Egri-Nagy Attila

Digitális evolúció

Charles Darwin *A fajok eredeté*ben (DARWIN 2000) választ adott a földi élővilág sokféleségének kérdésére. Ha az élőlények képesek exponenciális mértékű szaporodásra, de a környezetük eltartóképessége véges, akkor az egyes egyedek közti legapróbb különbségek is fontossá válnak, amikor eldől, hogy ki maradhat életben. Ha az apró eltérések örökletesek, akkor hosszú távon szükségszerű folyamattá válik a környezethez és a többi fajhoz történő adaptáció. A természetes szelekció tehát egy igen egyszerű elv, amely képes létrehozni a fajok hihetetlen sokszínűségét. Egyszerűsége mellett van még egy másik, figyelemre méltó tulajdonsága is: a természetes szelekció egy absztrakt elv, amely nem kötődik a földi élővilág részleteihez. Replikátor, véges erőforrásokkal rendelkező környezet, öröklődő variabilitás – ezek a fogalmak elvonatkoztathatók a bioszférától. Például a replikátor egy olyan entitás, amely létre tudja hozni saját másolatát, s nemcsak a baktériumok és a nyulak képesek ezt a trükköt megcsinálni. Ily módon lehetőség van arra, hogy az absztrakt fogalmakat többféle konkrét tartalommal töltsük fel, alternatív élővilágokat hozva ezzel létre. Jelen írás egy ilyen alternatív világot ismertet, a számítógépes programok digitális ökoszisztémáját. Ez a vállalkozás a mesterséges élet (LEVY 1993) diszciplínájába illeszkedik, amely tudományág alapfeltevése szerint az életjelenségek elvonatkoztathatók a konkrét megvalósulásaiktól és mesterségesen reprodukálhatók.

1. AZ ANALÓGIA KIBONTÁSA

Az ugyanazon absztrakt fogalmat kitöltő konkrét tartalmak analóg viszonyban állnak egymással. A földi élővilág és a digitális élőlények világa között is ilyen analóg kapcsolat van. Legelőször ezt az analógiát kell kibontanunk, összefoglalva a különböző digitális evolúciós rendszerek (RAY 1991; ADAMI–BROWN 1994; PARGELLIS 1996; EGRI-NAGY 2001) sajátosságait. Ennek az analógiának a felvázolásakor a legnehezebb azt eldönteni, hogy a kétféle világ közti különbségek az analógia hiányosságaiból vagy a digitális közeg eltérő jellemvonásából származnak.

1.1. A REPLIKÁTOR

A számítógépes program kettős természettel bír. Egyrészt végrehajtható utasítások sorozata, másrészt – mivel a programkód saját maga is a memóriában helyezkedik el – tekinthető egyszerű adatként, puszta számsorozatként. Ebből a dualitásból az következik, hogy a program képes saját magát feldolgozni mint input adatot. Az adattranszformáció speciális eseteként a program adatokat mozgathat a memória egyik területéről a másikra, s ha ennek az algoritmusnak az inputja saját maga (az a memóriaterület, ahol utasításai elhelyezkednek), akkor a program magáról készít egy

másolatot, s így a végrehajtás után már két példány van a számítógép memóriájában. Megvan tehát a replikátor, nevezhetjük akár digitális élőlénynek (vagy a továbbiakban némi hanyagsággal egyszerűen élőlénynek).

A magas szintű programozási nyelvek szigorú szemantikájuk miatt nem alkalmasak az evolúciós fejlődésre, hiszen egy apró változtatás tönkreteheti az egész programot. Így a digitális élőlények algoritmusa a gép belső reprezentációjához közel álló gépi kódú nyelven írható le a legkönnyebben. A fölösleges utasítás szintű részletek elhagyásával egy egyszerű digitális élőlény működésének fő lépései a következők:

1. Saját méretének megállapítása (hány elemi utasításból áll, azaz mennyi memóriacellát foglal el);
2. az utód számára megfelelő méretű memóriaterület lefoglalása;
3. másolási ciklus, memóriacellánként (utasításonként) lemásolja magát a frissen lefoglalt memóriarészre;
4. osztódás, az utód elválik a szülőtől és elkezdi végrehajtani saját algoritmusát.

Ha a digitális élőlények fenti formájának megfelelőjét keressük az élővilágban, akkor az a replikációra képes RNS-molekula lenne, hiszen a test gyakorlatilag egybeesik a genetikai kóddal.

1.2. ÖRÖKLŐDŐ VARIABILITÁS

Az evolúciós fejlődéshez szükség van változatosságra, amelyből a természetes szelekció mintegy kedvére válogathat. Ezt a variabilitást biztosítják a mutációk, melyek lehetnek másolási hibák, vagy külső hatásra (pl. kozmikus sugárzás) történő spontán változások. A számítógép azonban elvileg tévedhetetlen, így ha nem generálunk mesterségesen mutációkat, akkor csak teljesen azonos utódok születnek. A probléma megoldására bizonyos valószínűséggel és eloszlás szerint a digitális élőlények programjának végrehajtása során hiba következik be, vagy spontán adatváltozás történik. Ezenfelül a variabilitás további forrásai az ún. származtatott változások, melyek a már megváltozott program megváltozott futásából következnek (egy kis mutáció egy egyedben teljesen eltérő következő utódot eredményezhet).

1.3. KÖRNYEZET, VÉGES ERŐFORRÁSOK

1.3.1. Anyag

A digitális világban az anyag alapvető építőelemeit az egyszerű adatelemek jelentik: egész számok vagy – ami ugyanaz – egyszerű utasítások. Alighanem az analógia megsértése, hogy az eddigi rendszerek ezt az erőforrást korlátlannak tekintik. Egy újabb véges erőforrás új kompetíciót is jelentene, ami gazdagítaná a digitális élőlények interakciós lehetőségeit.

1.3.2. Energia, anyagcsere

Az energiát a processzor[1] számítási ereje jelenti. Ahogyan a Nap energiája folyamatos mozgásban tartja a földi élővilágot, úgy változtatja a processzor a memória adatmintázatait.

Az energia eloszlásának, elosztásának miként je a digitális világ hangolható paramétere. Lehet egyenletes, azaz minden digitális élőlény egyenlő mennyiséget kap, vagy lehet akár méretfüggő. De van más mód is a többletenergia megszerzésére: az anyagcsere mintájára (ahol a felvett anyag feldolgozásával nyerhető energia) a digitális világban az adattranszformációk, számítási feladatok végrehajtása eredményezhet több processzorciklust. Például a digitális élőlény számokat olvas be a saját processzorába, összeadja őket, aztán az eredményt visszadobja környezetébe, s ezért jutalmat kap. Ezen trükk révén foghatók a digitális élőlények hasznos munkára. Az anyagcsere persze maga is egy energiaigényes folyamat, így az analógia itt jól működik.

1.3.3. Tér

A digitális élőlények élettere a számítógép memóriája. Ennek a térnek a topológiája elfajult: a tér minden pontja (minden memóriacella) egyforma távolságra van egymástól (mindegy, hogy a szomszédos cellába másolunk adatot, vagy egészen más memóriacímre). Itt azonban megint felmerül a kérdés: vajon ezt az analógia hiányosságának, vagy a digitális közeg alapvető jellegzetességének kell-e tekintenünk? Vajon szükséges-e a háromdimenziós térszerkezet az evolúciós fejlődéshez?

A digitális élettérre is „ráhúzható” tetszőleges topológia, használhatók tetszőleges egész dimenziószámú sejtautomata-szerű térmodellek.

1.3.4. Biogén környezet

Ha a digitális környezet nem elég komplex, s időben nem változik, akkor a rendszer egy, az adott környezetre jellemző, optimális végállapot felé konvergál. Ez a célja a hagyományos genetikus algoritmusoknak (HOLLAND 1975), amelyek esetében a megoldást pont az a végállapot jelenti, amelyhez a populáció[2] – jó esetben – konvergál. Ha azonban az állandóan változó többi élőlény is a környezet része, egymásnak határozva meg az életfeltételeket, ha az egyes leszármazási vonalak koevolúciós viszonyban vannak egymással, akkor az állandóan változó követelmények révén az evolúciós rendszer nem tud megragadni egy állapotban.

[1] Ez a processzor nem a számítógép tényleges fizikai processzora, hanem egy szoftveresen szimulált virtuális gép. Így egyelőre nem fenyeget annak veszélye, hogy a digitális élőlények káros vírusok mintájára elszaporodjanak és számítógépes rendszereken élősködjenek.

[2] A populáció szó az evolúciós programozásban tipikusan csak a digitális élőlények összességét jelenti.

1.3.5. Szelekció

Szemben a standard genetikus algoritmusokkal, a digitális evolúciós rendszerekben mesterséges szelekció helyett természetes szelekciót alkalmazunk, azaz nem mi választjuk ki a túlélőket egy meghatározott szempont szerint, hanem ténylegesen az a digitális élőlény marad fenn, amely képes alkalmazkodni környezetéhez, s képes a szaporodásra. Ily módon a fitneszt csak mint utólagos mérőszámot értelmezzük:

$$f = \frac{m}{\gamma}$$

ahol *m* az érdem (merit), az élőlénynek jutó energia mennyisége; γ az utódlétrehozási idő (gestation time), azaz a két sikeresen végrehajtott utódnemzés között eltelt idő a szaporodáshoz szükséges utasítások számában mérve, beleértve a feladatok megoldásával töltött időt is. Ebből jól látható, hogy egy élőlény kétféleképpen tudja növelni életerejét: kódoptimalizálással lecsökkenti az utódlétrehozási időt, vagy egy új számítási feladat megtanulásával (a környezethez való alkalmazkodás) növeli az érdemét.

A szelekciót az az algoritmus jelenti, amely eldönti, hogy az élettér telítettsége esetén, egy új élőlény létrejöttekor melyik másik élőlényt kell elpusztítani. A döntés történhet az életkor, az értelmetlen utasítás-végrehajtásáért járó büntetőpontok vagy az érdem[3] alapján.

1.4. GENOTÍPUS, FENOTÍPUS

A genetikai kód a digitális élőlény esetében az azt alkotó utasítássorozat. A fenotípust, azaz a külsőleg is megjelenő tulajdonságokat az utasítássorozat végrehajtása adja. Egy egyszerű ciklust megadó genotípus három utasításból áll (címke, végrehajtandó utasítás, ugrás a címkéhez), a hozzá tartozó fenotípus azonban olyan hosszú, ameddig a ciklust futni hagyjuk.

Egy fenotípust több különböző genotípussal meg lehet adni, hiszen ugyanaz az algoritmus többféle kóddal megvalósítható. Ugyanakkor egy genotípushoz több fenotípus is tartozhat, ha a vezérlés a genotípus különböző részeire kerül valamely, a környezetből beolvasott adat alapján.

2. JELENLEGI EREDMÉNYEK

A kísérletek alapvetően két csoportba sorolhatók. Az első esetben egy, már meglévő kódrészletet szeretnénk optimalizálni, mind a tárhely, mind az idő tekintetében. A legegyszerűbb változat esetén ez a meglévő programkód a replikáció algoritmusa. A második esetben – a digitális élőlényeket információgazdag környezetbe helyezve – célunk az evolúciós tanulás elérése.

[3] Az érdemmel természetesen valamilyen mértékben visszacsempésződik a mesterséges szelekció.

2.1. KÓDOPTIMALIZÁLÁS

Minden élőlény egyforma méretű processzoridő-szeletet, azaz ugyanannyi energiát kap, így a szelekciós nyomás egyértelműen a gyorsabb replikáció irányába hat. A fejlődés nyilvánvalóan korlátolt, hiszen az optimális replikátor elterjedése után semmilyen evolúciós újítás megjelenése nem várható.

2.1.1. Optimalizációs technikák

Stabilan, minden implementációban megjelenik a fordítóprogramok által is ismert ún. *loop-unrolling*, melynek lényege a ciklusszervezés relatív költségének csökkentése a ciklusmag utasításainak megduplázása révén. Ily módon egy értékes (tipikusan másoló) utasítás végrehajtására kevesebb adminisztrációs („Kell-e még másolni?") jellegű utasítás jut. További technikák léteznek és alkalmazhatók az élőlények saját méretmeghatározásának gyorsítására: a hosszadalmas, utasításonkénti önmegmérés helyett például félig mérni, aztán szorozni kettővel, vagy egyéb számítással meghatározni, „kitalálni" a méretet.

Ezekkel a technikákkal az eredeti, ember által írt ősreplikátorhoz képest jelentős sebességnövekedést tudnak elérni. Átlagosan a duplájára növekszik a replikációs sebességük, azaz fele annyi idő alatt képesek szaporodni a kifejlődött digitális élőlények.[4]

2.1.2. Gazda-parazita koevolúció

A digitális evolúció leglátványosabb sikerét a Thomas S. Ray *Tierra* nevű programjában (RAY 1991) megjelenő információs parazitizmus jelentette. A paraziták kihagyták saját programjukból a másoláshoz szükséges utasításokat, nagyban lecsökkentve ezzel méretüket, s gyorsítva szaporodásukat. A másolást pedig a szomszéd gazdaélőlény megfelelő programrészletének végrehajtásával oldották meg. Ezután megjelentek a hiperparaziták, melyek a paraziták költségére voltak képesek replikálódni. A gazda-parazita koevolúció megszűnt az optimális replikátor létrejöttével.

2.2. EVOLÚCIÓS TANULÁS

Az anyagcsere analógiájára, a digitális élőlények számítási feladatok végrehajtásával nagyobb energiaszelethez juthatnak (ADAMI–BROWN 1994). Bár az individuális tanulás lehetősége nem kizárt, hiszen egy digitális élőlény bármikor átírhatja saját kódját, itt elsősorban arról van szó, hogy a leszármazási vonal tanul. Az ősreplikátor valamelyik leszármazottjában megjelenik az input adat beolvasásának kódja. Ennek valamelyik utódjában megjelenik az output adat környezetbe visszaírásának a kódja. Aztán egy következő utódban megjelenik a kettővel való szorzásé, és ekkor van egy digitális élőlényünk, amely „megtanulta" a számok duplázását.

[4] A pontos érték függ a processzor-architektúrától, valamint az ősreplikátor hatékonyságától.

2.3. A PROCESSZOROK FEJLŐDÉSE

Még több szabadságot jelent az evolúciós folyamat számára, ha a digitális élőlényeket futtató virtuális processzorok is fejlődhetnek (EGRI-NAGY 2001). Az univerzális Turing-gép mintájára, a processzor szerkezetének és utasításkészletének leírása is bekerül az élőlény genomjába, így ugyanúgy alá van vetve a mutációnak. Más szóval, a genotípus–fenotípus kapcsolat is az evolúció ellenőrzése alá kerül. Az eddigi megfigyelések összefoglalása a következő:

1. Az evolúciós folyamat során az utasításkészlet teljes átalakulása figyelhető meg.
2. Az utasítások száma, valamint a processzor struktúrája változatlan marad.[5] Ez a konzervativizmus alighanem az univerzális processzor konstrukciójának tudható be.

3. FELHASZNÁLÁSI LEHETŐSÉGEK

El kell ismerni: nem biztos, hogy a digitális evolúció kettős célja egy időben megvalósítható. Az elsődleges cél az, hogy a gépben a földi élethez mérhető spontán diverzitást hozzunk létre, míg a másik cél, hogy hasznos munkára fogjuk a digitális élőlényeket. Egy elméleti–biológiai és egy gyakorlati–informatikai cél. A gyakorlati megfontolások szeretnék a folyamatot irányítani, míg biológiai szempontból pont az evolúció „elszabadítása", a nyílt végű (open-ended) fejlődés a lényeg.

3.1. BIOLÓGIA

A digitális evolúció már evolúció annyira, hogy evolúciós (pl. a szaggatott-egyensúly elmélet, GOULD–ELDREDGE 1993) vagy genetikai hipotézisek (szekvenciák bonyolultsága) tesztelésére alkalmas legyen, mindemellett rendelkezik a számítógépes kísérletek összes jó tulajdonságával. A földi élővilág fejlődésével ellentétben, itt ténylegesen megfigyelhető maga a folyamat, s a megfigyelő totális ellenőrzése alatt tartja a rendszert, ugyanaz a kísérlet többször is megismételhető. Ezen túlmenően, a megfigyelés semmilyen hatással sincs a rendszer működésére. Ezek a tulajdonságok lehetővé teszik oktatási eszközként történő felhasználását is.

3.2. SZOFTVERFEJLESZTÉS

A szoftverfejlesztés célja – erősen leegyszerűsített megfogalmazásban – megbízhatóan működő komplex rendszerek létrehozása. Eme cél megvalósításával a mai napig gondok vannak, a különböző módszertanok kisebb-nagyobb sikerei ellenére. Tekintve, hogy az evolúció során létrejött élőlények pont ezekkel a kívánatos tulajdonságokkal bírnak, ésszerűnek látszik a digitális evolúciót programozási problémák megoldására használni. Ahogy a kísérleteknek is alapvetően két típusa van, úgy a lehetséges fel-

[5] Ez a konzervativizmus alighanem az univerzális processzor konstrukciójának tudható be. Bár az univerzális processzor ötletének lényege, hogy ne kelljen evolválható processzorokat tervezni, hanem hagyjuk azokat kifejlődni, szembe kell nézni az univerzális processzor tervezésének metaproblémájával.

használási mód is kétféle: már meglévő kódrészletek optimalizálása, illetve új algoritmusok tenyésztése.

Optimalizálás esetén a javítani kívánt kódrészletet kiegészítjük a replikációs algoritmussal, és a környezetet oly módon állítjuk be, hogy csak az adott feladat elvégzéséért jár jutalom (ezzel biztosítjuk az optimalizálandó kód megmaradását). A szelekciós nyomás a minél gyorsabb replikáció irányába hat, így a digitális élőlények kénytelenek gyorsítani a megadott feladat kódján is. A problémák ezzel a megközelítéssel a következők:

– Csak alacsony szintű, rövid programrészletek optimalizálhatók ily módon. Ez önmagában nem gond, hiszen pont a hardverszinthez közeli, sokszor lefutó programoknál van értelme az optimalizálásnak.
– Az evolúciós fejlődés után nem biztos, hogy könnyedén el lehet választani a feladatot végrehajtó részt a replikációt végrehajtó kódtól. A részek elválasztása, a koncepcionális határok kijelölése alapvető emberi igény a megértés biztosításának érdekében, de az evolúciós fejlődés során ezek a határok eltűnnek, a dolgok összekeverednek, egyszerre több minden történik.[6]
– A valós processzor utasításkészletének meg kell egyezni a digitális evolúció virtuális processzorában használttal. A jelenleg használt processzorok tervezésénél nem volt szempont az evolválhatóság, így azok gépi kódú nyelve alkalmatlan a mutációval történő transzformációkra. Erre a problémára ígér megoldást az evolválható hardverkutatás területe (Liu 2001).
– Ha a valódi processzorok tervezésénél szempont lenne az evolválhatóság tulajdonsága, akkor nem lenne szükségünk a szoftveres virtuális gépekre, de szembe kellene néznünk az evolúció elszabadulásának káros következményeivel.[7]

A felhasználás másik módja, amikor az evolúciós tanulást újféle algoritmusok kitenyésztésére használjuk. Itt is szembe kell néznünk a szétválaszthatóságnak, illetve a virtuális processzorok használatának a problémájával. Ezen túl a megoldani kívánt feladatot le kell bontanunk részfeladatokra (hisz valószínűtlen az összeadás képességének a megjelenése, ha nem előnyös a számok beolvasásának és kiírásának képessége). Ha pedig megvan a feladat felbontása, akkor az a legtöbb szoftverfejlesztési paradigma mellett a probléma megoldásának jelentős részét teszi ki, s nincs szükség az evolúciós fejlesztésre. Mindezen kritikák ellenére állítható, hogy van mit tanulni az emberi programozónak az evolúciótól, hisz az radikálisan másként „gondolkodik", s olyan jó megoldásokat talál, amelyek az embernek egyszerűen nem jutnak eszébe. Talán ebből származik a megfigyelőnek az az érzése, hogy valaki/valami intelligens van a folyamat mögött.

4. DIGITÁLIS EVOLÚCIÓ ÉS EMERGENCIA

Az emergencia, mint minden gyakran használt kifejezés, nehezen megfogható jelentésű fogalom. Ily módon az is kérdéses, hogy mely rendszerekre alkalmazható, s melyekre nem. Ezért szükséges tisztázni, mennyiben emergens jelenség a digitális evolúció, s mennyiben nem az.

[6] Az optimalizálás alapvetően az érthetőség ellen dolgozik.

[7] Amíg a digitális élőlények kontrollált körülmények között virtuális processzorokon futnak, semmilyen számítógépes vírus jellegű veszély nem áll fenn.

4.1. MODELL, RENDSZER

A digitális evolúció – mint minden mesterségesélet-projekt – azt állítja, hogy nem modellje, nem szimulációja semmilyen, a földi élővilágban fellelhető rendszernek. A digitális élőlények nem reprezentálják a földi élőlényeket, csak analóg viszonyban állnak velük, s a saját világukban teljesen önálló entitások. A standard érvelést ezzel szemben a „szimulált vihartól nem leszünk vizesek" típusú érvek jelentik. Az természetesen igaz, hogy mi nem leszünk vizesek, de mi nem vagyunk benne a rendszerben, a szimulációban szereplő földrajzi objektumok azonban igen. A digitális élőlények világát mint zárt rendszert[8] határozzuk meg, abban az értelemben, hogy teljesen be van ágyazva a mi világunkba, pontosabban a fizikailag létező számítógépekbe. Ezért van az, hogy semmilyen hatással nincs a digitális élőlényekre, ha felfüggesztjük futásukat, hálózaton átmásoljuk őket stb.

4.2. EMERGENCIA

Az emergencia abban az értelmében, hogy az egyszerű elemek kapcsolataiból valami komplexebb, más leírási szintet igénylő mintázat jelenik meg, nem alkalmazható közvetlenül a digitális evolúcióra. Az élet eredetének kérdését vizsgáló rendszerek túl bonyolult elemekkel (összetett utasításokkal) dolgoznak (PARGELLIS 1996). Az evolúciós tanulás során pedig az jön ki a rendszerből, amit beleraktunk, hiszen a feladatok definícióit, az input-output párosításokat mi adjuk. A mutációk, azaz apró változások összjátékát, ami gyakran egy nagyobb evolúciós lépésben kulminálódik, tekinthetjük emergens jelenségnek, de az emergencia elveit a digitális evolúciónál leginkább az elkövetkező kutatási irányok kijelölésére használhatjuk fel.

4.2.1. Digitális ökológia

Nyilvánvaló, hogy egy statikus környezetben az egész rendszer konvergál egy végállapot felé (vagy több lehetséges végállapot közül egyhez), ami biológiai szempontból nem túl érdekes jelenség. Próbálkozhatunk a környezet mesterséges manipulálásával (komputációs feladatok hozzáadása, elvétele), de ezzel megint csak azt kapjuk vissza a rendszerből, amit beleraktunk. Nyilvánvaló, hogy a dinamikusan változó környezetet a biogén környezetnek kell reprezentálnia. Biztosítanunk kell, hogy a digitális élőlények többféleképpen léphessenek kapcsolatba egymással, túl a kompetícióból adódó indirekt kölcsönhatáson, igazi, az élőlények kölcsönhatásán alapuló emergens fejlődést hozva ezzel létre. A digitális ökológia bonyolításának/egyszerűsítésének a következő lehetséges módjai kínálkoznak:

1. Táplálékláncok bevezetése. A komputációs feladatok elvégzéséhez a jelenlegi rendszerekben korlátlan mennyiségben áll rendelkezésre az input adat. Pontosabbá teszi az analógiát, ha ezeket is az élőlényeknek kell előállítani, kialakítva ezzel a termelők és fogyasztók közötti függést. Az input adatnak ekkor összetett szerkezettel kell bírnia, tükrözve a táplálékláncon való végighaladást.

[8] Ez nem jelenti azt, hogy a saját világán belül nem egy nyílt rendszer, amelyen energia és anyag halad át.

2. A predáció lehetősége. Szintén nincs korlátozva az utód élőlény felépítéséhez szükséges utasítások mennyisége. Ha az építőelemek nem állnak szabadon rendelkezésre, akkor értelmet nyer az anyagért és energiáért történő predáció.

3. Inhomogén környezet, a rendszer különböző paramétereinek (pl. mutációs ráta) változása a térben. Ily módon értelmet nyer a helyváltoztatás, a térért való versengés a digitális élőlények között és különböző „éghajlatokhoz" történő alkalmazkodás.

4. Ökológiai niche-k definiálása komputációs feladat-részhalmazokkal[9].

4.2.2. Kooperáció, többsejtűség

A digitális élőlények mint sejtek közötti interakciós lehetőségek[10] bővítése lehetővé teszi a kooperáció megjelenését. Ez előfeltétele az egyes genomrészletek különböző feladatokra történő specializálódásának, s így a többsejtű digitális élőlények létrejöttének.

A többsejtűség digitális megfelelője a párhuzamos működés, ahol egy adott program több processzoron fut egyszerre, más-más részletét hajtva végre a kódnak. A párhuzamos működés nagyon egyszerűen megvalósítható a virtuális processzor utasításkészletének megfelelő kiegészítésével, de ennek a kiegészítésnek a használatba vétele nem jelenti egy új szelekciós egység megjelenését, márpedig az új szelekciós szint megjelenése lehet a végső kritériuma a digitális evolúció tényleges evolúció voltának.

4.2.3. Szexuális szaporodás

A szexuális szaporodás kezdetleges formája (genomtöredékek bekebelezése) megfigyelhető (RAY 1991), de az igazi értelemben vett kétszülős szex még nem jelent meg. Valószínűleg a jelenlegi digitális ökoszisztémák nem annyira komplexek, hogy megérné bennük szexuálisan szaporodni. Az igazi kérdés természetesen az, mik a feltételei annak, hogy a szexuális szaporodás megjelenése szükségszerűvé váljon, és tudunk-e ilyen feltételeket teremteni.

4.2.4. Finomhangolás

Az életjáték példája azt mutatja, hogy a különleges viselkedést mutató rendszer szabályainak meghatározásához hosszadalmas kísérletezgetés, finomhangolás szükségeltetik (LEVY 1993). A digitális evolúció jóval bonyolultabb rendszer, mint az életjáték, így talán érthető, hogy miért igényel a digitális világ alaptörvényeinek meghatározása annyi időt. A szerző véleménye szerint a digitális evolúció (mely minden tekintetben

[9] Ökológiai niche (ökostátus): A populáció környezetében található forrásoknak az a része, amit a populáció egyedei kihasználhatnak. A niche jelentése fülke, tehát úgy értelmezhetjük, hogy az élettér a különböző populációk számára, a források által alkotott, képzeletbeli fülkécskékre tagolódik. Két azonos niche-ű populáció tartósan nem élhet ugyanazon a térbeli helyen. *(A szerk.)*

[10] A digitális evolúció jelenlegi fogalomrendszerének képlékenységét mi sem jellemzi jobban annál, mint hogy – a nézőponttól függően – a digitális élőlény lehet egy RNS molekulának, egy önálló sejtnek vagy egy magasabb rendű faj kifejlett egyedének analógiája.

megfelel a földi evolúciónak) megvalósíthatósága természetesen nem kérdéses. Tagadhatatlan előny, hogy van már egy működő példa (a földi élővilág), amely ötleteket adhat a részproblémák megoldásához. A feladat tehát a rendszer olyan irányú bővítése – vagy még inkább egyszerűsítése –, ami helyet csinál az emergens fejlődésnek.

5. AZ EVOLÚCIÓ JÖVŐJE

A digitális evolúció kutatása alig több mint egy évtizede kezdődött, s eddigi eredményei igazából nem válaszokat adtak, hanem új kérdéseket vetettek fel, teret nyitva a további kutatásnak – a munka nagyobbik része tehát még hátravan. Az analógia pontosítása nyilvánvalóan a biológusok és a programozók szoros együttműködését kívánja meg, hiszen úgy látszik, hogy a földi életről szóló biológiai tudás és az egyre nagyobb számítási erővel bíró és egyre jobban összekötött számítógépek találkozása révén az evolúció folyamatának belépése a digitális világba elkerülhetetlenné vált.

IRODALOM

Adami, Chris – Brown, C. Titus 1994. Evolutionary Learning in the 2D Artificial Life System „Avida". In Brooks, R. – Maes, P. (eds.): *Artificial Life IV*. Cambridge, MA, Bradford–MIT Press. 377–381. Lásd továbbá: http://dllab.caltech.edu

Darwin, Charles 2000. *A fajok eredete*. Budapest, Typotex.

Egri-Nagy Attila 2001. Physis – digitális evolúció. In Kampis György – Rokolyi László (szerk.): *Evolúció és megismerés*. Budapest, Typotex. Lásd továbbá: http://physis.sf.net

Gould, Stephen Jay – Eldredge, Niles 1993. Punctuated Equilibrium Comes of Age. *Nature*, 366. 223–227.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor, University of Michigan Press.

Levy, Steven 1993. *Artificial Life – Report from the Frontier Where Computers Meet Biology*. New York, Vintage Books.

Liu, Yong et al. (eds.) 2001. *Evolvable Systems: From Biology to Hardware*. Lecture Notes in Computer Science, vol. 2210. Tokyo, Japan.

Pargellis, A. N. 1996. The Spontaneous Generation of Digital „Life". *Physica D*, 91. 86–96.

Ray, Thomas S. 1991. An Approach to the Synthesis of Life. In Langton, C. G. et al. (eds.): *Artificial Life II*. 371–408. http://www.isd.atr.co.jp/~ray