- Nyelv: A kérdések megválaszolásának pontossága, az F1 teszt-statisztikával mérve, a Stanford Question Answering Dataseten (SQuAD) 2015-ről 2019-re 60-ról 95-re nőtt. A SQuAD 2 változaton a haladás gyorsabb volt, mindössze egy év alatt 62-ről 90-re. Mindkét pontszám meghaladja az emberi szintű teljesítményt.
- Emberi teljesítménymérés: 2019-re az MI-rendszerek emberhez hasonlóan teljesítettek, vagy az emberi teljesítményt meghaladták: a sakkban, a góban, a pókerben, a Pac-Manben, a Jeopardy!-ban, objektumfelismerésben ImageNeten, korlátozott témájú kínai–angol fordításban, Quake III-ben, Dota 2-ban, StarCraft II-ban, különféle Atari-játékokban, bőrrákkimutatásban, prosztatarák-detektálásban, fehérjehajtogatásban és diabetikus retinopátia diagnosztizálásában.

Mikorra érhetik el (ha valaha is elérik) az MI-rendszerek az emberi szintű teljesítményt a feladatok sokféleségében? Ford (2018) interjúkat folytatott az MI szakértőivel, és a megcélzott évek széles skáláját tárta fel, 2029-től egészen 2200-ig, 2099-es átlaggal. Egy hasonló felmérésben (Grace *et al.*, 2017) a válaszadók 50%-a szerint ez már a 2066-os évben lehetséges lenne, bár 10% olyan közelre gondolt, mint a 2025-ös év, néhányan viszont „sohá"-val válaszoltak. A szakértők abban is megoszlanak, hogy szükségünk van-e új áttörésekre, vagy csak a jelenlegi megközelítések finomítására. De ne vegyük az előrejelzéseiket túl komolyan. Amint ezt Philip Tetlock (2017) megmutatja, a világ eseményeinek előrejelzésében a szakértők az amatőröknél nem jobbak.

Hogyan fognak a jövőbeli MI-rendszerek működni? Ezt még nem tudjuk megmondani. Ahogyan ezt a jelen fejezetben részleteztük, a terület több történetet fogadtatott el magáról – először azt a merész gondolatot, hogy a gépi intelligencia egyáltalán lehetséges, majd hogy ezt a szakértői tudás logikába történő kódolásával el lehet érni, ezt követően pedig, hogy a világ valószínűségi modelljei lennének a fő eszköz, és a legutóbb, hogy a gépi tanulás olyan modelleket hozhat életre, amelyek semmilyen jól megértett elméleten nem kell hogy alapuljanak. A jövő megmutatja, hogy milyen modell lesz a következő.

Mit tehet az MI ma? Talán nem annyit, mint néhány optimista médiacikk szeretné velünk elhitetni, ám mégis nagyon sokat. Itt van néhány példa:

Robotikus járművek: A robotikus járművek története a 1920-as évek rádióvezérelt autókra nyúlik vissza, de az autonóm közúti vezetés első bemutatása, külön vezető nélkül, az 1980-as években történt (Kanade *et al.*, 1986; Dickmanns and Zapp, 1987). Miután a 2005-ben megrendezett, 213 km-es DARPA Grand Challenge-en földutakon (Thrun, 2006), majd 2007-ben az Urban Challenge-en utcai forgalomban is bizonyítottak, az önvezető autók fejlesztésére irányuló verseny komolyan beindult. 2018-ban a Waymo tesztjárművek elérték a 16 millió km-es, súlyos baleset nélküli közúti vezetés mérföldkövét, ahol az emberi gépkocsivezető átlagosan csak 10 000 km-enként vette át az irányítást. Nem sokkal ezután a cég elkezdett kereskedelmi robot-taxi szolgáltatást kínálni. A levegőben, autonóm, rögzített szárnyú drónok 2016 óta vért szállítanak a vadonban, Ruandában. Kvadkopterek figyelemre méltó műrepülő manővereket hajtanak végre, háromdimenziós térképek készítése közben tárnak fel épületeket, és autonóm formációkba képesek rendeződni.

Lábon járó robotok: BigDog, a Raibert *et al.* (2008) négylábú robotja fejre állította az elképzeléseinket, hogy egy robot hogyan mozog – többé már nem a hollywoodi

filmrobotok lassú, merev lábú, jobbra-balra billegő járása a jellemző, hanem valami olyan, ami nagyon hasonlít egy állatra, és képes az egyensúlyát visszanyerni, ha meglökik, vagy ha egy jeges pocsolyán megcsúszik. Atlas, egy humanoid robot, nemcsak sétálni képes egyenetlen terepen, hanem felugrik dobozokra, és hátrabukfenceket is csinál (Ackerman and Guizzo, 2016).

Autonóm tervkészítés és ütemezés: Több száz millió mérföldre a Földtől a NASA Remote Agent programja lett az első fedélzeti autonóm tervkészítő program, amely egy űrhajó műveleteinek ütemezését felügyelte (Jonsson *et al.*, 2000). A Remote Agent a terveit a Földről küldött magas szintű célokból generálta, és a tervek végrehajtása közben monitorozta az űrhajó működését, hibákat detektált, diagnosztizált és visszaállította a helyes működést, ha problémák léptek fel. Ma az Europa tervkészítő eszközkészletet (Barreiro *et al.*, 2012) a NASA Mars roverének napi műveleteiben használják, a Sextant rendszer (Winternitz, 2017) viszont lehetővé teszi a mélyűrben való navigációt, a globális GPS-rendszer határain túl.

Az 1991. évi Perzsa-öböl-válság alatt az amerikai erők a Dynamic Analysis and Replanning Tool Dart (Cross and Walker, 1994) rendszert vetették be a szállítás automatizált logisztikai tervezéséhez és ütemezéséhez. Ez egyszerre akár 50 000 járművet, rakományt és embert jelentett, továbbá a rendszernek kezelnie kellett a kiindulási pontokat, rendeltetési helyeket, útvonalakat, szállítási kapacitásokat, kikötői és repülőtéri kapacitásokat és az összes paraméter között fellépő konfliktusok feloldását. A Honvédelmi Kutatási Ügynökség (Defense Advanced Research Project Agency – DARPA) közleménye szerint csupán ezen egyetlen alkalmazás kapcsán megtérült a DARPA által 30 éven keresztül az MI-re fordított befektetés.

Nap mint nap olyan hívható személyszállító társaságok, mint az Uber, és olyan térképezési szolgáltatások, mint a Google Maps, gépkocsivezetők százmilliói számára nyújtanak vezetési irányinformációt, gyorsan megjelölve olyan optimális útvonalat, amely a jelenlegi és a jövőben várható forgalmi körülményeket figyelembe veszi.

Gépi fordítás: Az online gépi fordító rendszerek manapság lehetővé teszik a dokumentumok olvasását több mint 100 nyelven, ideértve az emberek több mint 99%-ának az anyanyelvét, és napi több száz milliárdnyi szót nyújtanak több száz millió felhasználó számára. Bár nem tökéletesek, a megértéshez általában megfelelőek. A szorosan kapcsolódó nyelveknél, ahol sok tanítóadat áll rendelkezésre (mint amilyen például a francia és az angol), a szűk szakterületet érintő fordítások már az emberi szinthez közel állnak (Wu *et al.*, 2016b).

Beszédfelismerés: 2017-ben a Microsoft megmutatta, hogy a Conversational Speech Recognition System terméke 5,1%-os szó hibaarányt ért el, amely megegyezett az emberi teljesítménnyel a telefonközpont jellegű feladatokban, ami magában foglalja a telefonbeszélgetések leiratozását (Xiong *et al.*, 2017). Világszerte a számítógépes kölcsönhatásoknak körülbelül egyharmada már hangalapú, és nem billentyűzeten keresztül történik. A Skype valós idejű beszéd–beszéd fordítást biztosít tíz nyelven. Az Alexa, Siri, Cortana és a Google kérdéseket megválaszoló és a felhasználó részére feladatokat végrehajtó asszisztenseket kínálnak. A Google Duplex szolgáltatás például beszédfelismerést és beszédszintézist használ éttermi asztalfoglalásokhoz a felhasználók részére, folyékony beszélgetést folytatva a nevükben.

Ajánlórendszerek: Olyan cégek, mint az Amazon, a Facebook, a Netflix, a Spotify, a YouTube, a Walmart és mások, gépi tanuláshoz folyamodnak, hogy javaslatokat tegyenek, mit is kívánhatunk a múltbeli tapasztalataink és a hozzánk hasonló más felhasználók tapasztalatai alapján. Az ajánlórendszerek területének hosszú a története (Resnick and Varian, 1997), amely a tartalmat (szöveg, zene, videó), valamint az előzményeket és a metaadatokat elemző új mélytanulási módszerek révén most gyorsan változik (van den Oord *et al.*, 2014; Zhang *et al.*, 2017). A spamszűrés az ajánlás (vagy az ajánlás elutasításának) egyik formájának tekinthető. A jelenlegi MI-technikák kiszűrik a spam levelek több mint 99,9%-át. Az e-mail szolgáltatások képesek a potenciális címzettek és a lehetséges válaszszövegek ajánlására.

Kétszemélyes játékok: Amikor a Deep Blue 1997-ben legyőzte Garry Kaszparovot, a sakkvilágbajnokot, az emberi fensőbbség védelmezői a reményeiket a góba helyezték. Az asztrofizikus és górajongó Piet Hut megjósolta, hogy ahhoz, „hogy egy számítógép az embereket góban legyőzze, évszázadoknak – talán még hosszabb időnek kell eltelnie". De épp csak 20 évvel később, az AlphaGo az összes ember játékát felülmúlta (Silver *et al.*, 2017). Ke Jie, a világbajnok, azt mondta: „Tavaly még meglehetősen emberszerűen játszott. De ebben az évben olyan lett, mint a gó istene." Az AlphaGo hasznot húzott az emberi gójátékosok által eddig játszott játékok százezreinek tanulmányozásából és a kutatócsapatban részt vevő szakértő gójátékosok megtisztított tudásából.

Az AlphaZero volt a következő góprogram, amely már nem használt emberektől bemenetet (a játék szabályait kivéve), és képes volt saját maga ellen játszva megtanulni minden, akár emberi, akár gépi ellenfelét legyőzni góban, sakkban és shogiban (Silver *et al.*, 2018). Eközben az MI-rendszerek olyan játékokban verték meg az emberi bajnokokat, mint a Jeopardy! (Ferrucci *et al.*, 2010), a póker (Bowling *et al.*, 2015; Moravčík *et al.*, 2017; Brown and Sandholm, 2019), a Dota 2(Fernandez and Mahlmann, 2018), a StarCraft II (Vinyals *et al.*, 2019) és a Quake III (Jaderberg *et al.*, 2019) videójátékok.

Képmegértés: Nem megelégedve azzal, hogy a kihívásokkal teli ImageNet objektumfelismerési feladaton az emberi pontosságot felülmúlták, a számítógépes látás kutatói a képfeliratozás nehezebb problémáját is felvállalták. Néhány lenyűgöző példa a következő: „egy motorkerékpárt vezető személy földúton", „két pizza a tűzhelyen lévő pizzasütő tetején", illetve „frizbi játékot játszó fiatalok egy csoportja" (Vinyals *et al.*, 2017b). A jelenlegi rendszerek azonban messze nem tökéletesek: a „sok étellel és itallal tele hűtőszekrény"-ről kiderült, hogy ez a parkolási tilalom jelzőtábla, amit sok apró matrica részlegesen eltakar.

Orvosi tudományok: Az MI-algoritmusok sok állapot diagnózisában a szakértő orvosokkal már megegyező képességűek, vagy azokat meg is haladják, különösen ha a diagnózis a képeken alapszik. A példák között szerepel: Alzheimer-kór (Ding *et al.*, 2018), áttétes rák (Liu *et al.*, 2017; Esteva *et al.*, 2017), szemészeti betegségek (Gulshan *et al.*, 2016) és bőrbetegségek (Liu *et al.*, 2019c). Egy szisztematikus áttekintés és metaanalízis (Liu *et al.*, 2019a) megállapította, hogy az MI-programok teljesítménye átlagosan az egészségügyi szakemberekkel egyenértékű volt. Az orvosi MI jelenlegi hangsúlya az ember és gép partneri kapcsolatának a megkönnyítésén van. Például a Lyna rendszer által a metasztázisos emlőrák diagnosztizálásában elért teljes

pontosság 99,6%, ami egy segítség nélküli emberi szakértő pontosságánál jobb, de a kombinációjuk még annál is jobb (Liu *et al.*, 2018; Steiner *et al.*, 2018).

Ezen technikák széles körű alkalmazását most nem a diagnosztikai pontosság korlátozza. Szükség van a klinikai eredmények javulásának bizonyítására, valamint az átláthatóság, az elfogultság hiányának és az adatvédelem biztosításának a felmutatására (Topol, 2019). 2017-ben, FDA csak két orvosi MI-alkalmazást hagyott jóvá, de ez a szám 2018-ra 12-re nőtt, és továbbra is emelkedik.

Időjárás: Tudósok egy csoportja elnyerte a 2018-as Gordon Bell-díjat, egy mélytanulási modellért, amely részletes információkat fedez fel a szélsőséges időjárási eseményekről, amik korábban éghajlati adatokban voltak elrejtve. Szuperszámítógépet használtak egy speciális GPU hardverrel, hogy az exaop (10^{18} művelet másodpercenként) szintet túllépjék. Ez volt az első gépi tanulási program, amihez ezt megtették (Kurth *et al.*, 2018). Rolnick *et al.* (2019) pedig egy 60 oldalas katalógust mutatott be, hogy a gépi tanulás hogyan használható fel az éghajlatváltozás kezelésére.

Ez csak néhány példa a ma már létező mesterséges intelligencia rendszerekre. Nem varázslat vagy sci-fi, hanem inkább a tudomány, a mérnöki munka és a matematika, amelyekhez ez a könyv bevezetést ad.

1.5. A mesterséges intelligencia kockázatai és előnyei

Francis Bacon filozófus, akinek érdeme a tudományos módszer kidolgozása, *The Wisdom of the Ancients* c. művében (1609) megjegyezte, hogy a „mechanikus tudományok használata nem egyértelmű, ugyanúgy kárt tehetnek, mint gyógyulást hozhatnak." Mivel az MI egyre fontosabb szerepet játszik a gazdasági, társadalmi, tudományos, orvosi, pénzügyi és katonai szférában, jól tennénk, ha mind a károkat, mind a gyógyíreket fontolóra vennénk. Modern szóhasználatban a kockázatokat és az előnyöket, amiket az MI hozhat. Az itt összefoglalt témákat részletesebben tárgyalja a 27. és a 28. fejezet.

Kezdjünk az előnyökkel: egyszerűen fogalmazva az egész civilizációnk az emberi intelligencia terméke. Ha lényegesen nagyobb gépi intelligenciához férnénk hozzá, ez az ambícióink felső határát jelentősen megemelné. Az MI és a robotika potenciálja, hogy az emberiséget az ismétlődő, szolgai munkától mentesítse, és drámai módon megnövelje a létrehozott áruk és szolgáltatások mennyiségét, előjele lehetne a béke és a bőség korszakának. Az MI képessége a tudományos kutatások felgyorsítására elősegíthetné a betegségek gyógyítását és megoldásokat kínálhatna az éghajlatváltozás problémáira és az erőforrások hiányára. Mint Demis Hassabis, a Google DeepMind vezérigazgatója javasolta: „Először oldjuk meg az MI-t, majd az MI-t használjuk fel minden más megoldásához."

Még mielőtt lehetőségünk nyílik az „MI-t megoldani", kitesszük magunkat – gondatlanságból vagy más okoknál fogva – az MI helytelen használatából adódó kockázatoknak. Néhány kockázat már nyilvánvaló, míg mások – a jelenlegi tendenciák alapján – igen valószínűek:

- *Halálos autonóm fegyverek*: Ezeket az Egyesült Nemzetek Szövetsége olyan fegyverekként határozza meg, amelyek emberi beavatkozás nélkül képesek megtalálni, kiválasztani és megsemmisíteni az emberi célpontokat. Az ilyen fegyverekkel kapcsolatos elsődleges probléma a *felskálázhatóságuk*: az emberi felügyeletre vonatko-

zó követelmény hiánya azt jelenti, hogy egy kis csoport tetszőlegesen nagyszámú fegyvert telepíthet bármilyen alkalmas felismerési kritérium által meghatározott emberi célpontok ellen. Az autonóm fegyverekhez szükséges technológiák hasonlóak az önjáró autókhoz szükséges technológiákhoz. Informális szakértői megbeszélések a halálos autonóm fegyverek kockázatáról 2014-ben az ENSZ-ben kezdődtek meg, majd 2017-ben, a Kormányzati Szakértők Fórumán eljutottak egy egyezmény megkötését megelőző fázisig.

- *Megfigyelés és rábeszélés*: Noha a telefonvonalak, a videokamera-hírcsatornák, az e-mailek és más üzenetküldési csatornák megfigyelése a biztonsági személyzet számára drága, unalmas és időnként jogi szempontból megkérdőjelezhető, az MI (beszédfelismerés, számítógépes látás és természetes nyelvi megértés) skálázható módon használható fel az egyének tömeges megfigyelésére és az érdekes tevékenységek felderítésére. Az információáramlás gépi tanulási technikákon alapuló személyre szabásával, a közösségi médiákon keresztül, a politikai magatartás bizonyos mértékben módosítható és ellenőrizhető – a 2016-ban kezdődő választások kapcsán merült fel ez az aggodalom.
- *Torzított döntéshozatal*: A gépi tanulási algoritmusok gondatlan vagy szándékosan rosszhiszemű alkalmazása olyan feladatoknál, mint például a feltételes szadadlábra helyezési vagy a bankikölcsön-kérelmek kiértékelése faji, nemi vagy egyéb védett kategóriák szempontjából torzított döntéseket eredményezhet. Gyakran maguk az adatok tükrözik a társadalomban elterjedt elfogultságot.
- *A foglalkoztatásra gyakorolt hatás*: A munkahelyeket megszüntető gépekkel kapcsolatos gondok évszázadosak. A dolog soha sem egyszerű: a gépek elvégeznek néhány olyan feladatot, amit az emberek egyébként elvégezhetnének, ám az embereket produktívabbá és ezáltal a munkapiacon alkalmazhatóbbá teszik, azontúl a vállalatok jövedelmezőbbek lesznek, és emiatt képesek magasabb fizetéseket adni. Bizonyos tevékenységek gazdaságilag életképessé válhatnak, amelyek egyébként nem lennének megvalósíthatók. Használatuk általában a jólét növekedését jelenti, de tendenciában a jólétet a munkaerőtől a tőke irányába mozdítja el, ami tovább súlyosbítja az egyenlőtlenség növekedését. A technológia korábbi fejlődése – például a mechanikus szövőszék feltalálása – a foglalkoztatásban súlyos zavarokat okozott, ám végül az emberek újfajta munkát találtak maguknak. Másrészt valószínű, hogy az MI ezeket az új típusú munkákat is képes lesz elvégezni. Ez a téma szerte a világon egyre nagyobb hangsúlyt kap a közgazdászok és kormányok körében.
- *Biztonságkritikus alkalmazások*: Az MI-technikák előrehaladtával egyre inkább használják azokat a nagy kockázatú, biztonsági szempontból kritikus alkalmazásokban, mint például az autók vezetése, vagy a városok vízellátásának kezelése. Már bekövetkeztek halálos balesetek, és rámutattak a gépi tanulási technikákkal kifejlesztett rendszerek formális ellenőrzésének és statisztikai kockázatelemzésének nehézségeire. Az MI területének olyan műszaki és etikai szabványokat kell kidolgoznia, amelyek legalább azokkal a műszaki és egészségügyi tudományágakkal mérhetők össze, ahol az emberek élete forog kockán.

2.4. Az intelligens ágensek felépítése

Az ágenseket eddig *viselkedésük* leírásán keresztül vizsgáltuk – azon cselekvés alapján, melyet egy adott érzékelési sorozat hatására végrehajtanak. Most neki kell esnünk a kemény diónak: azt megnézni, hogyan működnek belül. A mesterséges intelligencia feladata az **ágensprogram** megtervezése, amely egy ágensfüggvényt valósít meg – az érzékelések cselekvésekre való leképezését. Feltételezzük, hogy ez a program valamiféle fizikai érzékelőkkel és beavatkozókkal ellátott számítóeszközön fog futni – ezt **ágensarchitektúrá**nak nevezzük:

Ágensprogram

Ágensarchitektúra

$$\textit{ágens} = \textit{architektúra} + \textit{program}\,.$$

Nyilvánvalóan a kiválasztott programunknak olyannak kell lennie, ami megfelelő az architektúra számára. Ha a program például *Sétál* jellegű cselekvéseket fog javasolni, akkor jobb, ha az architektúrának vannak lábai. Az architektúra lehet egy közönséges számítógép, vagy lehet egy robotautó számos fedélzeti számítógéppel, kamerával és más érzékelőkkel. Általánosságban az architektúra a szenzoroktól érkező észleléseket elérhetővé teszi a program számára, futtatja a programot, és cselekvéseit, ahogy létrejönnek, továbbítja a beavatkozók felé. E könyv túlnyomó része az ágensprogramok tervezéséről szól, bár a 25. és a 26. fejezet kifejezetten az érzékelőkkel és a beavatkozókkal foglalkozik.

2.4.1. Ágensprogramok

E könyvben tervezett ágensprogramjaink mindannyian azonos vázzal rendelkeznek: bemenetként fogadják az aktuális érzékelést a szenzoroktól, és visszaküldenek egy cselekvést a beavatkozókhoz[6]. Vegyük észre a különbséget az ágensprogram, amely az aktuális érzékelést veszi bemenetként, és az ágensfüggvény között, amely a teljes érzékelési sorozatot fogadja. Az ágensprogram azért fogadja csak az aktuális érzékelést bemenetként, mivel nincs választása, csak az érkezik a környezettől. Ha az ágens cselekvéseinek a teljes érzékelési sorozattól függeniük kell, akkor az ágensnek emlékeznie kell az érzékelésekre.

Az ágensprogramot egy egyszerű pszeudokódú nyelv segítségével fogjuk leírni, amit a B. függelék definiál. (Az interneten elérhető kódtár valóságos programozási nyelveken írt megvalósításokat tartalmaz.) Például a 2.7. ábra egy egészen egyszerű ágenst mutat, amely nyomon követi az érzékelési sorozatot, és felhasználja egy cselekvéseket tartalmazó táblázat megcímzésére, hogy eldöntse, mit csináljon. A táblázat – aminek egy példája a 2.3. ábrán látható porszívóvilág – explicit módon reprezentálja az ágensfüggvényt, amit az ágensprogram testesít meg. Ahhoz, hogy ily módon egy racionális ágenst építsünk, nekünk, mint tervezőknek, olyan táblázatot kell megalkotnunk, amely minden lehetséges érzékelési sorozathoz a megfelelő cselekvést tartalmazza.

Példaértékű annak megfontolása, hogy az ágensek tervezésének táblázatvezérelt megközelítése miért van kudarcra ítélve. Legyen $\mathcal{P}$ a lehetséges érzékelések halmaza, és T az ágens élettartama (az általa vett érzékelések teljes száma). A megfeleltetési

[6] Vannak más lehetőségek is az ágensváz számára, például az ágensprogramokat készíthetjük **korutin**ok formájában, amelyek a környezethez képest aszinkron módon futnak. Minden ilyen korutin rendelkezik egy be- és egy kimeneti porttal, és egy olyan ciklust tartalmaz, amelyik beolvassa a bemeneti portról az érzékeléseket, és kiírja a cselekvéseket a kimeneti portra.

function Táblázatvezérelt-Ágens(*észlelés*) **returns** cselekvés
  persistent: *észlelések*, egy sorozat, kezdetben üres
    táblázat, az észlelések sorozatával címezhető táblázat, kezdetben teljesen kitöltött

  az *észlelés* beillesztése az *észlelések* végére
  cselekvés ← Lookup(*észlelések*, *táblázat*)
  return *cselekvés*

2.7. ábra A Táblázatvezérelt-Ágens program hívódik meg minden egyes új érzékelésre, és visszaad egy cselekvést minden alkalommal. A teljes érzékelési sorozatot megőrzi a memóriában.

tábla $\sum_{t=1}^{T} |\mathcal{P}|^t$ elemet fog tartalmazni. Vegyük példaként az automata taxit: az egyetlen kamerától érkező vizuális bemenet körülbelül 70 megabájt/másodperc sebességgel érkezik (30 kocka másodpercenként, 1080×720 képpont 24 bit színinformációval). Ez alapján egy óra vezetéshez olyan táblázatot kapunk, amely több mint $10^{600\,000\,000\,000}$ bejegyzést tartalmaz. Még a sakkhoz tartozó táblázat is – amely a való világ egy kicsiny, jól viselkedő részlete – (ahogy kiderül) legalább 10^{150} bejegyzést tartalmaz. Összehasonlításul: a megfigyelhető világegyetemben az atomok száma kevesebb, mint 10^{80}. Ezen táblázatok ijesztő mérete azt jelenti, hogy (a) ebben az univerzumban egyetlen fizikai ágensnek sem lesz elég helye a táblázat tárolására, (b) a tervezőnek nem lenne elég ideje a táblázat elkészítéséhez, (c) egyetlen ágens sem lenne képes a táblázat helyes bejegyzéseit megtanulni saját tapasztalatából.

Mindezek ellenére, a Táblázatvezérelt-Ágens *megteszi* azt, amit akarunk, feltételezve, hogy helyesen van kitöltve: megvalósítja a kívánt ágensfüggvényt. *Az MI alapvető kihívása, hogy hogyan írjunk olyan programot, amely – a lehetőségek határain belül – egy hatalmas táblázat helyett kis méretű programkóddal produkál racionális viselkedést.*

Sok példánk van arra, hogy ez más területeken megvalósítható: például az 1970-es évek előtt a mérnökök és az iskolás gyerekek által használt hatalmas négyzetgyök-táblázatokat ma már egy elektronikus kalkulátorokon futó ötsoros program váltotta fel, amely a Newton-módszert alkalmazza. A kérdés az, hogy megteheti-e az MI azt az általános intelligens viselkedésre, amit Newton megtett a négyzetgyökökre? Abban hiszünk, hogy a válasz igen.

E fejezet hátralévő részében négy alapvető ágensprogramtípust vázolunk, amely megtestesíti a szinte minden intelligens rendszer mögött meghúzódó alapelveket:

- egyszerű reflexszerű ágensek;
- modellalapú reflexszerű ágensek;
- célorientált ágensek; és
- hasznosságalapú ágensek.

Minden egyes ágensprogram adott komponenseket meghatározott módon kombinált a cselekvések előállítására. A 2.4.6. alfejezet általános fogalmakkal elmagyarázza, hogyan alakíthatók ezek a programok *tanuló ágensekké*, amelyek javítani tudják komponenseik teljesítményét, és így jobb cselekvéseket generálnak. Végezetül, a 2.4.7. alfejezet

többféle módot is leír a komponensek ágensen belüli reprezentálására. Ez a választék egy lényeges rendezőelv ezen a szakterületen és a könyvben is.

2.4.2. Egyszerű reflexszerű ágensek

Egyszerű reflexszerű ágens

A legegyszerűbb fajtájú ágens az **egyszerű reflexszerű ágens**. Ezek az ágensek az *aktuális* érzékelés alapján választják ki a cselekvéseket, figyelmen kívül hagyva az érzékelési sorozat többi részét. Például a porszívóágens, amelynek ágensfüggvényét a 2.3. ábra táblázatos formában mutatja, egy egyszerű reflexszerű ágens, mivel döntései csak a jelenlegi helyszínen és azon alapulnak, hogy ott van-e piszok. A 2.8. ábra mutatja ezen ágens programját.

function Reflexszerű-Porszívó-Ágens([*hely*,*státusz*]) **returns** cselekvés

  if *státusz* = *Piszkos* **then return** *Felszívás*
  else if *hely* = *A* **then return** *Jobbra*
  else if *hely* = *B* **then return** *Balra*

2.8. ábra Egyszerű reflexszerű ágens programja a kétállapotú porszívó környezetben. A 2.3. ábrán megadott ágensfüggvényt valósítja meg.

Vegyük észre, hogy a porszívóágens-program igen kicsi a hozzá tartozó táblázathoz képest. A legnyilvánvalóbb méretcsökkentés az érzékelési történet figyelmen kívül hagyásából következik, amely a lehetőségek számát 4^T-ről 4-re csökkenti. Egy további, kisebb csökkenés abból a tényből fakad, hogy amikor az aktuális négyzet piszkos, a cselekvés nem függ a helyszíntől. Bár az ágensprogramot ha-akkor-egyébként utasítások segítségével írtuk meg, elég egyszerű ahhoz, hogy akár egy logikai áramkör segítségével is meg lehet valósítani.

Egyszerű reflexszerű viselkedések bonyolultabb környezetekben is előfordulhatnak. Képzeljük magunkat az automata taxisofőr helyébe! Ha az előttünk haladó autó fékez, és a féklámpái kigyulladnak, akkor észre kell vennünk, és el kell kezdenünk fékezni. Más szavakkal, valamilyen feldolgozás zajlik a képi bemeneten, hogy megállapítsunk egy feltételt, amit „az előző autó fékez”-nek hívunk, azután ez kivált valamilyen kialakított kapcsolatot az ágens programjában a „kezdj fékezni” cselekvéshez. Ezt a kapcsolatot **feltétel–cselekvés szabály**nak hívjuk[7], és így írjuk le:

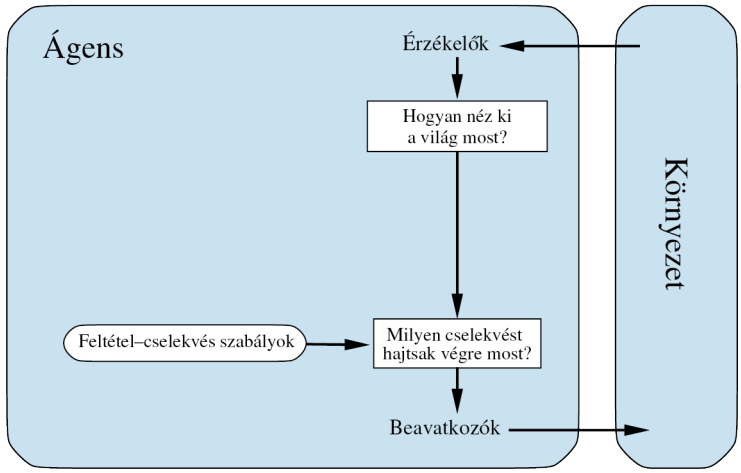
Feltétel–cselekvés szabály

ha *az-előző-autó-fékez* **akkor** *kezdj-fékezni*.

Az embereknek sok hasonló kapcsolatuk van, egy részük tanult válasz (ami a vezetést illeti), míg más részük feltétlen reflex (mint például a pislogás, amikor valami megközelíti a szemet). A könyv folyamán több különböző módot is látni fogunk arra, hogy ilyen kapcsolatok hogyan tanulhatók és valósíthatók meg.

A 2.8. ábra programja egy adott porszívókörnyezetre specifikus. Egy általánosabb és rugalmasabb megközelítés az, hogy először egy általános célú értelmezőt építünk feltétel–cselekvés szabályokra, majd szabályhalmazokat hozunk létre specifikus feladatkörnyezetek számára. A 2.9. ábra ezen általános program struktúráját adja meg

[7] Nevezik még **szituáció–cselekvés**, **produkciós**, vagy **ha–akkor szabály**nak is.



2.9. ábra Egyszerű reflexszerű ágens sematikus diagramja. Négyszögekkel jelezzük az ágens döntéshozatali folyamatának aktuális belső állapotát, és oválisokkal a folyamatban felhasznált háttérinformációkat.

function Egyszerű-Reflexszerű-Ágens(*észlelés*) **returns** cselekvés
  persistent: *szabályok*, feltétel-cselekvés szabályok halmaza

  állapot ← Bemenet-Értelmezése(*észlelés*)
  szabály ← Szabályillesztés(*állapot*, *szabályok*)
  cselekvés ← *szabály*.Cselekvés
  return *cselekvés*

2.10. ábra Egyszerű reflexszerű ágens. Egy szabály alapján működik, amelynek feltétel része illeszkedik a belső állapotra, amit az érzékelés határoz meg.

sematikus formában, megmutatva, hogy a feltétel–cselekvés szabályok hogyan teszik lehetővé az ágens számára az érzékelések és cselekvések összekötését. Ne aggódjon, ha ez túl egyszerűnek tűnik: hamarosan sokkal érdekesebbé válik.

A 2.9. ábra ágensprogramja a 2.10. ábrán látható. A Bemenet-Értelmezése függvény a bemenetből állítja elő az aktuális állapot absztrakt leírását, míg a Szabályillesztés függvény az első olyan szabályt adja vissza a szabályok halmazából, amely illeszkedik az adott állapot leírására. Vegyük észre, hogy a „szabály” és az „illeszkedés” fogalmakkal történő leírás tisztán koncepcionális, egyedi megvalósítások akár egy logikai áramkört megvalósító logikai kapukat tartalmazó halmazból is állhatnak. Alternatívaként egy „neurális” áramkör is használható, amelyben a logikai kapukat mesterséges neurális hálózatok nemlineáris elemeivel cseréljük fel (lásd a 21. fejezetben).

Az egyszerű reflexszerű ágenseknek megvan az az értékelendő tulajdonságuk, hogy egyszerűek, ugyanakkor igen korlátozott intelligenciájúnak bizonyulnak. A 2.10. ábrán látható ágens csak akkor fog működni, *ha a helyes döntés kizárólag az aktuális érzékelés alapján meghozható – azaz akkor, ha a környezet teljesen megfigyelhető.*

5.2.1. A minimax keresés algoritmus

Most, hogy ki tudjuk számítani a MINIMAX(s) értéket, ezt át tudjuk alakítani keresési algoritmussá, amely megtalálja a legjobb lépést MAX számára: ehhez ki kell próbálni az összes cselekvést, majd ezek közül azt kell kiválasztani, amelynek az eredményállapota a legnagyobb MINIMAX értékkel rendelkezik. Az 5.3. ábra mutatja be az algoritmust. Ez egy rekurzív algoritmus, amely lefelé halad a fában egészen a levélcsomópontokig, majd a minimax értékeket a fa mentén **visszafelé terjeszti**, ahogy a rekurzió visszalép. Az 5.2. ábrán például az algoritmus először rekurzív módon leereszkedik a három bal alsó csomóponthoz, majd a HASZNOSSÁG függvénnyel kiszámítja ezek értékeit, melyek rendre 3, 12 és 8. Ezután az algoritmus veszi ezen értékek minimumát, azaz 3-at, és ezt adja vissza a B csomópont visszaterjesztett értékeként. Hasonló folyamat eredményeként állnak elő a további visszaterjesztett értékek: 2 a C és 2 a D csomópont számára. Végül vesszük a 3, 2 és 2 értékek maximumát, hogy megkapjuk a gyökércsomópontba visszaterjesztett 3-as értéket.

```
function MINIMAX-KERESÉS(játék, állapot) returns egy cselekvés
    játékos ← játék.KI-LÉP(állapot)
    érték, lépés ← MAX-ÉRTÉK(játék, állapot)
    return lépés

function MAX-ÉRTÉK(játék, állapot) returns egy (hasznosság, lépés) pár
    if játék.VÉGÁLLAPOT-TESZT(állapot) then
        return játék.HASZNOSSÁG(állapot, játékos), null
    v ← −∞
    for each a in játék.CSELEKVÉSEK(állapot) do
        v2, a2 ← MIN-ÉRTÉK(játék, játék.EREDMÉNY(állapot, a))
        if v2 > v then
            v, lépés ← v2, a
    return v, lépés

function MIN-ÉRTÉK(játék, állapot) returns egy (hasznosság, lépés) pár
    if játék.VÉGÁLLAPOT-TESZT(állapot) then
        return játék.HASZNOSSÁG(állapot, játékos), null
    v ← +∞
    for each a in játék.CSELEKVÉSEK(állapot) do
        v2, a2 ← MAX-ÉRTÉK(játék, játék.EREDMÉNY(állapot, a))
        if v2 < v then
            v, lépés ← v2, a
    return v, lépés
```

5.3. ábra Egy algoritmus az optimális lépés meghatározására a minimax érték felhasználásával. Az optimális lépés az, amelyik a legnagyobb hasznosságú végállapothoz vezet, feltételezve azt, hogy az ellenfél szándéka a hasznosság minimalizálása. A MAX-ÉRTÉK és MIN-ÉRTÉK függvények végigmennek a teljes játékfán, le egészen a levélcsomópontokig, hogy meghatározzák a csomópont felfelé terjesztett értékét és az odajutáshoz szükséges lépést.

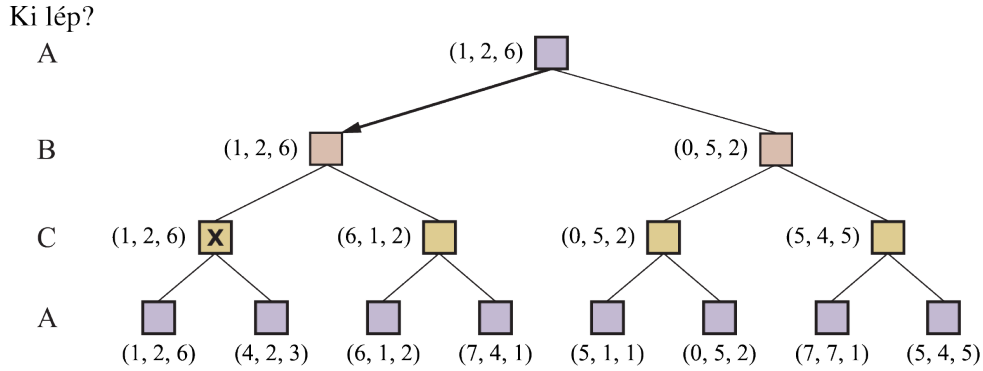
A minimax algoritmus a játékfa teljes mélységi feltárását végzi. Ha a fa maximális mélysége m és minden csomópontban b legális lépés létezik, akkor a minimax algoritmus időkomplexitása $O(b^m)$. A tárkomplexitása $O(bm)$, ha az algoritmus az összes cselekvést egyszerre számítja ki, és $O(m)$, ha a cselekvéseket egyenként generálja (lásd a 93. oldalon). Bonyolultabb játékok esetén ez az exponenciális időkomplexitás az algoritmust teljesen haszontalanná teszi. A sakk esetén például az elágazási tényező körülbelül 35, egy átlagos játék pedig körülbelül 80 lépésváltásból áll, és $35^{80} \approx 10^{123}$ állapot keresése nem kivitelezhető. A Minimax algoritmus azonban egy jó alap a játékok matematikai elemzéséhez. A minimax elemzés különféle módokon történő közelítésével pedig a gyakorlatban jobban használható algoritmusokat tudunk származtatni.

5.2.2. Optimális döntések többszemélyes játékokban

Számos népszerű játékban kettőnél több játékos is részt vehet. Vizsgáljuk meg, hogy a minimax ötletet hogyan terjeszthetjük ki többszemélyes játékokra. Technikai szempontból a dolog egyszerű, azonban felmerül néhány új, érdekes koncepcionális kérdés.

Először is egy csomópontokhoz rendelt egyetlen értéket egy értékvektorral kell felváltani. Például egy háromszemélyes játékban, amelyben három játékos: A, B és C vesz részt, minden csomóponthoz egy $\langle v_A, v_B, v_C \rangle$ vektort társítunk. Végállapotok esetén ez a vektor megadja az állapot hasznosságát minden játékos szemszögéből. (Kétszemélyes zéróösszegű játékokban a kételemű vektort egy értékre le lehet egyszerűsíteni, mert az értékek mindig ellentétesek.) Ezt úgy lehet legegyszerűbben implementálni, hogy a Hasznosság függvény a hasznosságok vektorát adja vissza.

Most tekintsük a nem-végállapotokat. Nézzük meg az 5.4. ábrán látható játékfában az X-el jelzett csomópontot. Ebben az állapotban a C játékos dönti el, hogy mit tegyen. Egyik választása a $\langle v_A = 1, v_B = 2, v_C = 6 \rangle$, míg a másik a $\langle v_A = 4, v_B = 2, v_C = 3 \rangle$ hasznosságvektorokkal rendelkező végállapothoz vezet. Mivel 6 több, mint 3, C-nek az első lépést kellene választania. Ez azt jelenti, hogy ha a játék az X csomópontot eléri, a következő lépés a $\langle v_A = 1, v_B = 2, v_C = 6 \rangle$ hasznosságú végállapothoz fog vezetni. X visszaterjesztett értéke így ez a vektor. Általánosságban egy n csomópont visszaterjesztett értéke annak a követő állapotnak a hasznosságvektora, amelynek az n csomópontnál választó játékos szempontjából legnagyobb az értéke.



5.4. ábra Három játékos (A, B, C) játékfája az első három lépésváltás esetén. Minden csomópontot megcímkéztünk az összes játékos szemszögéből számított értékkel. A legjobb lépést bejelöltük a gyökérnél.

sikeres ágens tervezésekor kombinálni kell a deklaratív és a procedurális elemeket, és hogy a deklaratív tudás gyakran lefordítható a deklaratív leírásnál hatékonyabb procedurális programmá.

Megadhatunk a tudásbázisú ágens számára egy olyan mechanizmust, amely képessé teszi őt arra, hogy saját maga tanuljon. Ez a mechanizmus, amit a 19. fejezetben tárgyalunk, általános, a környezetre vonatkozó tudást hoz létre érzetek sorozatának felhasználásával. Ily módon a tanuló ágens teljesen autonómmá válhat.

7.2. A wumpus világ

Wumpus világ

Ebben a fejezetben bemutatunk egy környezetet, amelyben a tudásbázisú ágens meg tudja mutatni a képességeit. A **wumpus világ** egy barlang, amely szobákból és az ezeket összekötő átjárókból áll. A wumpus egy szörnyeteg, aki a barlangban lapul valahol, és mindenkit megesz, aki a szobájába lép. Az ágens le tudja lőni a wumpust, de csak egyetlen nyila van ehhez. Néhány szoba feneketlen csapdát tartalmaz, amely mindenkit csapdába ejt, aki belép a szobába (kivéve a wumpust, aki túl nagy ahhoz, hogy beleessen). A wumpus környezetében az egyetlen csábító lehetőség, hogy egy halom aranyat lehet találni. Habár a wumpus világ meglehetősen unalmas a modern számítógépes játékokhoz képest, kiváló tesztkörnyezet az intelligens ágensek számára.

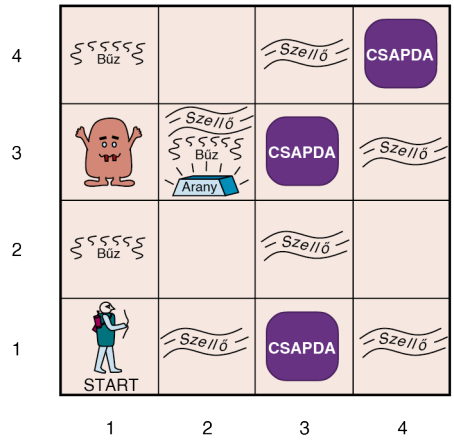
A wumpus világra a 7.2. ábrán látható egy példa. A példakörnyezet pontos definícióját, ahogy a 2.3. alfejezetben javasoltuk, a TKBÉ[1] leírással adjuk meg:

- **Teljesítménymérték**: +1000 az arany felvétele, −1000 a csapdába esés vagy ha a wumpus felfal, −1 minden végrehajtott cselekvés, −10 a nyíl használata. A játéknak akkor van vége, ha az ágens meghal vagy ha az ágens kijut a barlangból.
- **Környezet**: Egy szobákból álló 4×4-es háló. Az ágens mindig az [1,1]-gyel jelölt négyzetből indul, arccal jobbra nézve. Az arany és a wumpus elhelyezkedése véletlenszerűen, a kiinduló négyzeten kívüli négyzetek közül egyenletes eloszlás szerint van megválasztva. Ezen kívül még bármely, a kiinduló négyzeten kívüli négyzet 0,2 valószínűséggel lehet csapda.
- **Cselekvések (beavatkozók)**: Az ágens mozoghat *előre*, fordulhat *balra* 90°-kal, vagy fordulhat *jobbra* 90°-kal. Az ágens szörnyűséges halált hal, ha belép egy négyzetbe, ahol csapda van vagy egy élő wumpus található. (Biztonságos, habár meglehetősen rossz illatú egy olyan négyzetbe belépni, amelyben egy halott wumpus van.) Az előrelépésnek nincs hatása, ha egy fal van az ágens előtt. A *Megragad* cselekvést lehet arra használni, hogy az ágens felvegyen egy tárgyat, amely vele azonos szobában van. A *Lövés* cselekvést lehet használni egy nyílnak abban az irányban történő kilövésére, amerre az ágens éppen áll. A nyíl addig repül, amíg el nem találja (és egyben meg nem öli) a wumpust, vagy falnak nem ütközik. Az ágensnek csak egy nyila van, így csak az első *Lövés* cselekvésnek van hatása. Végül a *Mászik* cselekvés használható a barlangból történő kijutásra, de csak az [1,1] szobában.

[1] A „Teljesítménymérték, Környezet, Beavatkozók, Érzékelők” kezdőbetűiből alkotott betűszó. Angol megfelelője: PEAS (Performance, Environment, Actuators, Sensors). *(A ford. megj.)*

- **Érzékelők**: Az ágensnek öt érzékelője van, mindegyik egyetlen bitnyi információt ad:
  - A wumpus tartózkodási helyével közvetlenül (nem átlósan) szomszédos négyzetekben az ágens *Bűz*t érez[2].
  - A csapdával közvetlenül szomszédos négyzetekben az ágens *Szellő*t érzékel.
  - A négyzetben, ahol az arany található, az ágens *Ragyogás*t érzékel.
  - Ha az ágens falnak ütközik, akkor *falnak Ütődés*t érzékel.
  - Ha a wumpust megölték, akkor egy elkeseredett *Sikoly*t hallat, amit a barlangban bárhol hallani lehet.

  Az érzeteket az ágens egy öt szimbólumot tartalmazó lista formájában kapja meg, például ha bűz és szellő van egy négyzetben, de nincs ütés, ragyogás vagy sikoly, akkor az ágens egy [*Bűz*, *Szellő*, *Nincs*, *Nincs*, *Nincs*] érzetet kap.



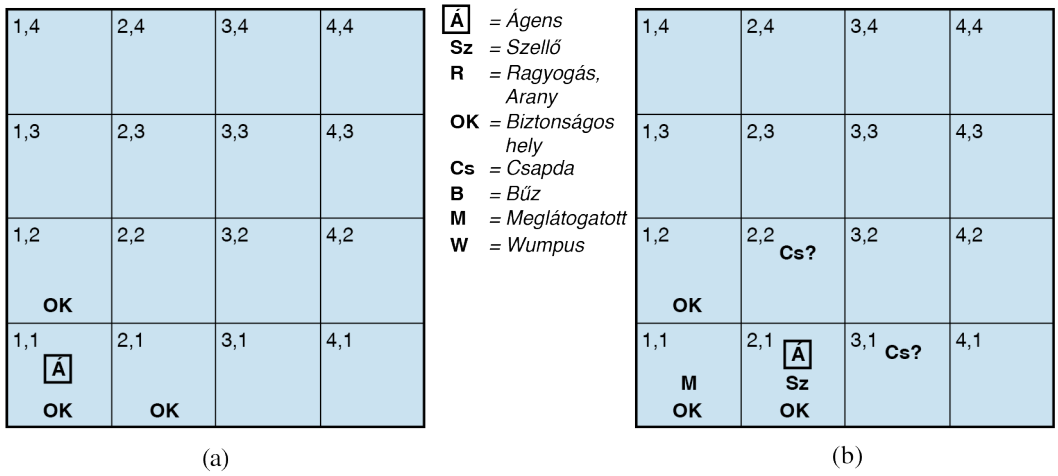
7.2. ábra Egy tipikus wumpus világ. Az ágens a bal alsó sarokban található, és kelet felé (jobbra) néz.

Mindezek mellett jellemezhetjük a wumpus környezetet a 2. fejezetben megadott különböző dimenziók mentén is. Egyértelmű, hogy ez determinisztikus, diszkrét, statikus és egyágenses környezet. (A wumpus szerencsére nem mozog.) Szekvenciális is a környezet, mivel a jutalom csak számos cselekvés után érkezhet. Továbbá részlegesen megfigyelhető, mivel az állapot néhány aspektusa nem közvetlenül érzékelhető: az ágens helye, a wumpus egészségi állapota és a nyíl rendelkezésre állása. A csapdák és a wumpus helye az állapot fel nem derített részének tekinthetők, és ilyenkor, mivel az állapotátmeneteket teljesen ismerjük, a csapdák helyének megtalálása kiegészíti az ágens tudását az állapotról. Egy másik megközelítésben viszont azt mondhatjuk, hogy magát az állapotátmenetek modelljét sem ismerjük, mert nem tudjuk, hogy melyik *Előre* cselekvés lesz végzetes. Így megfogalmazva a csapdák és a wumpus helyének felderítése az ágensnek az átmeneteket leíró modellről meglévő tudását egészíti ki.

[2] Feltehetően abban a szobában is, ahol a wumpus van, érezhető bűz, de bárkit, aki oda belép, felfal a szörny, mielőtt az illető bármit is érzékelne.

Az alapvető nehézség az ágens számára az, hogy kezdetben semmit sem tud a környezet konfigurációjáról, e tudatlanság feloldásához logikai következtetésre van szükség. A wumpus világok legtöbb példányában az ágens számára lehetséges az arany biztonságos megszerzése. Néhány környezetben azonban az ágensnek választania kell, hogy hazamegy-e üres kézzel vagy kockázatot vállal, ami vagy az aranyhoz, vagy a halálhoz vezet. És a környezetek 21%-a teljesen tisztességtelen, mivel az arany egy csapdában vagy csapdákkal körülvett mezőben van.

Nézzünk meg egy tudásbázisú wumpus ágenst, ahogyan felfedezi a 7.2. ábrán látható környezetet. Egy informális tudásreprezentációs nyelvet fogunk használni szimbólumokat írva a négyzethálóba (amint ez a 7.3. és a 7.4. ábrákon látható).



7.3. ábra Az ágens első lépése a wumpus világban. (a) A kezdeti helyzet a [*Nincs*, *Nincs*, *Nincs*, *Nincs*, *Nincs*] érzékelése után. (b) Az első lépés után, a [2,1]-ben érzékelve a [*Nincs*, *Szellő*, *Nincs*, *Nincs*, *Nincs*]-et.

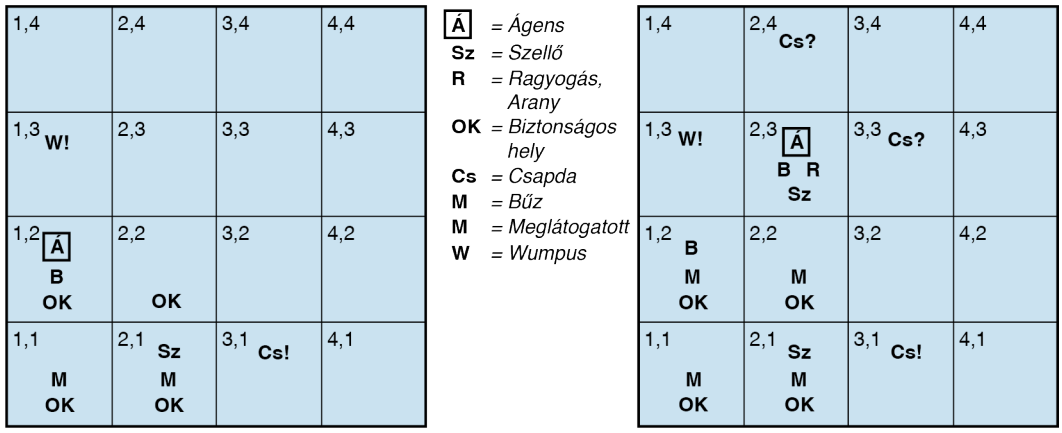
Az ágens kezdeti tudásbázisa a környezetet leíró szabályokat tartalmazza, az előzőekben leírtaknak megfelelően. Nevezetesen tudja, hogy az [1,1]-ben tartózkodik és hogy az [1,1] biztonságos hely. Ezt Á-val, illetve OK-val jelöljük az [1,1] szobában.

Az első érzékelés a [*Nincs*, *Nincs*, *Nincs*, *Nincs*, *Nincs*], amiből az ágens arra tud következtetni, hogy a szomszédos négyzetek, az [1,2] és a [2,1] biztonságosak, ezek is OK jelzésűek. A 7.3. ábra (a) része mutatja az ágens tudásának állapotát ezen a ponton.

Egy óvatos ágens csak olyan négyzetbe lép, amelyről tudja, hogy OK. Feltételezzük, hogy az ágens úgy dönt, hogy a [2,1]-be megy. Az ágens detektálja a szellőt a [2,1]-ben, tehát valamelyik szomszédos négyzetben csapdának kell lennie. A csapda nem lehet az [1,1]-ben a játék szabályai szerint, így csapdának kell lennie a [2,2]-ben vagy a [3,1]-ben vagy mindkettőben. A Cs? jelölés egy lehetséges csapdát jelez a mezőkben a 7.3. ábra (b) részén. Ezen a ponton csak egy olyan ismert négyzet van, ami OK, és amit az ágens még nem látogatott meg. Így a megfontolt ágens visszafordul, visszamegy az [1,1]-be és az [1,2]-be halad tovább.

Az ágens bűzt érzékel az [1,2]-ben, ami a 7.4. ábra (a) részén látható helyzetet eredményezi. A bűz az [1,2]-ben azt jelenti, hogy a wumpusnak a közelben kell lennie. De a wumpus a játék szabályai szerint nem lehet az [1,1]-ben és nem lehet a [2,2]-ben sem (mert akkor az ágens érezte volna a bűzt, amikor a [2,1]-ben járt). Így az ágens kikövetkeztetheti, hogy a wumpus az [1,3]-ban van. A W! jelölés ezt mutatja. Még érdekesebb, hogy a szellő érzet hiánya az [1,2]-ben azt jelenti, hogy nincs csapda a [2,2]-ben. De mi már kikövetkeztettük, hogy vagy a [2,2]-ben, vagy a [3,1]-ben van egy csapda, ami tehát azt jelenti, hogy a csapdának a [3,1]-ben kell lennie. Ez egy viszonylag nehéz következtetés, mivel különböző időpontokban és különböző helyeken gyűjtött tudást használ fel, és egy érzet hiányára támaszkodva végez el egy fontos lépést.

Az ágens így bebizonyította önmaga számára, hogy nincs se csapda, se wumpus a [2,2]-ben, így a mozgás ebbe a négyzetbe OK. Nem mutatjuk be az ágens tudását a [2,2]-ben, csak feltételezzük, hogy fordul és átlép a [2,3]-ba, ami a 7.4. ábra (b) részén látható. A [2,3]-ban az ágens detektálja a ragyogást, így megragadja az aranyat és visszatér a kiindulási négyzethez.



(a) (b)

7.4. ábra Két későbbi helyzet az ágens előrehaladása során. (a) A harmadik lépést követően, miután visszalépett [1,1]-be, majd onnan [1,2]-be, az ágens [*Bűz*, *Nincs*, *Nincs*, *Nincs*, *Nincs*]-et érzékelt. (b) Az ötödik lépés, vagyis a [2,2]-be, majd [2,3]-ba való lépés után, a [*Bűz*, *Szellő*, *Ragyogás*, *Nincs*, *Nincs*] érzékelésekor.

Bármely esetben, amikor az ágens következtetéseket von le a rendelkezésre álló információkból, a következmény *garantáltan* helyes lesz, ha a rendelkezésre álló információk helyesek. Ez alapvető jellegzetessége a logikai következtetéseknek. A fejezet hátralevő részében megmutatjuk, hogyan építhetünk olyan logikai ágenseket, amelyek képesek reprezentálni a szükséges információkat és következtetéseket vonnak le, ahogy azt az eddigi fejezetekben leírtuk.

7.3. A logika

Ez az alfejezet áttekintést nyújt a logikai reprezentáció és következtetés alapvető fogalmairól. A logika szép tulajdonságai függetlenek az egyes logikai kifejezésmódoktól. Ezért a kifejezések technikai részleteinek bemutatását a következő fejezetre halasztjuk, ehelyett inkább az aritmetika területének ismerős példáját használjuk.

Szintaxis

A 7.1. alfejezetben említettük, hogy a tudásbázis mondatokból áll. Ezeket a mondatokat a reprezentációs nyelv **szintaxis**a szerint fejezzük ki, amely specifikálja az összes jól formált, nyelvtanilag helyes mondatot. A szintaxis fogalma elég tiszta a szokásos aritmetikai műveleteknél: „x+y=4” egy jól formált mondat, míg az „x4y+ =” nem az.

Szemantika
Igazság
Lehetséges világ

A logikának a mondatok **szemantiká**ját, avagy a jelentését is definiálnia kell. A szemantika definiálja minden mondat **igazság**át minden egyes **lehetséges világ**ra vonatkozóan. Például egy szokásos aritmetikához választott szemantika meghatározza, hogy az „$x + y = 4$” mondat igaz abban a világban, ahol x értéke 2 és y értéke 2, de hamis abban a világban, ahol x értéke 1 és y értéke is 1. A standard logikákban minden mondat vagy igaz, vagy hamis minden egyes lehetséges világban, és nem lehet „valahol az igaz és hamis között”[3].

Modell

Amikor szükséges, hogy pontosak legyünk, a **modell** kifejezést fogjuk használni a „lehetséges világ” helyén. Miután a lehetséges világokat úgy képzelhetjük el, mint (potenciálisan) valós környezeteket, amelyekben az ágens ott lehet vagy nem lehet ott, a modellek olyan matematikai absztrakciók, amelyek csak rögzítik az igazság vagy hamisság értékét minden releváns mondatnak. Például, tegyük fel, hogy x és y a férfiak és nők száma, akik egy kártyaasztal körül ülnek és bridzset játszanak, és az $x + y = 4$ mondat igaz, amikor négyen vannak összesen. Formálisan, a lehetséges modellek nem mások, mint minden lehetséges nemnegatív egész szám hozzárendelése az x és y változókhoz. Minden ilyen hozzárendelés bármely olyan aritmetikai mondat igazságát rögzíti, amely az x és y változókat tartalmazza. Ha egy α mondat igaz egy m modellben, akkor azt mondjuk, hogy m **kielégíti** α-t, vagy néha azt, hogy m **modell**je α-nak. Az $M(\alpha)$ jelölést használjuk α összes modelljének halmazára.

Kielégítés

Vonzat

Most, hogy van egy képünk az igazság fogalmáról, tudunk beszélni a logikai következtetésről. Ennek része a mondatok közötti logikai **vonzat** reláció, annak kifejezése, hogy egy mondat logikusan következik egy másik mondatból. Matematikai jelöléssel ezt így írjuk:

$$\alpha \models \beta\,,$$

aminek az a jelentése, hogy az α mondat maga után vonzza a β mondatot. A vonzat formális definíciója a következő: $\alpha \models \beta$ akkor és csakis akkor, ha minden modellben, amelyben α igaz, β szintén igaz. Alkalmazva a most bevezetett jelölést, ezt így írhatjuk le:

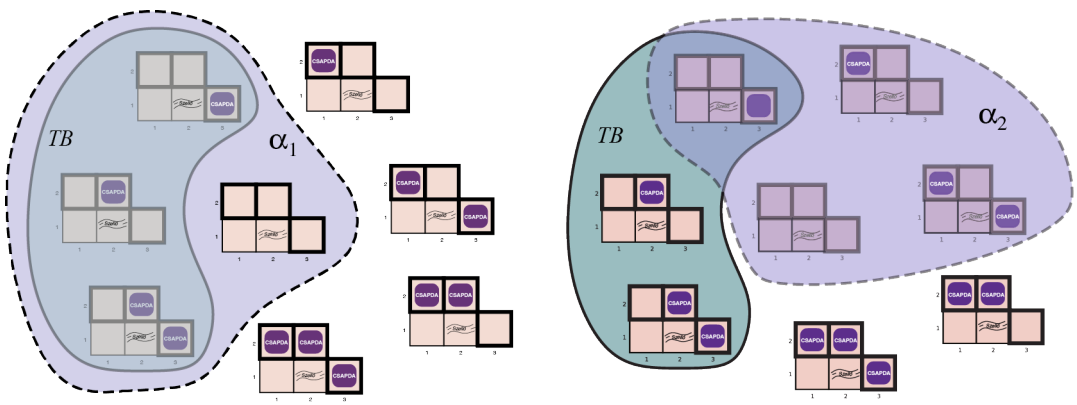
$$\alpha \models \beta \text{ akkor és csakis akkor, ha } M(\alpha) \subseteq M(\beta)\,.$$

(Vegyük észre, hogy a $\subseteq$ reláció iránya itt mit jelent: ha $\alpha \models \beta$, akkor α *erősebb* állítás, mint β, vagyis α több lehetséges világot zár ki.) A vonzat reláció ismerős az

[3] A **fuzzy logika**, amelyet a 13. fejezetben mutatunk be, megengedi az igazság mértékének kezelését.

aritmetikából: elfogadjuk azt a gondolatot, hogy az $x = 0$ mondat maga után vonzza az $xy = 0$ mondatot. Nyilvánvalóan bármely modellben, ahol x értéke 0, ott xy értéke is az (függetlenül y értékétől).

Alkalmazhatunk hasonló elemzést az előző részben bemutatott wumpus világbeli következtetési példára is. Tekintsük a 7.3. ábra (b) részén látható szituációt: az ágens nem észlelt semmit az [1,1]-ben és szellőt észlelt a [2,1]-ben. Ezek az érzetek, kombinálva az ágensnek a wumpus világ szabályaira vonatkozó tudásával, alkotják a *TB*-t. Az ágenst az érdekli, hogy vajon a szomszédos [1,2], [2,2], [3,1] négyzetek tartalmaznak-e csapdát. A három négyzet bármelyike tartalmazhat csapdát, vagy sem. Így (figyelmen kívül hagyva a világ más aspektusait) a példa esetében $2^3 = 8$ lehetséges modell létezik. Ez a 8 modell látható a 7.5. ábrán[4].



7.5. ábra A csapda jelenlétének lehetséges modelljei az [1,2], [2,2] és [3,1]-ben. A folytonos vonallal határolt halmaz tartalmazza azokat a lehetséges modelleket, melyek megfelelnek a tudásbázisban szereplő azon megfigyeléseknek, melyek szerint az [1,1]-ben semmi és a [2,1]-ben szellő érezhető. (a) A tudásbázis és α_1 (nincs csapda [1,2]-ben) modelljei szaggatott vonallal körbe rajzolva. (b) A tudásbázis és α_2 (nincs csapda [2,2]-ben) modelljei szaggatott vonallal körbe rajzolva.

A TB-t tekinthetjük a mondatok halmazának, vagy egyetlen mondatnak, amely minden egyes egyedi mondatot együttesen állít. A TB hamis azokban a modellekben, amelyek ellentmondanak annak, amit az ágens tud. Például a *TB* hamis minden modellben, ahol az [1,2] tartalmaz csapdát, mivel nincs szellő az [1,1]-ben. Valójában csak három olyan modell van, amelyben a *TB* igaz, ezeket a 7.5. ábra a modellek egy részhalmazaként mutatja. Most tekintsünk két lehetséges következményt:

α_1 = „Nincs csapda az [1,2]-ben.” α_2 = „Nincs csapda a [2,2]-ben.”

[4] Habár az ábra a modelleket részleges wumpus világokként jeleníti meg, ezek valójában nem mások, mint az *igaz* és *hamis* értékek hozzárendelései a „csapda van az [1,2]-ben” mondathoz. A modellek, matematikai értelemben, nem igénylik, hogy szörnyűséges illatú wumpusok legyenek benne.

11.1.2. Példa: A pótkerékprobléma

Tekintsük azt az esetet, amikor egy autón kereket kell cserélnünk (lásd a 11.2. ábrát). Pontosabban a célunk az, hogy egy jó pótkerék legyen felszerelve az autó tengelyére, ahol a kiinduló állapotban egy lapos kerék van felszerelve, míg a jó pótkerék a csomagtartóban található. Az egyszerűség kedvéért a mi feladatunk elég absztrakt, azaz nincsenek beragadt csavarok vagy egyéb más nehézségek. Csak négy cselekvés van: a pótkerék kivétele a csomagtartóból, a lapos kerék eltávolítása a tengelyről, a pótkerék felszerelése és végül az autó magára hagyása reggelig. Feltételezzük, hogy az autó egy igen rossz környéken áll, ahol az autó magára hagyása azt eredményezi, hogy reggelre eltűnnek a kerekek. Ekkor [*Eltávolít*(*Laposkerék*, *Tengely*), *Eltávolít*(*Pótkerék*, *Csomagtartó*), *Felszerel*(*Pótkerék*, *Tengely*)] a feladat megoldása.

$$
\begin{aligned}
&\mathit{Kiindulás}(\mathit{Kerék}(\mathit{Lapos}) \land \mathit{Kerék}(\mathit{Pót}) \land \mathit{Ott}(\mathit{Lapos}, \mathit{Tengely}) \\
&\qquad \land \mathit{Ott}(\mathit{Pót}, \mathit{Csomagtartó})) \\
&\mathit{Cél}(\mathit{Ott}(\mathit{Pót}, \mathit{Tengely})) \\
&\mathit{Cselekvés}(\mathit{Eltávolít}(\mathit{objektum}, \mathit{hely}), \\
&\quad \text{Precond:} \ \mathit{Ott}(\mathit{objektum}, \mathit{hely}) \\
&\quad \text{Effect:} \ \neg\mathit{Ott}(\mathit{objektum}, \mathit{hely}) \land \mathit{Ott}(\mathit{objektum}, \mathit{Föld})) \\
&\mathit{Cselekvés}(\mathit{Felszerel}(k, \mathit{Tengely}), \\
&\quad \text{Precond:} \ \mathit{Kerék}(k) \land \mathit{Ott}(k, \mathit{Föld}) \land \neg\mathit{Ott}(\mathit{Lapos}, \mathit{Tengely}) \\
&\qquad \land \neg\mathit{Ott}(\mathit{Pót}, \mathit{Tengely}) \\
&\quad \text{Effect:} \ \neg\mathit{Ott}(k, \mathit{Föld}) \land \mathit{Ott}(k, \mathit{Tengely})) \\
&\mathit{Cselekvés}(\mathit{OtthagyÉjszakára}, \\
&\quad \text{Precond:} \\
&\quad \text{Effect:} \ \neg\mathit{Ott}(\mathit{Pót}, \mathit{Föld}) \land \neg\mathit{Ott}(\mathit{Pót}, \mathit{Tengely}) \land \neg\mathit{Ott}(\mathit{Pót}, \mathit{Csomagtartó}) \\
&\qquad \land \neg\mathit{Ott}(\mathit{Lapos}, \mathit{Föld}) \land \neg\mathit{Ott}(\mathit{Lapos}, \mathit{Tengely}) \\
&\qquad \land \neg\mathit{Ott}(\mathit{Lapos}, \mathit{Csomagtartó}))
\end{aligned}
$$

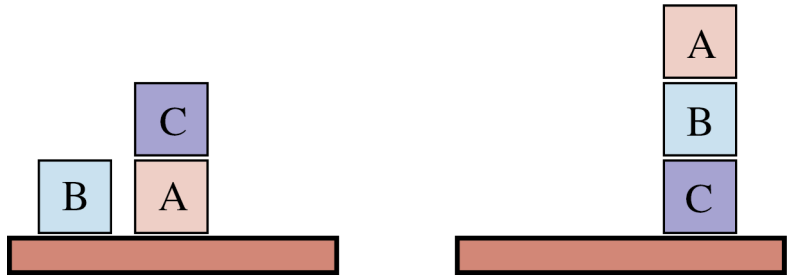
11.2. ábra Az egyszerű pótkerékprobléma.

11.1.3. Példa: A kockavilág

Az egyik leghíresebb tervkészítési terület a **kockavilág** probléma. A terület egy tetszőleges nagyságú asztallapon elhelyezett kockákból áll.[1] A kockákat egymásra rakhatjuk, de egy kockán közvetlenül mindig csak egyetlen másik helyezhető el. A kockákat egy robotkarral mozgathatjuk, amely fel tud venni egy kockát, majd azt vagy az asztalra, vagy egy másik kocka tetejére le tudja tenni. A robotkar egyszerre csak egy kockát tud felemelni, vagyis olyat nem, amelynek a tetején egy másik kocka van. Egy tipikus cél lehet például a B kockát a C tetejére, majd az A kockát a B tetejére helyezni (lásd a 11.3. ábrát).

A $\mathit{Rajta}(b, x)$ jelölést használjuk annak leírására, hogy a b kocka az x-en van, ahol az x egy másik kockát vagy az asztallapot jelenti. A $\mathit{Mozgat}(b, x, y)$ cselekvés a b kockát az x tetejéről az y tetejére mozgatja. A b kocka mozgatásának előfeltétele az, hogy

[1] A tervkészítés kutatásában használt kockavilág sokkal egyszerűbb, mint a Shrdlu verziója (lásd a 24. oldalon).



11.3. ábra A 11.4. ábrához tartozó kockavilág képe.

Kiindulás $(Rajta(A, Asztal) \wedge Rajta(B, Asztal) \wedge Rajta(C, A)$
$\wedge Kocka(A) \wedge Kocka(B) \wedge Kocka(C) \wedge \ddot{U}res(B) \wedge \ddot{U}res(C)$
$\wedge \ddot{U}res(Asztal))$
Cél$(Rajta(A, B) \wedge Rajta(B, C))$
Cselekvés$(Mozgat(b, x, y),$
PRECOND: $Rajta(b, x) \wedge \ddot{U}res(b) \wedge \ddot{U}res(y) \wedge Kocka(b) \wedge Kocka(y) \wedge$
$(b{\neq}x) \wedge (b{\neq}y) \wedge (x{\neq}y),$
EFFECT: $Rajta(b, y) \wedge \ddot{U}res(x) \wedge \neg Rajta(b, x) \wedge \neg \ddot{U}res(y))$
Cselekvés$(AsztalraTesz(b, x),$
PRECOND: $Rajta(b, x) \wedge \ddot{U}res(b) \wedge Kocka(b) \wedge Kocka(x),$
EFFECT: $Rajta(b, Asztal) \wedge \ddot{U}res(x) \wedge \neg Rajta(b, x))$

11.4. ábra A kockavilág tervkészítési problémája: egy három kockából álló torony építése. Az $[AsztalraTesz(C, A), Mozgat(B, Asztal, C), Mozgat(A, Asztal, B)]$ cselekvéssor egy lehetséges megoldás.

rajta ne legyen semmi. Ennek leírása az elsőrendű logikában a $\neg\exists\, x\ Rajta(x, b)$ vagy $\forall\, x\ \neg Rajta(x, b)$. Az alap PDDL a kvantorokat nem engedélyezi, ahelyett az $\ddot{U}res(x)$ predikátumot vezetjük be, ami akkor igaz, ha semmi nincs x-en. (A probléma teljes leírása a 11.4. ábrán látható.)

A *Mozgat* cselekvés a b kockát az x-ről az y-ra mozgatja, ha mind a b, mind pedig az y üres. A mozgatás után az x üres, de az y már nem. A *Mozgat* séma leírására az első próbálkozás a következő:

Cselekvés$(Mozgat(b, x, y),$
PRECOND:$Rajta(b, x) \wedge \ddot{U}res(b) \wedge \ddot{U}res(y),$
EFFECT:$Rajta(b, y) \wedge \ddot{U}res(x) \wedge \neg Rajta(b, x) \wedge \neg \ddot{U}res(y))\,.$

Sajnos ez a cselekvés nem kezeli jól az $\ddot{U}res$ predikátumot, ha az x vagy az y az asztalon van. Ha x az *Asztal*, a cselekvés következményei között szerepel az $\ddot{U}res(Asztal)$ is, de az asztalnak nem kell kiürülnie a mozgatás után, ha pedig $y = Asztal$, megjelenik az $\ddot{U}res(Asztal)$ előfeltétel, holott az asztalnak nem kell üresnek lenni ahhoz, hogy bármit is tehessünk rá.