Az Intelligent Tutoring System tudásbázis

FARKAS I. JÓZSEF

A magyar közoktatásban ma még csak kevesen ismerik az Intelligens Tutor Rendszer (a továbbiakban ITS – Intelligent Tutoring System) kifejezés teljes tartalmát, bár néhány pedagógus készít és használ oktatójellegű számítógépes programokat. Ugyanakkor az utóbbi évek szűkös anyagi körülményei ellenére, a számítógépek csaknem valamennyi hazai oktatási egységben megjelentek. Sajnos, a magas szintű technológia gyors beáramlása nem járt együtt a szemlélet hasonló ütemű formálódásával. Kevés kivételtől eltekintve gyakorló tanáraink többsége idegenkedik a számítógép használatától, nem ismervén kezelésének szakmai fogásait, a forgalomban megtalálható programok használatát, lehetőségeit. E témakörnek szinte egyáltalán nincs magyar nyelvű szakirodalma, a szűk körben forgó, többségében idegen nyelvű irodalom nem pótolja az ismeretek általános elterjedéséhez szükséges publicitást. Ezt a dolgozatot felhívásnak szánom a hasonló témakörben dolgozó kollegákhoz, egy tudományosan megalapozott kutató-fejlesztő publikációs polémiára, amelyben a közös gondolkodással egymást segítve próbáljuk behozni azt a lemaradást, ami a fejlett, oktatásszempontú számítástechnikai alkalmazókkal szemben a mai magyar szemléletet jellemzi.

Az ITS

Első lépésben meg kell határoznunk, hogy mit takar az ITS betűszó, mire lehet használni ezeket a számítógépes programokat, hogyan lehet beilleszteni a mai oktatási gyakorlatba, mint oktatásmódszertani eszközt.

Etien Wenger szerint az ITS olyan technikai eszköz, amellyel a tanárt mint szakértőt helyettesíthetjük, tehát egy szakterület teljes ismeretanyagát feldolgozásra kész állapotban, időhöz és helyhez való kötöttség nélkül a tanuló rendelkezésére bocsátjuk. Az ITS programok szerkezeti felépítésükkel a humán gondolkodás, problémamegoldás folyamatát szimulálva próbálják megtanítani azt, amit *tudásbázisukba* beépítettek. Az ITS nem egy egyszerű oktatóprogram! Arra nézvést, hogy *természetes nyelven* kommunikálhatunk a számítógéppel, hogy a program, futási időben, tehát az anyag feldolgozásával párhuzamosan, folyamatában értékeli a tanuló teljesítményét, ehhez igazodva adja meg a haladás ütemét, az elkövetett hibák okát, illetve eredetét képes felderíteni, annak javítására tud tanácsot adni, sőt önállóan generál feladatokat, Wenger kijelentését elfogadhatónak tartjuk. Azt azonban ki kell jelentenünk, hogy az élő tanítás komplex folyamatát ITS-el, a pillanatnyi felkészültségünk mellett nem tudjuk kiváltani. Összefoglalva tehát azt mondhatjuk, hogy *az ITS egy, az oktatásban használható számítógépes segédeszköz, amellyel egy pontosan meghatározott, jól körülírt problémakör részfeldolgozása, megtanulása közvetlen tanári közreműködés nélkül megoldható.*

Az ITS rövid története

Közismert tény, hogy a számítástechnika szülőhazája az Amerikai Egyesült Államok. Nem véletlen tehát, hogy itt kezdtek először foglalkozni a oktatás számítógépes támogatásának lehetőségeivel. Az 1960-as évek elején az Államok két egyetemén, a *Stanford University* és az *University of Illionis* intézeteiben indított fejlesztő munka eredményeként született a *PLATO Computer System* (1). Ez volt az első teljes, kifejezetten oktatási célra alkalmazott számítógépes programcsomag. Itt nem vesszük figyelembe a katonai célú, speciális esetekre fejlesztett rendszereket (például az 1957-ben már alkalmazott programot, az AN/MPQ-57 HIPIR-t – *High-Powered Illuminating Radar*), ugyanis ezek szervezési szempontjai rendkívül távol állnak a közoktatás szempontjaitól.

A PLATO kísérleti stádiumában egyszerre 30 munkahelyen (*terminálon*) heti 60 órában 450 tanulót oktattak fizikára, kémiára, matematikára, de alkalmazása során, idegen nyelv oktatásra is felhasználták. Például az University of Illionis-on egyszerre 80 terminálon futott a program. Jellemzésként azt mondhatjuk a rendszerről, hogy egy jól felszerelt tanítási segédeszköz, olyan keretrendszer (*shell*), amellyel tetszőleges bázisra épülő oktatási metodikát lehet kialakítani, a pedagógust felszabadítva a nagytömegű adminisztráció, rutinmuka időt rabló, monoton terhei alól.

Érdekességként megemlítjük, hogy 1970-ben az Amerikai Egyesült Államokban a középiskolák 31%-ában használtak számítógépet és ebből 13%-nyit, kizárólag oktatási célra alkalmaztak (2).

Az oktatóprogramok második generációját a TICCIT (*Time-shared Interactive Computer Controlled Information Television*) és a MACC (*Minnesota Educational Computing Consortium*) rendszerek képviselik. E két projekt volt a szisztematikus fejlesztési és kutatási munkák második állomása. Itt, az oktatórendszer egy előteszt eredményeire építkező modell alapján, már tervezte a haladás ütemét, kihasználta a más médiák nyújtotta lehetőségeket. Interaktív kapcsolatot tartott a tanár, illetve a tanuló és a technikai eszközök, úgymint filmvetítő, magnetofon, TV, video között. Az irányítást még folyamatosan ellenőrizte a pedagógus, csak az egyes részegységek (*unitok*) feldolgozási folyamata került a gép, pontosabban a program hatáskörébe. 1975-ben a TICCIT rendszer 128 tanulópad egyidejű munkája során párhuzamosan kezelt színes TV-ket, 10 db videolejátszót, hurokvetítőt és magnetofont (2 db. Nova800 Central 64K és 32K RAM-mal szervezett procasszorral, 175 Mb oszthatő háttértárral működött. Ára 4.000$/munkapad volt.) Ugyanekkor a CDC PLATO IV programcsomag egyszerre több száz munkapadon kiváló grafikai modulokkal, audió-, mikrofilmvetítő kapcsolatokkal, elvileg korlátlan memóriaháttérrel dolgozott.

A mai értelemben használt ITS megjelenését *Jaime Carbonell* nevéhez köthetjük, aki a BBN-nél (Bolt Beranch and Newman Inc. Cambridge, Massachusetts) dolgozta ki programját a SCHOLAR-t (3). Ez a program, Dél-Amerika (innen származik Carbonell) földrajzi neveit, fogalmait megtanító rendszer. Négy elkülönülő modulba szervezett program, amelynek szerkezete azóta is kiindulási alap a korszerű ITS-ek tervezésénél, fejlesztésénél. Ebben a rendszerben jelent meg először a természetes nyelvi kommunikáció (lokális résnyelvtanra épülő, kész, nyitott mondatok halmazával), az input kiértékelő tanuló-diagnózis modul. Itt különül el a tényanyag felsorolásjellegű halmazától (*a tudásbázistól*) a *szakértő*, a kezelést, szervezést, irányítást ellátó szabálybázis (*operátor bázis*), és itt találhatjuk meg első ízben az aktív kognitív modellen alapuló oktatási stratégia számítógépes alkalmazását.

A kiépítés, illetve a szabálybázis felállítása célspecifikus megfontolások eredménye. A későbbiekben ez a rész vált a legintenzívebben kutatott területté. A mennyiségi értékeléstől a minőségi szimulációig haladó fejlődést jól szemlélteti a SOPHIE I-III ICAI

(*Intelligent Computer Aided Instruction*) alakulása (4). Bár az első és a harmadik változat megjelenése között alig 5 év telt el, a rendszer fejlesztésének gyökerei 17 év távlatára nyúlnak vissza. Az adatbázis kezelését, természetes nyelvi *interface*-szel, az első változatban *John Seely Brown* és *Richard Burton*, az *University of California at Irvine* laboratóriumában kiépített, kérdés-válasz moduljával oldották meg, egy véges állapotú automatát szimuláló operátor bázissal vezérelve. Hátrányait (a tanuló modelljének hiánya, a pedagógiai aspektus minimális jelenléte, a hibakeresésen alapuló egyszerű értékelő rutin asszociációs problémái) a második, illetve a harmadik változat csaknem teljesen kiküszöböli (5,6).

A felsorolt programok és projektek forrásainak kiértékelésével nyomon követhetjük azt a fejlődési utat, amelyen az ITS az elmúlt 30 esztendő alatt keresztülment. A kronologikus és teljes történeti áttekintést kiváló monográfiákban találhatjuk meg (2,7,8). Ezekben a művekben egyértelmű választ kapunk arra is, hogy milyen mélységig érdemes a gépesített oktatási forma kínálta lehetőségekre támaszkodni, hol az a pont, ahol már semmiképpen sem váltható ki a pedagógus egyénisége.

Az ITS alkalmazásai

Mint azt már említettük, az ITS rendszerek használata bizonyos mértékig jól definiált területekre korlátozott. Ha azonban meggondoljuk, hogy polihisztort, az emberiség teljes történetét tekintve sem találunk túl sokat, a specializálódást természetes folyamatnak tekinthetjük. Ez a tény egyben segítség is a felhasználhatóság pontos leírásához. Vegyük besorolási szempontként a program adatbázisának kezelési, tárolási, adatlehívási módszerét és a kommunikációs rendszer felépítését. Azért választjuk e két paramétert, mert mint látni fogjuk, egy probléma megoldásakor a rendelkezésre álló ismeretek kezelése, azok alkalmazásának, illetve közlésének módszere egyértelműen utal az adatbázis tudományterületi forrására. A fentieket figyelembe véve, azokat a programcsomagokat, rendszereket, amelyeket mint *intelligens*, számítógépes munkaeszközt, egy adott területen használhatunk, (9) nyomán négy nagyobb kategóriába sorolhatjuk

1. Matematika, geometria, programnyelv-típusú feladatok oktatására, tudásszint mérésére alkalmazott programok. A szabálybázis itt axiomatikus szemantikájú metanyelvre épülő rendszer, hangsúlyosan szereplő hiba-felderítő-elemző (*troubleshooting*) modullal. A felhasználóval menüvezérelt online segítség, javasolt problémamegoldó rutinok tarják a kapcsolatot. Például a LISP-TUTOR (10).

2. Orvosi diagnosztikára, vegyészeti labor-, illetve üzemi szimulációra, elektronikai szettek, panelek tervezésére, szimulációs bemérésére használható programok. A kommunikációs szint egy irányított, természetesnyelvi interaktív blokk, vagy menüvezérelt ikon-rendszer, a leíró jellegű tényanyag kezelése IF-THEN, vagy T szabályon alapuló szabálybázis oldja meg. pl. MYCIN, GUIDON (11,12). Ma is működő orvosi diagnosztikus ITS.

3. Alkalmazói software-ek managment, marketing, irodarendszerek kezelésére, adminisztrációs, könyvelési feladatok ellátására. A kapcsolat „felhasználóbarát", ikonstrukturált kommunikációs nyelveken alapul, a ténybázis menüvezérelt irányítással, blokkstrukturált rutinrendszerrel szerveződik. pl. E-mail rendszerek, SIGHTPLAN (13).

4. Logikát fejlesztő, gondolkodási szokásokat befolyásoló, térben gondolkodtató feladatgenerátoros ITS-ek. Ebbe a kategóriába azok a programok tartoznak, amelyek a humán gondolkodás folyamatának szimulálásával tetszőleges probléma megoldható. A ténybázis szervezése az aktív kognitív tanulási modellen alapul, *szabad* természetes nyelvi kommunikációs kapcsolattal rendelkezik, esteleg *text-to-speach* beszédgenerátorral élő beszédet generál, inputanalízisen nyugvó automatikus tanulói-hipoté-

zis modellt felállító rutinja van, célgenerálással, hibafelderítővel, paralel segítővel (*online help funkció*) a tanulás folyamatát követni, korrigálni, irányítani tudja. pl. IDEBUGGY (14).

Hogy a saját tervezésű programjaink célorientált tudásbázisát miként kíséreljük meg feltölteni, hogyan kezeljük az adat-hozzáférések folyamatát, azok monitorra kerülését, azaz milyen jellegű operátorbázist építsük ki, ahhoz tekintsük át a lehetséges módszereket.

A tudásbázis fejlődése

Minden oktatási forma három, általánosítható és egymástól jól elkülöníthető szempont alapján bontja fel a megtanítandó tananyagot (15):

1. a hatékonyság szempontja,
2. a kapcsolódó készségek kialakításának szempontja,
3. a megismerés, az elsajátítás szempontja.

A korai ITS fejlesztői – bár még nem ismerhették a fenti metodikát – megpróbálták úgy szervezni programjaik működését, hogy az élő tanítás mozzanatainak szimulálása valóban befogadható legyen a tanuló számára és a fenti kritériumokat, mintegy ráérzéssel kielégítik. A természetes nyelvi kapcsolat megjelenésekor azonban hamar kiderült, hogy az input elemzése – szabad válaszok esetén – képtelenül nagy szabályrendszert, adatbázist, memóriát és műveleti sebességet igényel. Úgy tűnt, hogy a megoldást a szerkesztői nyelv, az *„Authoring Language"* jelenti, amely speciális problémákhoz, speciális nyelvet és szemantikát ad. Mint kapcsolódó kutatási terület, a *számítógépes nyelvészet*, nyelvi modellek sokaságát szolgáltatja, mégis, az ITS-ek fejlesztése során, alapvetően csupán két programnyelvet, a LISP, illetve a PROLOG nyelveket alkalmazzák. A tényleges megoldást, legalábbis a gyakorlat ezt mutatja, a *tudásbázis*t szervező, kezelő operátorbázis adja, melynek nyomait már Carbonell SCHOLAR-jában is felfedezhetjük.

A szemantikus háló

A SCHOLAR-ban, szemantikus hálóba szervezett csomópontokban tárolódnak a földrajzi nevek és fogalmak, ahol az egyes csomópontok (*node*-ok), a saját relátoraiknak megfelelő hierarchikus szintekre szerveződnek illetve felértékelődnek szuper-attribútumnak, szuper-fogalomnak, szuper-résznek. Például: Santiagó Dél-Amerikában van, s mivel Santiagó Chilében van, Chile Dél-Amerikában van. A relátorok természetesen kapcsolatot tudnak teremteni két csomópont között is, pl. Santiago és Buenos Aires két dél-amerikai főváros.

Általánosan megfogalmazva, a kidolgozott modell egy metanyelvet definiálva adja meg egy-egy inputfűzér linearizált szemantikus viszonyát. Minden, az adatbázisban szereplő fogalomhoz függvénnyel rendel egy szimbólumot, amit típusnak nevez, majd a szimbólumok halmazán parciális rendezéssel hozza létre a hierarchiát, amit típushálónak nevez el. A háló maximális eleme az univerzális típus, minimális eleme az abszurd típus. (A minimális elem az üres halmaz, nem a 0!) Egyedi jelölőket használva, amelyek lehetnek egyediek illetve generikusak, a módszer tetszőleges szemantikus hálókat képes generálni, ahol az f_i fogalomnak megfelelő predikátum a típus (f_i), az i argumentum pedig a háló i-edik eleméhez tartozó f_j fogalom azonosítója. Formalizálva: ([VÁROS:#12] => (STAT) => [FŐVÁROS] => (LOC) => [CHILE]) —>]x]y(VÁROS(#12))&STAT(#12,x)&FŐVÁROS(x)&LOC(x,y)&(y)).

Ahol a formula teste a predikátumok konjunkciója, kvantifikációja pedig $]x_1]x_2...]x_n$, a #12=egyedi jelölő. Például ha a [VÁROS] egyedi jelölője (*referense*) a #12, akkor az, az n darab lehetséges város közül a 12-diket, példánkban Santiagót azonosítja.

Ha összevetjük a humán gondolkodás és a *szemantikus network* (Carbonell tudásbázisa) működését, hamar kiderül, hogy csak egyszerű esetekben vonhatunk párhuzamot. Azonban néhány speciális körülmény definiálásával, nem túl nagy adatbázissal, a fenti algoritmus jó eredményeket képes szolgáltatni. A módszer használata sok esetben napjainkban is célravezető. Mivel azonban a gondolkodás folyamatának gépi modelljei ekkor még nem voltak képesek szimulálni a tényleges valóságot, célszerűnek tűnt a működés egyes mozzanatait kiragadva részfolyamatokat kidolgozni, és azokat az egyedi esetekre alkalmazni.

A szókrateszi metodika

A kutatók figyelme ekkor a dialóguson alapuló szókrateszi metodus felé fordult, amelyben a tanár sohasem exponál direkt módon egy tárgyra, hanem a tanuló intellektusának megfelelő kérdésekkel engedi, hogy a megfogalmazott válaszok alapján, felfedezésként élje meg az új ismeret elmélyülését. A metodika számítógépes adaptálására *Collins* dolgozott ki algoritmust, amely kiválóan szimulálja a rávezető kérdésekkel operáló szókrateszi metodikát (16). A teljes modellt, a 60 szabályon alapuló algoritmust, a WHY nevű, földrajzi ITS-ben alkalmazták először – a föld csapadékzónáinak elhelyezkedését, jellegét megtanító ITS (17). Példaként a 6. szabályt tesszük közzé Collinstól átvéve.

Rule 6: From a general rule for an insufficient factor.

IF the student gives an explanation of one or more factors that are not sufficient

THEN formulate a general rule asserting that the factors given are sufficiant and ask the student if the rule is true

REASON for *use*

to force the student to pay attention to other casual factors

EXAMPLE If the student gives water as the reason why they grow rice in Chine, ask him „Do you think any place with enought water can grow rice?"

A WHY-ban a tényanyagot egy leíró-hierarchikus, sztereotíp kapcsolatrendszer tárolja, amelyben a leírószekvenciák állandóak és az esetek közötti kapcsolatépítés könnyen vizsgálható. Ezt kihasználva, az általános kérdésgeneráló, az egyszerűtől indulva – a leírás függőségében – halad a bonyolult felé. (A kognitív „top down-bottom up" megismerési folyamathoz hasonlóan.) Az algoritmus működését egy próbafutás eredményével lehet illusztrálni:

WHY: Do you think the AMAZON jungle has heavy rainfall or light rainfall?
(Kérdés az egyik predikátum megközelítésére, egy eset kapcsán.)

Student: Heavy rainfall.

WHY: Why does the Amazon have heavy rainfall?
(Kérdés, a priori.)

Student: Because it's near the Atlantic.

WHY: Yes, the Amazon jungle is near the equatorial Atlantic. How does that affect rainfall in the Amazon jungle?
(Egy közbevetett kérdés az ok felderítésére.)

Student: The water in the current is warm.

.
.
.

WHY: Do you think that any place with mountains has heavy rainfall?

(A megsejtett szabály általánosítására tett kísérlet.)

Student: Yes.

WHY: Southern California has mountains. Why doesn't Southern California have heavy raifall?

(A megtanultnak feltételezett szabály helyes alkalmazásának próbája.)

A kiragadott példa nem csak a szókrateszi metodika egy kiváló szimulálását szemlélteti, hanem világosan megmutatja, hogy a természetesnyelvi interfész az ITS-ek Akhillesz-sarka.

A véges állapotú automata

A szókrateszi modell legnagyobb hátránya, hogy az aktuális cél meghatározásakor elveszítjük a tanuló más területről vett asszociációinak ellenőrzési lehetőségét (hiszen a tudásbázis speciális esetre épült ki), ami esetleg diktátumszerűvé teszi tanulás folyamatát. Ezt a problémát a METEOROLOGY ITS-ben megkísérelték feloldani. Ebben a programban a leíró adatbázist Véges-Állapotú-Automaták *(Finit State Automata – FSA)* halmaza kezeli. Ezek, a cimkézett csomópontokat irányított élekkel összekötő gráfok, amelyeknek definiált kezdő és végállapotaik vannak. Ha egy input hatására valamelyik automata végigfut egy élsorozaton és a végállapotába jut, akkor az input kiértékelése sikeres, ellenkező esetben valamilyen hibát hordoz. Mivel egy tudásbázis több kezdő és végállapottal bír, a rendszernek ismernie kell az azonos cimkéjű kezdőállapotok teljes halmazát, így szükség esetén valamennyi releváns automatát működésbe tudja hozni. Ha egy automata valamely élrendszeren végigfut, végrehajthat egy átmenetet a „lejárt" automata nevével cimkézve, így azon él végpontjából folytatja a működést, amelyből az előbb kilépett. Mivel az átmenetek az inputtól függenek, akkor lesz sikeres az átmenet, ha az input egy predikátummal megegyezik. Ezzel a program akciósorozatok sokaságát végezheti el, amit Woods „Bővített Átmenethálónak" nevezett el (18).

Röviden, az FSA-t tekinthetjük, egy tényekből és azok tulajdonságaiból, mint attributumokból álló csomópont-halmaz egyes elemei között kapcsolatot létesítő gépnek. A kötéseket logikai *és* (&) művelettel definiálja, s akkor sikeres a folyamat egy-egy lépése, ha az egyes tények és azok tulajdonságaik & kapcsolata a szabálybázisban rögzítetteknek nem mondanak ellent. Az így összegyűjtött csomópontok végeredménye az FSA egy eredményes végállapota, ami akkor helyes megoldás, ha a rögzített tudásbázis tartalmaz ilyen FSA végállapotot, illetve olyan végállapot sorozatot, amelyet az input generálta FSA-k befutottak.

Ebből a tudásbázis-kezelő módszerből fejlődött ki a „színezett" élekkel operáló AND-OR gráftechnika, ami a Mesterséges Intelligencia (*Artificial Intelligence* – AI) kedvelt eszköze a tények tárolási, lehívási, kezelési folyamatát vezérlő operátor-bázis kiépítésének.

A mentális modell

A fenti módszert továbbfejlesztve a SOPHIE ITS programban dolgoztak ki egy olyan rendszert, amelyben először bukkan fel az *aktív* kognitív tudásreprezentáció, mint a tanuló gondolkodását mérlegelő *mentális modell*. (5). A SOPHIE I. még nem komplex ITS, de mint automatikus labor-szimuláció, már természetesnyelvi interface-szel dolgozott. Inputértékelő rendszere hibakeresésen alapszik és azonnal (*real-time*) értékel, így lehetőség nyílik a hibás beidegződések kizárására. A tudásbázis felépítésekor különválasztották a készségalapú tudást (*procedural knowledge*) és a tényalapú tudást (*declarative knowledge*). Szervezésüket a már jól ismert szemantikus háló csomópontjain végigfutó FSA-k halmaza oldja meg, amelyben az operátorok négy szempont alapján keresik meg az összefüggő csomópontokat:

1. Válasz egy feltételezett eset kimenetelére. Például, mi történik, ha egy szabály valamelyik összetevőjét elhagyjuk. Ez a szempont egy formula pontos elsajátítását, alkalmazását, a komponensek kapcsolatának értését teszteli.

2. A lehetséges kimenet végállapotának meghatározása. A beérkező input alapján felépített gráfból meg tudjuk határozni a tanuló által inicializált FSA végállapotát és ezzel még a megoldás előtt valószínűsíthetjük az eredményességet.

3. Az input lehetséges kimeneteleinek teljes listája. Mivel egy feladatnak több jó megoldása lehetséges, minden megoldáshoz kell lennie egy FSA-nak. Ezek teljes listáját elő kell állítani.

4. Az újszerű megoldások kiértékelése. Ha egy feladatot fogalmazunk meg, tudnunk kell, hogy mennyi és milyen adatra van szükség amivel korrekt megoldást lehet generálni.

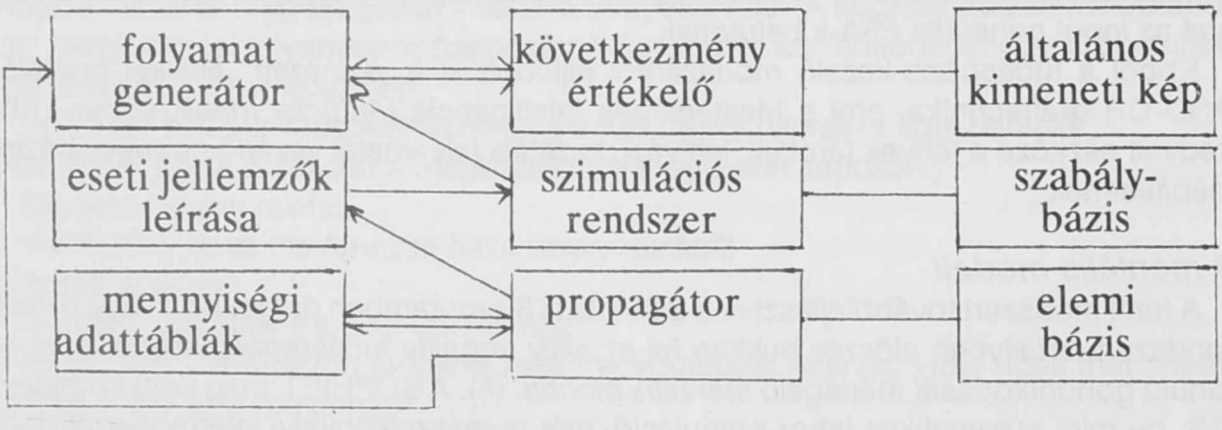
Ez azért lényeges, mert a program is felállít egy megközelítési hipotézist, és ha ennek az FSA-nak „nincs megoldás" a végállapota, a tanuló minden megoldását jónak ítéli, lévén a beérkező input vezérelte FSA végállapota: „nincs megoldás". Ugyanakkor, ha a tanuló talál egy eddig nem rögzített, de a végeredménnyel egyező végállapotú automatát, amelynek átmenetei érintik az összes szükséges és elégséges csomópontot (ami véletlenül is bekövetkezhet), meg kell határozni, hogy a bejárás menynyire releváns az aktuális feladathoz.

A mentális modellt a SOPHIE II illetve III programokban továbbfejlesztették. Beépítettek egy hibafelderítő és elemző modult, amellyel lehetővé vált a hiba elkövetésének pillanatában történő elemzése. Az értékelés eredményét felhasználva generálhatunk egy dialóguson alapuló azonnali korrekciós rutin, ami a hiba jellegétől, hatáskörétől és helyétől függ. A módszer önálló működésre is alkalmas, „reaktív tanuló-környezetnek" nevezték el (19).

A hibakereső modell

Ez az irányzat az eddigi próbálkozások közül a legsikeresebb kísérlet a humán gondolkodás elektronikus szimulálására. A SOPHIE III-ban a tudásbázis (*elektronics expert*) felett a hibakereső (*troubleshooting*) működik. Stratégiája egy időben jelent meg a BLOCK TUTOR (6) hibaelemzőjével és azzal algoritmus szempontjából csaknem teljesen megegyezik. Az optimalizált hibakereső három szintre tagolódik:

Az alsó szinten mennyiségi adatok – a SOPHIE III-ban például (és a további eseteket



is innen idézzük) áramköri elemek jellemző kimeneti ill. bemeneti adatai, a kapcsolhatósági jellemzők, kapcsolási módok és a felhasználható elemek bázisa található. A második szint a minőségi jellemzők tára, a kapcsolási esetek, a kimeneti tulajdonságok és a törvények leírása. A harmadik szint az általános szint, ahol a megtervezett áramkör tulajdonságait szimuláló folyamat szerveződik. Az általános kimeneti kép egy, az inputokra épülő működési gráf, amelynek felépítésébe bárhol be lehet avatkozni és

a módosításoknak megfelelően a fagenerátor újraszervezi azt.

A fenti tudásbázist mechanisztikus mentális, illetve causal – quantitativ modellnek nevezték el, egyben jó példa az elektronikus, optimalizáló hibakereső modell működésére.

Amint azt a fentiekből láthatjuk, az Intelligens Tutor Rendszerek elméleti alapja három tudományos terület eredményeire épül. A mesterséges intelligencia (Artifical Intelligence – AI), a kognitív pszichológia és a számítógépes nyelvészet törvényeit, szabályait ötvözve építi fel tudásbázisát, amellyel a humán gondolkodás folyamatát szimulálva megkísérli a tanítás gépi adaptálását. Ha megkíséreljük a mai magyar oktatási rendszerbe integrálni a számítógépes oktatást, mint kiegészítő eszközt, olyan programcsomagokat kell összeállítanunk, amelyek a fenti modellek működéséhez hasonlóan, lehetővé teszik minden pedagógus és tanuló számára az egyszerű hozzáférést, a könnyű használatot. Ehhez azonban több részletkérdést tisztázni kell. Ezt csak rendszeresen közzétett tapasztalatok, fejlesztési eredmények, kutatási folyamatok ötvözésével lehet megoldani.

Legsúlyosabb probléma a magyar nyelvű természetes nyelvi interfész hiánya, bár több kísérlet történt már ennek megvalósítására (20). De megoldatlan még a hazai szokásokra, beidegződésekre épülő felhasználói felület (*user interface*) optimális kialakítása, vagy a tudásbázis kezelésére rendszeresíthető olyan specifikus operátorbázis kiépítése, amelyik a magyar nyelv kötetlen szórendű mondataiból is képes kihámozni a beszerkesztett informémákat. Nincs intézményesített programgyűjtő, tároló, terjesztő szervezet, nem koordinálja semmi az önállóan dolgozó műhelyek, egyének munkáját.

E dolgozat felhívás is egyben, hogy ezeket a szervezeti hiányosságokat legalább publikációs szinten, részlegesen feloldjuk. Tegyük közzé tapasztalatainkat, elképzeléseinket rendszeresen, hogy a kiemelkedően magas színvonalú, de elszigetelten dolgozó számítástechnikai szakembergárda felzárkózhasson a nyugati technológia alkalmazóinak élvonalához.

IRODALOM

(1) *Smith, S.G. & Sherwood, B.A.*: Educational Uses of the PLATO Computer System. Science, 1976/192. sz. 344-352. p.
(2) *Price, R.V.*: Computer Aided Instruction. A Guide for Authors. Brook/Cale Publishing Corp. USA, 1991.
(3) *Carbonell, J.R.*: Mixed Initiative Man-Computer Instructional Dialogues. Doktori disszertáció, Massachusetts Institute of Technology Cambridge, 1970.
(4) *Yazdani, M.*: Intelligent Tutoring Systems: An Overview. In Artifcal Intelligence an Education. volume one (10) Ablex Publishing, 1987. 184-201. p.
(5) *Brown, J.S., Burton, R.R. & Bell, A.G.*: SOPHIE: A step towards a reactive learning enviroment. International Journal of Man-Machine Studies, 1974/7. sz. 675-696. p.
(6) *Brown, J.S., Burton, R.R.*: A paradigmatic example of an artifically intelligent instructional system. International Jurnal of Man-Machine Studies, 1978/10. sz. 323-339. p.
(7) *Bork, A.*: Learning with computer. Bedford, MA: Digital Press, 1981.
(8) *Uttal, W.R.*: On conversational interaction. In *J.E. Coulson* (Ed.) Programmed Learning and Computer Based Instruction. New York:Willey, 1962. 1-11. p.
(9) *Kidd, A.L.*: Knowledge Acquisition for Expert Systems. Plenum Press, New York, 1987.
(10) *Anderson, J.R., Corbett, A.T., & Reiser, B.J.*: Essential LISP. Reading, MA: Addison-Wesley, 1986.
(11) *Shortliffe, E.H.*: Computer-based medical consultations: MYCIN. American Elsevier, New York, 1976.
(12) *Clancey, W.J.*: Tutoring Rules for guiding a case method dialogue. In *D. Sleeman & J.S. Brown* (Eds.) Intelligent Tutoring Systems. Academic Press London, 1982. 201-225. p.
(13) *Tomelein, I.D., Jonson, M.V. (J.r.), Hayes-Roth, B. & Levitt, R.E.*: SIGHTPLAN: A Blackboard Expert System for Construction Site Layout. In *J. Gero* (Ed.) Expert systems in computer aided

design. Elsevier Science Publishers B.V. North-Holland, (1987).
(14) *Van Lehn, K.*: Felicity conditions for human Skill acquisition validating an AI-based theory. (Tech.Rep. CIS-21) Palo Alto, CA:Xerox Palo Alto Researc Center, 1983.
(15) *Reigeluth, C.M. & Curtio, R.V.*: Learning situations and instructional models. In *R. Gagnè* (Ed.) Instructional Technology: Foundations (7) Laurence Erlbaum Associates Publisher, Hillsdale, New Jersey. 175-206. p.
(16) *Collins, A.*: Process in Acquiring Knowledge. In *R.C. Anderson, R.J. Spiro & W.E. Montague* (Eds.) Schooling and the Acquisition of Knowledge. Laurence Erlbaum Assotiates, Hillsdale, New Jersey, 1977.
(17) *Steven, A.L. & Collins, A.*: The goal structure of a Socratic tutor. Proceedings of the National ACM Conference, Seattle, Washington. Association for Computing Machinery, New York, 1977. 256-263. p.
(18) *Woods, W.A.*: Transition Network Grammar for natural Language Analysis. CACM. 13. (10), 1970. 591-606. p.
(19) *Brown, J.S., Burton, R.R. & de Kleer, J.*: Pedagogical natural language and knowledge engeneering techniques in SOPHIE I and III. In *D. Sleeman & J.S. Brown* (Eds.) Intelligent Tutoring Systems Academic Press London, 1982. 227-282. p.
(20) *Prószéki, G.*: Számítógépes nyelvészet. Akadémiai Kiadó, Budapest, 1990.