Algoritmusok Oktatása Algoritmus Vizualizációval

Bende Imre

beiraai@inf.elte.hu
ELTE IK

Absztrakt. Útmutatást szeretnék adni arra, hogy az egyes algoritmusokat, milyen módon célszerű bemutatni, megtanítani algoritmus vizualizációs eszközökkel, hogy a diákok számára az algoritmusok elsajátítása hatékonyabb, látványosabb, élvezhetőbb legyen. Illetve kitekintek pár példával arra, hogy mely algoritmusoknál, mire kell odafigyelni, egy-egy eszköz kiválasztásakor. Időrendben kategorizálva vizsgálom az eszközöket, illetve azok újításait, tulajdonságait. Az egyes kategóriák hatékonyságát már publikált tanulmányokkal is alátámasztom.

Kulcsszavak: algoritmus vizualizáció, programozásoktatás, algoritmus

1. Bevezetés

A cikk során négy nagyobb részre osztom szét az algoritmus vizualizációs eszközöket (röviden: AV). Ezeket a kategóriákat a megjelenés időpontja, jelentősebb újítás bejövetele és az interaktivitási szintjétől függően alkottam meg. Az utóbbi, mostanában is releváns kategóriákát részletesebben is áttekintem vizsgálva azt, hogy hogyan lehetne az ezen csoportbéli eszközöket oktatásban is minél hatékonyabb oktatásmódszertani eszközként felhasználni, kiemelve pár konkrét algoritmuscsoport oktatási tapasztalatával, egyes esetekben megnézve annak „mérhető" eredményességet (a mérhetőt idézőjelbe tettem, mivel ez nem egy átfogó, minden esetben igaz eredményt ad, hanem csak a vizsgált csoportok sikerességét mutatja).

2. Algoritmus vizualizációk csoportjainak bemutatása

Az 1980-as években jelent meg [1] először az algoritmus vizualizáció a programozásoktásban. Ez idő alatt rengeteg eszköz jelent meg és nagy változások mentek végbe, tanulva az előzők hibáiból, hiányosságaiból. A következőkben négy részre szétosztva, időrendben tekintem át, részletezem az egyes időszakokat.

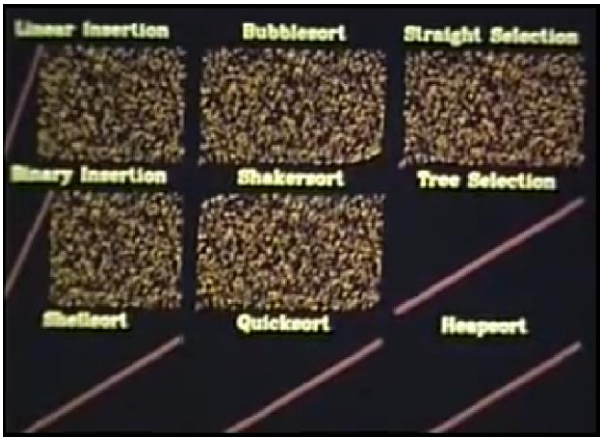
2.1. Statikus vizualizációk

Az algoritmusokat oktató tankönyvekben statikus vizualizációkat találhatunk meg. Ez a módszer az algoritmus lépéseit statikus illusztrációkkal próbálja elmagyarázni, melyekhez társulnak magyarázatok, illetve megjelenik egy leíró nyelven az algoritmus maga is (struktogramban, pszeudokódban, vagy egy adott programozási nyelvben vett implementációjában). Ezek bizonyos mértékben könnyebbé teszik az algoritmus működésének elképzelését, de összetettebb algoritmusoknál ez nem tud elégséges segítséget adni a működés megértéséhez, statikus mivolta miatt.

2.2. Animált/dinamikus vizualizációk

Az 1980-as évektől megjelentek az első algoritmus animációk, melyek egy-egy algoritmus működését próbálták ábrázolni, bemutatni. Az előző részhez hasonló illusztrációk most már folyamatában jelentek meg, így az algoritmus működése átláthatóbbá vált, gördülékenyebbé téve az algoritmus egész működésének a megtekintését. Az első ismertebb példa a „Sorting out Sorting" című videó volt [1. ábra], mely a rendezéseket mutatta be egy-egy példán keresztül, amelyekhez szóbeli magyarázat is társult [1]. Többségében az animációk értelmezéséhez azonban szükséges megfelelő előzetes tudás az algoritmusról, így egyéb segédanyag megléte is szükséges az oktatáshoz. Az efféle vizualizációk nem

mutattak szignifikáns javulást a tanulás hatékonyságában [3]. Ez főleg amiatt lehetett, hogy a tanulók csak az animáció nézői lehettek, nem pedig a tanulási folyamat résztvevői. Az algoritmus animációk azonban mindenképpen fontos alapot szolgáltak a későbbi vizualizációk létrejöttéhez.



1. ábra: Részlet a „Sorting Out Sorting" nevű algoritmus animációból, amelyen a rendezéseket hasonlítja össze (többek között a beszúró, buborékos, gyors, shell rendezéseket)

2.3. Interaktív vizualizációk

2.3.1. Interaktív vizualizációk bemutatása

Az 1990-es évektől, köszönhetően a személyi gépek elterjedésének, illetve később az Internet térhódításának megjelentek az interaktív algoritmus vizualizációk, ahol a hallgatók már résztvevőkké válhattak (nézőkből az algoritmus léptethetőségével, bemenet módosíthatóságával és egyéb módszerekkel a vizualizáció irányítóivá váltak). Az Internet segítségével könnyen, szabadon, ingyenesen elérhetővé váltak mások munkái, így a megfelelő forrás megtalálása után könnyen felhasználhatóvá váltak az eszközök.

A vizualizációk léptethetővé, visszatekerhetővé váltak (az egyes lépéseket az algoritmus megjelenítésénél is követhetjük). A bemenet állítható lett, így bármely inputra megnézhetővé vált az algoritmus működése. Egyes eszközök pedig az interakciót úgy próbálták növelni, hogy kérdéseket tettek fel azzal kapcsolatban, hogy mi lehet a következő lépés, mi változhat (ez az elem, ahogyan a diákoktól való interaktivitás szükségességét növelte, úgy a tanulás hatékonyságát is javította, ez részletesebben Naps tanulmányában is látszik [5]).

Ennél a fázisnál már érvényesülhettek a Myller által megfogalmazott kiterjesztett interaktivitási szintek [4] (ezek a szintek a megtekintés, ellenőrzött megtekintés, bemenet beírása, válaszadás, változtatás, módosítás, összerakás, bemutatás, reagálás).

Emellett egyre több olyan eszköz, környezet, keretrendszer is létrejött, amelyekkel a hallgatók önmaguk is létrehozhatnak vizualizációkat, így a kiválasztott algoritmus működését mélyebben megvizsgálva, még jobban megérthetik azt, sőt társaiknak bemutatva fejleszthetik egyéb kompetenciáikat is (kreativitás, kommunikációs, előadói készség).

2.3.2. Általános algoritmusok vizualizációi

A tanulók elsőként a programozási tételekkel találkoznak oktatási környezetben (például: összegzés, megszámolás, maximum-kiválasztás, keresés). Ezeken keresztül sajátítják el aztán az algoritmikus gondolkodást, így fontos, hogy már az elején átlássák ezeknek az algoritmusoknak az alapjait, működését. Nagy segítség lehet az algoritmus vizualizációs eszközök használata ahhoz, hogy tisztán lássák azt, hogy mi történik, milyen változások történnek az egyes lépésekben.

2. ábra: Az általam készített AV eszközből[1] egy minta, a megszámolás tétel bemutatására

A tanári szakdolgozatomhoz [2] készített algoritmus vizualizációs honlapot igyekeztem minél teljesebben létrehozni, hogy minden szükséges elem fent legyen azon, hogy a diákok könnyebben megérthessék, megtanulhassák az adott algoritmust. Így tehát nézzük meg, hogy milyen komponensekből állhat egy AV [2. ábra], illetve, hogy ez az én eszközömön, hogyan jelennek meg ezek:

- Vizualizáció. Az algoritmus működése, illetve a hozzá tartozó adatszerkezet látványos, könnyen értelmezhető formában kerül ábrázolásra. Jelen esetben a bemeneti tömböt végig láthatjuk, ezen különböző színekkel vannak jelölve a speciális elemek (megszámolás tételnél zölddel a feltételt teljesítő elemek, kékkel pedig az éppen vizsgált elem).
- Bemenet állíthatósága. Az algoritmus bemenetét tudjuk módosítani, így különböző eseteknél is meg tudjuk vizsgálni az algoritmus működését, kimenetét. A weboldalamnál a bemeneten kívül a megszámolás feltétele is állítható.
- Algoritmus. Az algoritmus megjelenik valamilyen leíró nyelven. Lehet az pszeudókód, struktogram, vagy akár valamilyen konkrét programozási nyelven megírott implementációja. Az aktuális lépés utasítását a szoftver jelzi (egy fekete háromszöggel).
- Lépések állíthatósága. Az egyes utasítások között megvan a lehetőség, hogy (előre/visszafele) léptethessük a vizualizációt. Sőt folyamatában is meg tudjuk tekinteni azt.
- Változók követése. Látjuk az aktuális lépésnél, hogy milyen változók vannak, illetve milyen értékkel rendelkeznek. Ez a vizualizációnál és az algoritmusnál is leolvasható.
- Leírás. A felületen van egy rövid leírás az algoritmusról, illetve az egyes lépések változásait is rövid megjegyzésekkel látja el.

[1] http://ermi46.web.elte.hu/dev/algotan/

2.3.3. Algoritmus felfedezés

Az algoritmus felfedezés célja, hogy „megengedjük a diákoknak, hogy tapasztalati úton felfedezzék a teljesítménybeli jellemzőket és relatíve különböző lehetőségeket egy-egy témakörön belül, ilyen témák lehetnek például a keresőfák, hash, rendezési algoritmusok" [9]. Így nem csak egy-egy algoritmust tudnak megismerni, hanem annak futásidejű és memóriabeli hatékonyságát is, illetve össze is tudnak hasonlítani többfajta megoldást hatékonyság szempontjából.

A Shaffer és munkatársai által írt tanulmányban [9] ennek eredményességét is vizsgálták a hash algoritmusok oktatásában. Két szemeszterben két-két csoportot vizsgáltak. Az elsőt hagyományos módszerekkel tanították, a hallgatók kaptak az anyaghoz külön felhasználható irodalmat, míg a másodiknál rövid bemutatás után a diákok egyedül, vagy párokban dolgozhatták fel az előre elkészített tutorial tartalmát, amiben algoritmus vizualizációkkal mutatták be az egyes algoritmusokat, amin keresztül egy probléma megoldását is összehasonlíthatták (az oktatótól persze lehetett közben kérdezni) több különböző elvvel, koncepcióval, így az egyes esetek teljesítményét, hatékonyságát is láthatták. Ezután a diákok egy tesztet töltöttek ki, melynek eredménye a következő táblázatban látszik:

Módszer	**2008 ősz**	**2009 ősz**
eszköz nélkül	67.8% (22 fő)	68.6% (31 fő)
eszközzel	77.3% (19 fő)	73.4% (48 fő)

1. táblázat: Összegzés a hash kvíz teljesítményéről [9]

Az eredményekből látszik, hogy akiket a tutoriallal tanítottak jobb pontszámot értek el a teszten, mint akiket a hagyományos módszerekkel. Viszont nagyon óvatosan szabad csak használni az ilyesfajta megközelítést, hiszen megvan a maga kockázata is. Ahogy Pinker írta: „Felfedezés útmutatás nélkül nagyon kockázatos és túl sok időt vesz igénybe. Útmutatás felfedezés nélkül pedig csak irányítás, ami jobban támaszkodik a memorizálásra, mint a megértésre." [6]

2.4. Komoly játékok

Habár nem feltétlenül sorolnám be időrendben ezt a csoport az előző kategóriabesorolásba, de mindenképpen említést szeretnék tenni a komoly játékokról (serious games). A komoly játékok: „bármilyen formájú interaktív számítógép-alapú játékszoftverek, egy vagy több játékos játszhatja valamilyen platformon, amelyet azért hoztak létre, hogy a használata a játékosoknak több legyen, mint szórakozás" [7]. A gamifikáció fogalom népszerűvé válása előtt, már igyekeztek a programozásoktatást is élvezhetőbbé és így hatékonyabbá tenni azáltal, hogy a feladatokat számítógépes játékokba ágyazták, illetve azok elveivel építették össze.

Miért is volt kézenfekvő alap a számítógépes játékok, illetve azok világának felhasználása a tanulásban, illetve az oktatásban? (itt alapul szolgáltak a [8]-ban már megfogalmazott pontok)

- A számítógépes játékok népszerűek. Közismert, hogy a fiatalok (sőt az átlagéletkor egyre magasabbá válik) nagyon sok időt töltenek számítógépes játékokkal, így kézenfekvő, hogy célszerű oktatási eszközként felhasználni a mai világban.
- A számítógépes játékok interaktívak. Teljes egészében a felhasználó irányítja, az ő általa véghez vitt műveletek kontrollálják az eseményeket a szoftverben.
- A számítógépes játékok kompetitívek. A játékosok rengeteg időt fektetnek nem csak a játékokba, hanem azok megértésébe, megismerésébe is. Mindezt azért, hogy jobbak legyenek abban, javítsanak eredményeiken, illetve legyőzhessék önmagukat, vagy akár más játékosokat is.
- A számítógépes játékok szórakoztatóak. A számítógépes játékok használata az algoritmus tanulást élvezhetővé, érdekessé teszi.

A komoly játékok témakörében a már meglévő játékoknál vizsgálták a tanulási lehetőségeket, majd ebből kiindulva hoztak létre olyan játékokat, amelyek a diákok valamilyen készségeit fejlesztette. Ezek célja többnyire a nyelvi készségek fejlesztése, ügyességfejlesztés, valamilyen témakör tudásának átadása (többek között használtak ilyen szoftvereket a katonaságban és az egészségügyben is). Így a programozásoktatásban nem, vagy csak közvetetten használhatók fel ezek a játékok, azonban a következő részben látjuk meg azt, hogy mi lesz akkor, ha vegyítjük a programozásoktatást és a számítógépes játékokat oktatási környezetben.

2.5. Játékos vizualizációk

2.5.1. Játékos vizualizációk bemutatása

A 2010-es években vált népszerű kifejezéssé az oktatásban a gamifikáció [10]. Ez az algoritmusoktatást is érinti, megjelennek az olyan vizualizációs eszközök, amely a tanulási folyamatot igyekeznek játékos környezetbe ültetni, ezzel növelve az interakció mélységét, ami pedig ezáltal a tanulás hatékonyságát is fejleszti (több eredmény is született erre vonatkozóan, többek között Karavirta disszertációjában gyűjtötte össze a kísérletek eredményeit, amiből szépen látszik ez az eredménye [3]).

3. ábra: Feladat a „Sort Attack" nevű játékban [10]

A „Sort Attack" nevű alkalmazás [3. ábra] igyekszik az algoritmustanulást játékossá tenni különböző elemekkel, illetve emellett algoritmus vizualizációt használ közvetítő közegként (emiatt is került bele ebbe a részbe). A program rendezési algoritmusokat mutat be, illetve egyre nehezedő pályákon kell az algoritmusok lépéseit végig követni megfelelő hibázási lehetőséggel. A következő felsorolásban áttekintem, hogy milyen módon jelennek meg a Yohannis és munkatársai által [11]-ben már megfogalmazott pontok, amelyek alapként szolgálnak, hogy milyen elemek, gondolatok szükségesek ahhoz, hogy egy játékos eszközről beszélhessünk:

- Játékosok. A tanulók, az eszköz felhasználói, akiknek jelen esetben a feladata a négyzetek sorba rendezése.
- Környezet. Az eszköz adja meg a környezetet, amelyen keresztül sorba lehet rendezni a négyzeteket, illetve valamiféle információt, visszajelzést is ad (helyes/helytelen lépés, állapot mutatása, eltelt idő).
- Szabályok. Az adott rendezési algoritmus követése, amelyet a felhasználóknak be kell tartania, ahhoz, hogy teljesíthessék a pályákat/játékot. Illetve megfelelő „élettel" (hibázási lehetőséggel) rendelkezhetnek egy-egy feladat megoldása alatt. Ha ez idő/lehetőség alatt nem sikerül, akkor újra kell kezdeni a felhasználónak a pályát.

- Kihívások. Kihívásnak vehetjük az előbb említett szabályok betartását. Tehát, ebben az esetben a négyzetek megfelelő helyére húzását az adott lépéskorláton belül (algoritmus működésének/lépéseinek megfelelően).
- Interakció. Az alkalmazásban megjelennek az algoritmus főbb elemei (algoritmus egy leíró nyelven, adat, reprezentáció), amelyek a játékosok cselekedetei, interakciói révén módosulnak Jelen esetben az adott rendezési algoritmus lépése szerint kell a diákoknak a megfelelő elemet kicserélniük, áthúzniuk a megfelelő helyre, amiről aztán visszajelzést is kapnak (egy felvillanó piros/zöld színnel).
- Célok. A cél, hogy sikeresen végig kövesse egy rendezési algoritmus lépéseit a megfelelő hibázási lehetőséggel, ezáltal elsajátítsa az algoritmus működését, majd azt saját maga is fel tudja használni, le tudja programozni különböző feladatokban.
- Érzelmi tapasztalok. A kihívás, hogy teljesítse a rendezést a megfelelő szabályok szerint az adott környezetben növeli a diák motivációját. Ha nem sikerült egy feladat, akkor az felkeltheti az érdeklődését, plusz erőt adhat arra, hogy újra próbálkozzon. Míg siker esetén örömet, elégedettséget érezhet, ezután kihívást kereshet a következő, nehezebb pályákban.
- Számszerűsíthető kimenet. Ilyenfajta kimenet lehet egy-egy jutalom, pontszám, pecsét, amelyeket saját teljesítményük alapján kapnak. Például: sikerül hamarabb/kevesebb élet felhasználásával teljesíteni a rendezést. Ezek alapján látjuk, hogy hol tartanak, illetve felkészültek-e nehezebb feladatok megoldására.
- Lehetséges következmények. Végtelen számú újra próbálkozási lehetőség van, így lehetőségük nyílik a tanulóknak, hogy felfedezzék, megértsék az algoritmus működését.

2.5.2. Hatékonyságbeli eredmény

Egy egyetemen megvizsgálták a játékos vizualizációk és a statikus vizualizációk hatékonyságbeli összehasonlítását [10]. Két csoport volt: az elsőnél először hagyományos módon tanítottak a rendezési algoritmusokat (jelen esetben a buborékos, beszúró és a kiválasztó rendezéses algoritmusokat), a másodiknál játékos vizualizációval tették ezt meg. Ezután egy tesztet töltettek ki, melyben egy-egy számsorozatot kellett rendezni a tanult rendezési algoritmusokkal. A teszt befejezte után fordult a szerep, és a csoportok a másik csoport eszközeit használták fel a tanuláshoz. Végül az előző teszthez hasonló feladatsor zárta a kutatást.

Csoport	**Könyv-Első**			**Játék-Első**			**Különbség**	
Típus	**1. Teszt**	**2. Teszt**	**Kül.**	**1. Teszt**	**2. Teszt**	**Kül.**	**1. Teszt**	**2. Teszt**
Buborékos	8.71	10.00	1.29	8.93	10.00	1.07	0.22	0.00
Beszúró	7.31	10.00	2.69	9.64	10.00	0.36	2.33	0.00
Kiválasztó	7.42	10.00	2.58	10.00	10.00	0.00	2.58	0.00
Összeségében	7.81	10.00	2.19	9.52	10.00	0.48	1.71	0.00

2. táblázat: Rendezéses tesztek eredményei az egyes részek végén [10]

Az első teszt végeredménye alapján az látszik, hogy akik először a játékos eszközöket használták jobb eredményeket értek el, mint, akik a hagyományos eszközöket használták. A második teszt végeredményéből csak annyit szűrhetünk le, hogy a végére mindenkinek sikerült elsajátítani a kiválasztott rendezéses algoritmusok működését.

3. Összegzés

Igyekeztem bemutatni minél többféle módszert arra, hogy hogyan lehetne minél sokszínűbb, minél hatékonyabb oktatási segédeszközként felhasználni az algoritmus vizualizációt. Megnézve az interaktív, játékos vizualizációk alapköveit, illetve egy-egy példán keresztül ezek, hogyan is valósulnak meg. Láttuk, hogy a megfelelő módszerekkel a megfelelő eszközöket használva kimutatható hatékonyságbeli eredményt érhetünk el az algoritmusoktatásban. Azt már előző cikkemben is kifejtettem, hogy minél interaktívabb egy AV, annál hatékonyabb lehet egy eszköz a tanulásban. Láthattuk az idő haladtával egyre interaktívabbakká váltak az eszközök (egyre több ehhez kapcsolódó komponens tartalmaztak, illetve a Naps által meghatározott szintekből is egyre több jelent meg bennük), ezáltal kimondhatjuk azt is, hogy tanuláshatékonyságban is javult, fejlődött a módszer. Most ezt kiegészítve láthatjuk, hogy ha a diákoknak a tudást csak rejtve, játékba ágyazva próbáljuk átadni, akkor még inkább hasznos eszközzé válhat. A tanulók felfedezni vágyásból, illetve a játék örömeért ássák bele magukat egyre jobban az eszköz megértésébe, annak felhasználásába, és így az algoritmus megértésébe is. Érdemes minél többfajta AV-t bevinni az osztályba (persze csoporttól függően), így az algoritmusoktatás érdekessé, látványossá, változatossá válhat.

Terveim között szerepel, hogy a jövőben én magam is készítek egy webes játékos algoritmus vizualizációs eszközt, amelyen keresztül megmérném az eszköz tanuláshatékonyságát. Ezt az eszközt úgy valósítanám meg, hogy több feladat/pálya is legyen különböző algoritmusokhoz, a játékosok eredményei utólag is visszanézhetők, valamint a játékosok/tanulók is folyamatosan nyomon követhetik az előrehaladásukat, fejlődésüket. Ezt az ELTE Informatika Karán a „Programozás” című tárgy keretein belül tenném meg.

Köszönetnyilvánítás

EFOP-3.6.1-16-2016-00023: Kutatás-fejlesztési tevékenység megvalósítása az Eötvös Loránd Tudományegyetem szombathelyi kampuszán – A projekt a Magyar Állam és az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Irodalom

1. R. Baecker: *Sorting out Sorting: A Case Study of Software Visualization for Teaching Computer Science*. Software Visualization: Programming as a Multimedia Experience, pp369-381, 1998.
2. I. Bende: *Algoritmusok lépésenként*. ELTE, 2017.
3. Ville Karavirta: *Facilitating Algoritm Vizualization Creation and Adoption in Education*. Helsinki University of Technology, 2009.
4. Niko Myller, Roman Bednarik, Erkki Sutinen, Morechai Ben-Ari: *Extending the engagement taxonomy: Software visualization and collaborative learning*. ACM Transactions on Computing Education, New York, USA, 2009.
5. L. Naps, G. Rössling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, J. Á. Velázquez-Iturbide: *Exploring the role of visualization and engagement in computer science education*. ITiCSE on Innovation and Technology in Computer Science Education (ITiCSE'02). ACM Press, pp131–152, 2002.
6. R. Pinker: *How do students learn from models?*. Concord Consortium Newsletter 11, pp14-15, 2007.
7. U. Ritterfield, M. Cody, P. Vorderer: *Serious Games: Mechanism and Effects*. Routledge, London, 2009.
8. S. Shabanah, J. X. Chen: *Symplifying algorithm learning using serious games*. Proceedings of the 14th Western Canadian Conference on Comuting Education, 2009.
9. Clifford A. Shaffer, Arpit Kumar, Mayank Agarwal, Alexander Joel D. Alon, Stephen H. Edwards: *Going Beyond Algorithm Visualization to Algorithm Exploration*. Virginia Tech, 2009.
10. Alfa R. Yohannis, Yulius D. Prabowo: *Sort Attack: Visualization and Gamification of Sorting Algorithm Learning*. VS-GAMES, pp1-8, 2015.
11. Alfa R. Yohannis, Y. D. Prabowo, A. Waworuntu: *Defining Gamification: From lexical meaning and process viewpoint towards a gameful reality*. International Conference on Information Technology System and Innovation, Bali, 2014.