepts to date.

- *Autonomy*: a property of agents whose behavior is determined by their own experience rather than solely by their initial programming.
- *Reflex agent*: an agent whose action depends only on the current percept.
- *Model-based agent*: an agent whose action is derived directly from an internal model of the current world state that is updated over time.
- *Goal-based agent*: an agent that selects actions that it believes will achieve explicitly represented goals.
- *Utility-based agent*: an agent that selects actions that it believes will maximize the expected utility of the outcome state.
- *Learning agent*: an agent whose behavior improves over time based on its experience.

Exercise 2.4.#GBUT

Write pseudocode agent programs for the goal-based and utility-based agents.

The design of goal- and utility-based agents depends on the structure of the task environment. The simplest such agents, for example those in chapters 3 and 10, compute the agent's entire future sequence of actions in advance before acting at all. This strategy works for static and deterministic environments which are either fully-known or unobservable

For fully-observable and fully-known static environments a policy can be computed in advance which gives the action to by taken in any given state.

For partially-observable environments the agent can compute a conditional plan, which specifies the sequence of actions to take as a function of the agent's perception. In the extreme, a conditional plan gives the agent's response to every contingency, and so it is a representation of the entire agent function.

In all cases it may be either intractable or too expensive to compute everything out in advance. Instead of a conditional plan, it may be better to compute a single sequence of actions which is likely to reach the goal, then monitor the environment to check whether the plan is succeeding, repairing or replanning if it is not. It may be even better to compute only the start of this plan before taking the first action, continuing to plan at later time steps.

Pseudocode for simple goal-based agent is given in Figure S2.2. GOAL-ACHIEVED tests to see whether the current state satisfies the goal or not, doing nothing if it does. PLAN computes a sequence of actions to take to achieve the goal. This might return only a prefix of the full plan, the rest will be computed after the prefix is executed. This agent will act to maintain the goal: if at any point the goal is not satisfied it will (eventually) replan to achieve the goal again.

At this level of abstraction the utility-based agent is not much different than the goal-based agent, except that action may be continuously required (there is not necessarily a point where the utility function is "satisfied"). Pseudocode is given in Figure S2.3.

Exercise 2.4.#FURN

Consider a simple thermostat that turns on a furnace when the temperature is at least 3 degrees below the setting, and turns off a furnace when the temperature is at least 3 degrees

```
function GOAL-BASED-AGENT(ṗercept) returns an action
  persistent: ṡtate, the agent's current conception of the world state
              ṁodel, a description of how the next state depends on current state and action
              ġoal, a description of the desired goal state
              ṗlan, a sequence of actions to take, initially empty
              ȧction, the most recent action, initially none

  ṡtate ← UPDATE-STATE(ṡtate, ȧction, ṗercept, ṁodel)
  if GOAL-ACHIEVED(ṡtate,ġoal) then return a null action
  if ṗlan is empty then
ṗlan ← PLAN(ṡtate,ġoal,ṁodel)
  ȧction ← FIRST(ṗlan)
  ṗlan ← REST(ṗlan)
  return ȧction
```

Figure S2.2 A goal-based agent.

```
function UTILITY-BASED-AGENT(ṗercept) returns an action
  persistent: ṡtate, the agent's current conception of the world state
              ṁodel, a description of how the next state depends on current state and action
              ŭtility-function, a description of the agent's utility function
              ṗlan, a sequence of actions to take, initially empty
              ȧction, the most recent action, initially none

ṡtate ← UPDATE-STATE(ṡtate, ȧction, ṗercept, ṁodel)
if ṗlan is empty then
ṗlan ← PLAN(ṡtate,ŭtility-function,ṁodel)
ȧction ← FIRST(ṗlan)
ṗlan ← REST(ṗlan)
return ȧction
```

Figure S2.3 A utility-based agent.

above the setting. Is a thermostat an instance of a simple reflex agent, a model-based reflex agent, or a goal-based agent?

The thermostat is best understood as a simple reflex agent. Although the temperature setting might be viewed as a goal, the agent has no notion of how its actions will lead to the goal; so it's better to view the temperature setting as part of the agent's percepts. A more complex control system might well have an internal model of the house, the behavior of its occupants, the capabilities of the heating system, and so on, and would be viewed as a goal-based system in much the same way as a self-driving car that tries to reach a specified

destination quickly and cheaply.

Exercise 2.4.#AGPR

Here is pseudocode for three agent programs A, B, C:

```
function A(percept)
return f_A()
```

```
function B(percept)
return f_B(percept)
```

```
function C(percept)
persistent: percepts, initially []
percepts ← push(percept,percepts)
return f_C(percepts)
```

In each of these agents, the function f is some arbitrary, possibly randomized, function of its inputs with no internal state of its own; the agent program runs on a computer with unbounded memory but finite clock speed. We'll assume also that the environment and its performance measure are computable.

a. Suppose the environment is fully observable, deterministic, discrete, single-agent, and static. For which agents, if any, is it the case that, for *every* such environment, there is *some* way to choose f such that the agent is perfectly rational?

b. Suppose the environment is *partially* observable, deterministic, discrete, single-agent, and static. For which agents, if any, is it the case that, for *every* such environment, there is *some* way to choose f such that the agent is perfectly rational?

c. Suppose the environment is partially observable, *stochastic*, discrete, single-agent, and *dynamic*. For which agents, if any, is it the case that, for *every* such environment, there is *some* way to choose f such that the agent is perfectly rational?

a. B, C. For a fully observable environment, only the current percept is required for an optimal decision. Because the environment is static, computation is not an issue. Note that Agent A cannot make optimal decisions because it always makes the *same* decision (or samples a decision from the same probability distribution), having no internal state.

b. C. Agent B, the reflex agent, cannot always function optimally in a partially observable environment because it ignores previous percepts and therefore fails to take into account relevant information for the current decision.

c. None of the agents can be optimal for an arbitrary dynamic environment, because we can make the environment complex enough to render optimal decisions infeasible for any finite-speed machine.

EXERCISES 3

SOLVING PROBLEMS BY SEARCHING

3.1 Problem-Solving Agents

Exercise 3.1.#FORM

Explain why problem formulation must follow goal formulation.

In goal formulation, we decide which aspects of the world we are interested in, and which can be ignored or abstracted away. Then in problem formulation we decide how to manipulate the important aspects (and ignore the others). If we did problem formulation first we would not know what to include and what to leave out. That said, it can happen that there is a cycle of iterations between goal formulation, problem formulation, and problem solving until one arrives at a sufficiently useful and efficient solution.

3.2 Example Problems

Exercise 3.2.#PRFO

Give a complete problem formulation for each of the following problems. Choose a formulation that is precise enough to be implemented.

a. There are six glass boxes in a row, each with a lock. Each of the first five boxes holds a key unlocking the next box in line; the last box holds a banana. You have the key to the first box, and you want the banana.

b. You start with the sequence ABABAECCEC, or in general any sequence made from A, B, C, and E. You can transform this sequence using the following equalities: AC = E, AB = BC, BB = E, and Ex = x for any x. For example, ABBC can be transformed into AEC (using BB = E), and then AC (using Ex = x), and then E (using AC = E). Your goal is to produce the sequence E.

c. There is an $n \times n$ grid of squares, each square initially being either unpainted floor or a bottomless pit. You start standing on an unpainted floor square, and can either paint the square under you or move onto an adjacent unpainted floor square. You want the whole floor painted.

d. A container ship is in port, loaded high with containers. There 13 rows of containers, each 13 containers wide and 5 containers tall. You control a crane that can move to any

location above the ship, pick up the container under it, and move it onto the dock. You want the ship unloaded.

a. Initial state: as described in the question.
Goal test: you have banana.
Successor function: open any box you have the key for, get the contents of any open box.
Cost function: number of actions.

b. Initial state: ABABAECCEC.
Goal test: is the state E
Successor function: apply an equality substituting one subsequence for the other.
Cost function: number of transformations.

c. Initial state: all floor squares unpainted, you start standing on one square unpainted floor square.
Goal test: all floor squares painted.
Successor function: paint current tile, move to adjacent unpainted floor tile.
Cost function: number of moves.

d. Initial state: all containers stacked on the ship.
Goal test: all containers unloaded.
Successor function: move crane to a certain location, pick up a container, put down container.
Cost function: time taken to unload ship.

Exercise 3.2.#RMAZ

Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

a. Formulate this problem. How large is the state space?

b. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?

c. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?

d. In our initial description of the problem we already abstracted from the real world, restricting actions and removing details. List three such simplifications we made.

a. We'll define the coordinate system so that the center of the maze is at $(0, 0)$, and the maze itself is a square from $(-1, -1)$ to $(1, 1)$.

Initial state: robot at coordinate $(0, 0)$, facing North.
Goal test: either $|x| > 1$ or $|y| > 1$ where (x, y) is the current location.
Successor function: move forwards any distance d; change direction robot it facing.
Cost function: total distance moved.

The state space is infinitely large, since the robot's position is continuous.

b. The state will record the intersection the robot is currently at, along with the direction it's facing. At the end of each corridor leaving the maze we will have an exit node. We'll assume some node corresponds to the center of the maze.

Initial state: at the center of the maze facing North.
Goal test: at an exit node.
Successor function: move to the next intersection in front of us, if there is one; turn to face a new direction.
Cost function: total distance moved.

There are $4n$ states, where n is the number of intersections.

c. Initial state: at the center of the maze.
Goal test: at an exit node.
Successor function: move to next intersection to the North, South, East, or West.
Cost function: total distance moved.

We no longer need to keep track of the robot's orientation since it is irrelevant to predicting the outcome of our actions, and not part of the goal test. The motor system that executes this plan will need to keep track of the robot's current orientation, to know when to rotate the robot.

d. State abstractions:

(i) Ignoring the height of the robot off the ground, whether it is tilted off the vertical.
(ii) The robot can face in only four directions.
(iii) Other parts of the world ignored: possibility of other robots in the maze, the weather in the Caribbean.

Action abstractions:

(i) We assumed all positions we safely accessible: the robot couldn't get stuck or damaged.
(ii) The robot can move as far as it wants, without having to recharge its batteries.
(iii) Simplified movement system: moving forwards a certain distance, rather than controlled each individual motor and watching the sensors to detect collisions.

3.3 Search Algorithms

3.4 Uninformed Search Strategies

3.5 Informed (Heuristic) Search Strategies

3.6 Heuristic Functions

Exercise 3.6.#GRRB

You have a 9×9 grid of squares, each of which can be colored red or blue. The grid is initially colored all blue, but you can change the color of any square any number of times. Imagining the grid divided into nine 3×3 sub-squares, you want each sub-square to be all one color but neighboring sub-squares to be different colors.

a. Formulate this problem in the straightforward way. Compute the size of the state space.

b. You need color a square only once. Reformulate, and compute the size of the state space. Would breadth-first graph search perform faster on this problem than on the one in (a)? How about iterative deepening tree search?

c. Given the goal, we need consider only colorings where each sub-square is uniformly colored. Reformulate the problem and compute the size of the state space.

d. How many solutions does this problem have?

e. Parts (b) and (c) successively abstracted the original problem (a). Can you give a translation from solutions in problem (c) into solutions in problem (b), and from solutions in problem (b) into solutions for problem (a)?

a. Initial state: all squares colored blue.
Goal test: each sub-square colored the same, and adjacent subsquares colored different.
Successor function: color a square red or blue.
Cost function: number of times squares are colored.

There are 2^{81} states.

b. To avoid recoloring squares we could either record squares that have been colored, and not allow them to be colored again, or color all the squares in a fixed order. We describe the latter:

Initial state: all squares colored blue, the top-left square is next to be colored.
Goal test: each sub-square colored the same, and adjacent subsquares colored different.
Successor function: if we haven't colored all squares yet, color the current square red or blue (this automatically updates the next square to be colored).
Cost function: number of squares colored.

There are $82 \cdot 2^{81}$ states, since we need to record whether there are squares left to be colored, and if so which square is next.

c. Initial state: all sub-squares colored blue, the top-left sub-square is next to be colored.
Goal test: adjacent sub-squares colored differently.
Successor function: if we haven't colored all sub-squares yet, color the current sub-square red or blue.
Cost function: number of sub-squares colored.

The state space has $10 \cdot 2^9$ nodes.

d. This problem has 2 solutions: as soon as you choose the color of any square, the color of any other square is determined.

e. Given a solution for problem (c) first determine final coloring of squares, after all actions are taken. Then color each of the squares the required color in order.

Given a solution for problem (b) first determine final coloring of squares. We know that each sub-square will be colored a single color, so we can color them in order to reach the same goal state.

Exercise 3.6.#ROMF

Suppose two friends live in different cities on a map, such as the Romania map shown in Figure 3.1. On every turn, we can simultaneously move each friend to a neighboring city on the map. The amount of time needed to move from city i to neighbor j is equal to the road distance $d(i, j)$ between the cities, but on each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible.

a. Write a detailed formulation for this search problem. (You will find it helpful to define some formal notation here.)

b. Let $D(i, j)$ be the straight-line distance between cities i and j. Which of the following heuristic functions are admissible? (i) $D(i, j)$; (ii) $2 \cdot D(i, j)$; (iii) $D(i, j)/2$.

c. Are there completely connected maps for which no solution exists?

d. Are there maps in which all solutions require one friend to visit the same city twice?

a. State space: States are all possible city pairs (i, j). The map is *not* the state space.

Successor function: The successors of (i, j) are all pairs (x, y) such that $Adjacent(x, i)$ and $Adjacent(y, j)$.

Goal: Be at (i, i) for some i.

Step cost function: The cost to go from (i, j) to (x, y) is $max(d(i, x), d(j, y))$.

b. In the best case, the friends head straight for each other in steps of equal size, reducing their separation by twice the time cost on each step. Hence (iii) is admissible.

c. Yes: e.g., a map with two nodes connected by one link. The two friends will swap places forever. The same will happen on any chain if they start an odd number of steps apart. (One can see this best on the graph that represents the state space, which has two disjoint sets of nodes.) The same even holds for a grid of any size or shape, because every move changes the Manhattan distance between the two friends by 0 or 2.

d. Yes: take any of the unsolvable maps from part (c) and add a self-loop to any one of the nodes. If the friends start an odd number of steps apart, a move in which one of the friends takes the self-loop changes the distance by 1, rendering the problem solvable. If the self-loop is not taken, the argument from (c) applies and no solution is possible.

Exercise 3.6.#PART

Show that the 8-puzzle states are divided into two disjoint sets, such that any state is reachable from any other state in the same set, while no state is reachable from any state in the other set. (*Hint:* See Berlekamp *et al.*, (1982).) Devise a procedure to decide which set a given state is in, and explain why this is useful for generating random states.

From `http://www.cut-the-knot.com/pythagoras/fifteen.shtml`, this proof applies to the fifteen puzzle, but the same argument works for the eight puzzle:

Definition: The goal state has the numbers in a certain order, which we will measure as starting at the upper left corner, then proceeding left to right, and when we reach the end of a row, going down to the leftmost square in the row below. For any other configuration besides the goal, whenever a tile with a greater number on it precedes a tile with a smaller number, the two tiles are said to be **inverted**.

Proposition: For a given puzzle configuration, let N denote the sum of the total number of inversions and the row number of the empty square. Then $(N \mod 2)$ is invariant under any legal move. In other words, after a legal move an odd N remains odd whereas an even N remains even. Therefore the goal state in Figure 3.3, with no inversions and empty square in the first row, has $N = 1$, and can only be reached from starting states with odd N, not from starting states with even N.

Proof: First of all, sliding a tile horizontally changes neither the total number of inversions nor the row number of the empty square. Therefore let us consider sliding a tile vertically.

Let's assume, for example, that the tile A is located directly over the empty square. Sliding it down changes the parity of the row number of the empty square. Now consider the total number of inversions. The move only affects relative positions of tiles A, B, C, and D. If none of the B, C, D caused an inversion relative to A (i.e., all three are larger than A) then after sliding one gets three (an odd number) of additional inversions. If one of the three is smaller than A, then before the move B, C, and D contributed a single inversion (relative to A) whereas after the move they'll be contributing two inversions - a change of 1, also an odd number. Two additional cases obviously lead to the same result. Thus the change in the sum N is always even. This is precisely what we have set out to show.

So before we solve a puzzle, we should compute the N value of the start and goal state and make sure they have the same parity, otherwise no solution is possible.

Exercise 3.6.#NQUE

Consider the n-queens problem using an efficient incremental formulation where a state is represented as a an n-element vector of row numbers for queens (one for each column). Explain why the state space has at least $\sqrt[3]{n!}$ states and estimate the largest n for which exhaustive exploration is feasible. (*Hint*: Derive a lower bound on the branching factor by considering the maximum number of squares that a queen can attack in any column.)

The formulation puts one queen per column, with a new queen placed only in a square

that is not attacked by any other queen. To simplify matters, we'll first consider the n–rooks problem. The first rook can be placed in any square in column 1 (n choices), the second in any square in column 2 except the same row that as the rook in column 1 ($n-1$ choices), and so on. This gives $n!$ elements of the search space.

For n queens, notice that a queen attacks at most three squares in any given column, so in column 2 there are at least $(n-3)$ choices, in column at least $(n-6)$ choices, and so on. Thus the state space size $S \geq n \cdot (n-3) \cdot (n-6) \cdots$. Hence we have

$$\begin{aligned} S^3 &\geq n \cdot n \cdot n \cdot (n-3) \cdot (n-3) \cdot (n-3) \cdot (n-6) \cdot (n-6) \cdot (n-6) \cdot \cdots \\ &\geq n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdot (n-4) \cdot (n-5) \cdot (n-6) \cdot (n-7) \cdot (n-8) \cdot \cdots \\ &= n! \end{aligned}$$

or $S \geq \sqrt[3]{n!}$.

Exercise 3.6.#COMP

Give a complete problem formulation for each of the following. Choose a formulation that is precise enough to be implemented.

a. Using only four colors, you have to color a planar map in such a way that no two adjacent regions have the same color.

b. A 3-foot-tall monkey is in a room where some bananas are suspended from the 8-foot ceiling. The monkey would like to get the bananas. The room contains two stackable, movable, climbable 3-foot-high crates.

c. You have a program that outputs the message "illegal input record" when fed a certain file of input records. You know that processing of each record is independent of the other records. You want to discover what record is illegal.

d. You have three jugs, measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. You can fill the jugs up or empty them out from one to another or onto the ground. You need to measure out exactly one gallon.

a. Initial state: No regions colored.
Goal test: All regions colored, and no two adjacent regions have the same color.
Successor function: Assign a color to a region.
Cost function: Number of assignments.

b. Initial state: As described in the text.
Goal test: Monkey has bananas.
Successor function: Hop on crate; Hop off crate; Push crate from one spot to another; Walk from one spot to another; grab bananas (if standing on crate).
Cost function: Number of actions.

c. Initial state: considering all input records.
Goal test: considering a single record, and it gives "illegal input" message.
Successor function: run again on the first half of the records; run again on the second half of the records.
Cost function: Number of runs.