Genetikus algoritmusok

Tanulmány az Adatbázisok haladóknak c. tárgyhoz

Zsolnai Károly

2. évf. MSc Mérnök informatikus szak

2012/2013 tanév I. félév

Tartalomjegyzék

Bevezetés

A jelen dolgozat elkészítésének fő célja, hogy a Genetikus Algoritmusok c. előadásához készült diasor [1] érthetőségét segítse. Ez alapján a tanulmány nem önmagában, hanem a diasorral együtt való tanulmányozása során nyer értelmet.

Előkövetelmények

A genetikus algoritmusok és az adatbáziskezelési esettanulmány témaköreinek kibontása során az alábbi ismeretekre lesz szükség:

- adatbázisok – heap szervezés, indexelt szervezés, többszintű ritka indexek, B*-fa [2],
- a legelemibb matematikai és lineáris algebrai ismeretek – vektor, halmaz jelölések, permutációk, euklideszi távolság, lineáris kombináció, véges sorok fogalmának ismerete,
- a legalapvetőbb algoritmuselméleti fogalmak ismerete, mint az ordó jelölés (Big O notation) az algoritmusok futásideje vagy tárigényének felső becslésére.

Genetikus algoritmusok

Definíció és felhasználási területek

A genetikus algoritmusokat elsősorban optimalizálási feladatok megoldására használjuk, ahol egy általában folytonos függvénnyel adott problématérben, bizonyos megkötések és feltételek megadása mellett keressük ennek a függvényének a maximumát, vagy minimumát. A diasorban a hegymászó algoritmus működését az egyszerűség kedvéért egy gráfon mutattuk be, de a fentiek alapján a gráf helyébe nyugodtan tetszőleges folytonos függvényeket is képzelhetünk.

A genetikus algoritmusok érdekessége és természete, hogy a kiadott optimalizálási feladatra nem egy megoldást, hanem a megoldások egy populációját állítják elő, amelyeket különböző keresztezési, mutációs és szelekciós operátorokkal kombinálja össze annak reményében, hogy ezek során egy jobb megoldásra tesz szert.

Elsősorban „nehéz” problémákra alkalmazzuk, ahol a nehézséget az alábbi szempontok alapján definiáljuk:
- Sokdimenziós keresési terek jellemzők és a fontosabb változók közötti összefüggés ismeretlen,
- A tradicionális megoldások elfogadhatatlan futási időt adnak (NP-nehéz, NP-teljes problémák),
- A keresési tér nehezen, vagy nem is szűkíthető,
- Egy megoldás jósága gyorsan ellenőrizhető, de egy jó megoldás előállítása nehézkes.

Mivel az algoritmusok ezen családját optimalizálási feladatok megoldására alkalmazhatjuk, így a mérnöki, fizikai és közgazdasági területen felmerülő problémák igen széles skálájára képesek megoldást adni, ilyen például a konvex lencsék tervezése, adott erőhatásoknak és időjárási körülményeknek ellenálló, legkisebb költségű tartószerkezetek tervezése, számítógépes mesterséges intelligencia, legkisebb összsúlyú Hamilton-kör keresése gráfban, vagy éppen a számítástudomány területén megismert, egyéb exponenciális futásidejű NP-nehéz problémák megoldása.

A fitness függvény fogalma

Az algoritmus futásának lelke a fitness függvény működése. Ez mondja meg, hogy egy megoldásjelölt (kromoszóma) mennyire tekinthető optimálisnak. Az evolúció fogalmát úgy definiáljuk, hogy a fittebb („erősebb”) egyedek örökítik tovább a génjeiket, míg a gyenge egyedek kiszelektálódnak. Az algoritmus futásidejének a túlnyomó részét a fitness függvény kiértékelése tölti ki, azaz egy kromoszómához való skalár érték rendelése, így arra kell törekednünk, hogy a lehető legegyszerűbben számítható módon adjuk meg azt.

A probléma formalizálása a genetikus algoritmusok nyelvén

A kromoszóma génjeinek kódolása lehet egész, tört, pozitív, negatív, vagy éppen bináris értékű, de tetszőleges, a feladat szempontjából releváns, értelmezhető reprezentációs alkalmazhatunk, mint

például nyelvtani feladat megoldásakor akár betűket, szavakat, vagy egy labirintusban bolyongó program mozgási és fordulási irányait. Mindezen értékek típusa akár génről-génre változhat. A diasorban a 0-1 hátizsák probléma egy genetikus programjának definíciójában a kromoszómákat a tárgyak számával azonosan hosszú bináris vektorral reprezentáljuk, amely vektor az adott indexen 1 értéket vesz fel, ha az adott tárgy szerepel a hátizsákban, illetve 0, amennyiben nem. A fitness függvény a hátizsákba tett tárgyak érték-összege, a megkötés pedig az, hogy a befogadóképességével adott hátizsákot ne pakoljuk túl.
A Hamilton-kör probléma megoldása lehet egy permutáció, míg a fenti labirintusos problémára egy kromoszóma elemei lehetnek a haladási irányok, és az elfordulási szögek sorrendje.

Evolúciós operátorok

A meglévő kromoszómáinkból az evolúciós folyamat szerint különböző sémák szerint keresztezhetjük, akár egy, vagy két ponton, illetve véletlen és aritmetikai keresztezéssel is. Egy gén mutációja bináris esetben az invertálás operátorával adott, valós esetben lehet akár az adott génhez való tetszőleges véletlen érték hozzáadása vagy kivonása.
Amennyiben egy már meglévő populációból új egyedeket állítunk elő, és szeretnénk elvárni, hogy a jó tulajdonságú egyedek génjei öröklődjenek tovább, akkor az is a feladatunk, hogy a gyenge egyedeket eltávolítsuk a populációból. Erre a szelekciós operátorok adnak lehetőséget. Az előadás során a fitness-arányos szelekciót tárgyaltuk, ahol az egyes kromoszómák túlélési valószínűségét a fitness-értékeik arányában adjuk meg, illetve az elitista szelekcióról is esett szó, mely a a populáció egyedeit fitness-szerinti sorrendbe rendezi, és az első tetszőleges számú helyezett utáni kromoszómákat elpusztítja.

Az algoritmus futása

A következő módon lehet lefuttatni egy generációt:

- kezdeti populáció inicializálása,
- fitness-függvények kiértékelése a kezdeti populáció tagjaira,
- szelekció,
- keresztezés-mutáció,
- az új egyedek fitness értékeinek kiszámítása,
- az új egyedek elhelyezése az új populációba.

A fenti műveletsor egy generáció lejátszását jelenti, így a fentieket futtatjuk, amíg az alábbi, tetszőleges leállási feltétel be nem következik:

- Generáció limit – azaz egy felső határ, hogy meddig futhat az algoritmus,
- Max fitness limit – egy érték, amely megoldásról már úgy véljük, hogy elégedettek lehetünk vele,
- Átlag fitness limit – ha több megoldásra van szükség, akkor a populáció átlag fitnessét is figyelhetjük,
- Konvergencia beállása – ha úgy vesszük észre, már sok generáció óta nem állt be javulás.

Adatbáziskezelési esettanulmány

Az előadásanyagban egy olyan esettanulmányt vizsgáltunk meg, ahol egy adatbázis tervezése során olyan indexelést szeretnénk alkalmazni, ami egyaránt minimalizálja az adatbázisból való olvasás és írás sebességét. A megoldásban két fitness függvényt alkalmaztunk, az egyikkel a lekérdezések, a másikkal pedig az olvasások futásidejét kívánjuk optimalizálni. A kettőből egy összetett (kompozit) függvényt csináltunk, ahol a két kritérium egy tetszőleges súlyozott összegével finomhangolhatjuk, hogy az olvasás és az írás milyen fontos szerepet tölt be az adatbázis működése során.

Egyéb alkalmazások

Az előadás során egy intuitív példát is mutattunk egy teljes generáció lefuttatására, illetve párhuzamos implementációkat vizsgáltunk a 0-1 hátizsák problémára [3], illetve a Roger Alsing-féle Mona Lisa festés problémájára is [4].

Összefoglalás

A fentiek alapján a genetikus algoritmusokról az alábbiakat mondhatjuk el:

- Nagy részük probléma- és reprezentációfüggetlen,
- Az átlagos fitness érték az optimumhoz konvergálnak,
- A konvergálás kapcsán már útközben látható a pillanatnyi eredmény,
- Tetszőlegesen nagy dimenziójú problémákra alkalmazhatók,
- Mutációval kiszabadulhatnak a lokális optimumhelyek rabságából,
- Más gyors, de nem optimális keresési algoritmusokkal jól keverhetők,
- Kiválóan párhuzamosíthatók, alacsony a kommunikációs bonyolultságuk,
- Tárigényük a populáció méretével és hosszával lineárisan arányos.

A genetikus algoritmusok családja implementációs szinten is kívánatos tulajdonságokkal bír, egy újonnan felmerült, nehéz optimalizálási feladatot rövid idő alatt formalizálni, és implementálni lehet, akár párhuzamos számítási architektúrákon is.

Irodalomjegyzék

[1] Zsolnai Károly – Genetikus algoritmusok, *2012*
http://cg.iit.bme.hu/~zsolnai/gfx/genetic/zsolnai_genetikus_adatb_haladoknak.pdf

[2] Gajdos Sándor – Adatbázisok, *Műegyetemi kiadó, 2001.*

[3] Zsolnai Károly – Párhuzamos implementációk a 0-1 hátizsák, és a „Mona Lisa” problémára, *2012*
http://cg.iit.bme.hu/~zsolnai/

[4] Roger Alsing - Genetic Programming: Evolution of Mona Lisa
http://rogeralsing.com/2008/12/07/genetic-programming-evolution-of-mona-lisa/