Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Villamos Energetika Tanszék

Evolúciós algoritmusok alkalmazása a középfeszültségű hálózat üzem helyreállítás segítésére

TDK DOLGOZAT

Készítette
Csatár János

Konzulens
dr. Dán András
Gaál Róbert

2011. október 27.

Tartalomjegyzék

Kivonat

A villamosenergia szolgáltatás minősége napjainkra egyre fontosabb jellemzővé válik. A fogyasztás növekedésével a hálózat is növekszik. Egyre nehezebb a hálózatok karbantartása és tervezése úgy, hogy emellett a minőségi mutatók is megfelelőek maradjanak. A MEH mutatókat a villamos energia szolgáltatóknak bizonyos határokon belül kell tartaniuk, ami nagy feladatot jelent számukra.

A kiesések egyik oka a középfeszültségű hálózaton bekövetkezett üzemzavar. A hibák azonosítása, megtalálása és kijavítása bonyolult folyamat, melyben várhatóan számítógépes tanácsadó rendszerekkel jelentős hatékonyság növekedés érhető el. A TDK dolgozat ennek a területnek egy részével foglalkozik: Üzemzavar során gyakran van szükség kapcsolásokra, például a zárlat megkereséséhez egy hálózatrész elkülönítéséhez, vagy a már megtalált zárlat elkülönítéséhez. Eközben a lehető legtöbb fogyasztó ellátásáról kell gondoskodni, valamint a lehető legrövidebb kiesésre kell törekedni. Ez egy-egy kisebb hálózaton, ahol nincs túl sok keresztág és kapcsoló nem tűnik nehéz feladatnak, ám ha bonyolultabb hálózattal áll szemben a diszpécser, vagy egyszerre több körzetet is felügyelnie kell, már nehezebb átlátni a lehetőségeket. Gyakran nagy nyomás nehezedik a diszpécserekre.

A dolgozatban bemutatásra kerül egy új, genetikus algoritmuson alapuló eljárás, amivel a kapcsolások helyét és sorrendjét lehet meghatározni.

Cél a minél több fogyasztó ellátása, kevés kapcsolással, rövid időn belül. Lehetőség van a kapcsolók bizonyos állapotban történő rögzítésére, biztosítva, hogy a hibás hálózatrész mindenképpen elkülönítve maradjon, valamint az olyan esetek kezelésére ahol esetleg hibás működés folytán nem lehet átkapcsolni egyet (vagy többet is akár). Az algoritmus a megoldás során figyelembe veszi a vezetékek és transzformátorok terhelhetőségét valamint a feszültséget a szabványban definiált határokon belül kell tartania mindenhol. Természetesen szükség van load-flow számításra ezekhez (ami lehetőleg gyors, hiszen a futás során rengetegszer kell számítani), ezért a hálózat paramétereinek ismerete is fontos. Szerencsére ez a szolgáltatóknál ma már általában megtalálható digitális formában. Ezenkívül szükség van még a fogyasztások nagyságára, ez jól becsülhető a leágazás feszültsége és árama alapján, ami szintén rendelkezésre áll legtöbbször valós időben. A kapcsolók állásait is nyilvántartják. Tehát az eljárás nem igényli újabb eszközök telepítését a hálózaton. Természetesen abban az esetben ha extra információk állnak rendelkezésre, felhasználásukkal pontosítható a számítás és így az eredmények is. A megoldási folyamat két részre van bontva, az elsőben a kívánatos topológia kialakítása történik, a második részbe az optimális kapcsolási sorrend meghatározása került.

Az első rész egy több célú, feltételekkel határolt feladat, megoldására egy NSGA-II-őn alapuló genetikus algoritmust dolgoztam ki.

A második rész egy egy célú és feltétellel határolt feladat, megoldására egy dynamic stochastic ranking szelekcióval kiegészített genetikus algoritmust dolgoztam ki.

A tesztek során kapott megoldások megbízható, jó működést mutatnak. Az eredmények felhasználhatók hálózat tervezésnél (például távműködtethető kapcsolók optimális elhelyezése), vagy tréningszimulátorban való alkalmazásra (például a diszpécserek képzésénél a véletlenszerű helyzetekben nyújtott önálló teljesítményük értékeléséhez nyerhetők közel objektív mutatók), esetleg később egy valódi tanácsadó rendszer készítésére.

Abstract

The quality of power supply is more and more important in today's industry. The distribution network has to keep up with the growing consumption. This requires increasing effort in design and maintenance to provide quality power supply.

In many cases the reason of supply disturbance is caused by faults on medium voltage networks. Recognizing the problems, then locating and repairing it is often a complicated task. With decision support by intelligent systems much improvement can be achieved.

This work focuses on one aspect of this problem: Throughout system restoration many switching has to be done (eg. to separate a certain part of the network for trials). Meanwhile as many consumer as possible have to be energized with the shortest interruption. In a small network with few switches and tie lines this doesn't look like a challenging task, but in a more complex network it poses difficulties.

A new genetic algorithm based method is introduced in this article to address the issues mentioned above, which can determine the necessary switching operations.

The objective is to supply the maximum number of consumers, with few switching operations and with short interruption. Switches can be locked in case that certain switches can not be operated, or are required to stay in opened or closed position. Some restriction has to be considered: Over-current on lines and transformers has to be avoided and the voltage level need to be in the regulation defined threshold. For this reason a load-flow calculation is necessary, which is preferably fast and simple, because it needs to be calculated many times. Fortunately the parameters needed for the load-flow are available at the utilities here in digital format. With the help of real time measurements on the HV/MV transformers, consumption can be estimated. Only this and the position of the switches has to be online available at starting of the algorithm (these are also available today). Thus new equipment installations on the network are not necessary. Of course if extra information can be used, it improves the quality of the results.

The solving process is separated in two parts. Thereby the complexity of the problem is reduced to a more manageable level. Firstly the desirable network topology is determined, secondly the order of the switching operations is decided. The first part is a multi-objective constrained problem, for which a genetic algorithm based on NSGA-II is developed.

The second part is a single-objective constrained problem. For which a genetic algorithm with dynamic stochastic ranking is developed.

Through testing it is observed that the method provides good solutions reliably. The results can be applied to

- computer assisted decision making,
- network design (eg. Remote controllable switch placement),
- training simulators (eg. At staff trainings, we can have objective indicator of their performance in random situations).

Bevezető

A villamos energia szolgáltatás minősége napjainkra egyre fontosabb jellemzővé válik. A fogyasztás növekedésével, a hálózat is növekszik. Egyre nehezebb a hálózatok karbantartása és tervezése úgy, hogy emellett a minőségi mutatók is megfelelőek maradjanak. Szabványos mutatók alapján értékeli a MEH (Magyar Energia Hivatal) a szolgáltatók tevékenységét. A MEH mutatókat a villamos energia szolgáltatóknak bizonyos határokon belül kell tartaniuk, ez nagy feladatot jelent számukra (például nemrég kötelezték árcsökkentésre az EON-t, mivel nem tudta betartani a követelményeket[1]). A kimaradások egy jelentős részéért a középfeszültségen bekövetkezett üzemzavarok felelősek.

Ezért a középfeszültségű hálózat üzemzavar elhárítás hatékonyságának növelése alapvető igény a szolgáltatóknál. Ennek egyik része a diszpécserek tréningezése, képzése, ám ez egy határon túl nem ad jelentős javulást.

További javulás érhető el a telemechanizáltság növelésével, így a távolabbi, vagy gyakrabban használt kapcsolók rövid idő alatt működtethetők, valamint a távleolvasott szenzorok jelzései hamar megérkeznek a diszpécserekhez, hamarabb elkezdődhet a munka. Ilyen eszközök hiányában, pusztán a beérkezett fogyasztói visszajelzésekre támaszkodva, a rendszer aktuális állapota kevéssé ismert, ami hátráltatja a gyors reakciót és a kapcsolásokhoz is hosszas utazgatások szükségesek.

Tehát a hatékonyság növelését egyrészről a hálózat aktuális üzemállapotára vonatkozó tudásunk bővítésével (zárlatjelzők, alállomási mérések, esetleg később okos mérők jelzései stb...), a távműködtetésű beavatkozó eszközök számának növelésével, másrészről az intézkedések optimalizálásával érhetjük el.

Ez utóbbi (amennyiben kizárólag ember által végzett) nagy helyismeretet és tapasztalatot igényel és sok egyéb szubjektív tényezőn is múlik. Például üzemzavarok esetén gyakran nagy a nyomás a személyzeten, hogy mihamarabb visszatérhessenek normál üzemállapotra. Ilyen körülmények között higgadtnak maradni és minden szempontot figyelembe venni, mérlegelni nem könnyű. A döntéseket segíthetjük számítógépes tanácsadással, vagy akár teljesen automatizálhatjuk is. Így többminden figyelembe vehető, és nem függ a megoldás minősége a személyzet figyelmetlenségeitől, tudásától. Ezenkívül kevesebb diszpécserrel is megoldható a rendszer felügyelete, így gazdasági szempontok alapján is előnyös ezen rendszerek használata.

Az iparban nálunk jellemzően a diszpécserek magukra vannak utalva döntés hozatalkor a szükséges intézkedések sorozatát illetően, a SCADA rendszer annyiban segíti őket, hogy

[1] `http://www.eh.gov.hu/gcpdocs/201107/sajtokozlemeny_2011_julius_22.pdf`

látják a hálózatot és a távolról is leolvasható, működtethető eszközökhöz nyújt hozzáférést, segíti a munkájuk rendezését.

A fejlett funkciók nincsenek a gyakorlatban alkalmazva. Fejlesztések folynak ez irányban is természetesen. Elterjedt a Siemens Spectrum SCADA rendszer az üzemirányításban, ennek újabb verzióiban már van lehetőség szakértői rendszeren alapuló tanácsadásra. Egyelőre ennek bevezetésére készülnek.

A dolgozat felépítése: A következő fejezetben kicsit pontosabban szemügyre veszem a megoldandó problémát, az azok megoldására alkalmas módszereket, néhány más kutatási eredményt is röviden elemzek.

Az ezt követő fejezetben részletesebben bemutatom, hogy a megoldásra milyen módszert dolgoztam ki.

Végül az utolsó fejezetben röviden értékelem az eredményeket és az alkalmazási, fejlesztési lehetőségekkel is foglalkozom.

1. fejezet

A probléma bemutatása

A feladat a beérkezett jelzések alapján az üzemzavar helyének meghatározása (például zárlat vagy szakadás), valamint ezek ismeretében az üzemhelyreállítás megkezdése, elhárítása. Több lépésre bontható a folyamat:

- Első lépésben fel kell ismerni és meg kell találni a hibát,
- Ezt követően el kell különíteni,
- Mindvégig biztosítani kell a lehető legtöbb fogyasztó ellátását,
- Majd végül a hiba elhárítása és a normál üzemállapothoz történő visszatérés következik.

Eközben sok kapcsolásra van szükség.

1.1. Lehetőségek az üzemzavar elhárítás gyorsítására

A hiba automatikus, vagy támogató rendszerrel segített hatékony felderítéséhez sok szenzorra van szükség. Minél több távleolvasott szenzor, távműködtetésű kapcsoló és egyéb kiegészítő eszköz működik a hálózaton, annál jobban vehetők igénybe automatikus módszerek. Sajnos ezeken a területeken jelenleg a középfeszültségű és főleg a kisfeszültségű hálózat jelentős lemaradásban van az átviteli hálózathoz képest, ezért jelenleg még nagyon kevés valósidejű információ áll rendelkezésünkre a hálózatot illetően. Ez a smart metering terjedésével és egyéb smart grid fejlesztésekkel a jövőben megváltozhat. Ezen részfeladat gyorsítása főként a hálózaton elhelyezett új eszközökkel érhető el. Szoftveres támogatás ezután következhet csak.

A javítási munkálatok sokszor lassúak és nincs sok lehetőség a meggyorsításukra számítógépes tanácsadással.

A legnagyobb hasznot az üzemzavar következtében kialakult és a munkálatok miatti kiesések (áramszünet) minimalizálása hozhatja. Sajnos a legtöbb üzemzavar esetében lesznek olyan fogyasztók, akik ellátás nélkül maradnak egy időre, de ez az idő, valamint az érintettek köre csökkenthető. Például egy hibát úgy is lehet kezelni, hogy a munkálatok idejére teljesen lekapcsolják a leágazást az alállomásban -ezzel rengeteg fogyasztó marad

áram nélkül, vagy csak a hiba helyét különítik el és ezáltal lényegesen kevesebbeknek kell áramszünetet elviselniük. Ez persze egy szélsőséges eset, de sokszor nehéz átlátni, hogy honnan lehet ellátni azokat a területeket amelyek az üzemzavar következtében nem érhetők el a normál úton. Például milyen feszültségviszonyok alakulnak ki, ha egy szomszédos leágazásból (ami esetleg másik alállomásból van táplálva) történik az ellátás, a szabványos értékeken belül marad-e mindenhol.

Meg kell tehát találni azt a kapcsolási műveletsorozatot amellyel a hiba könnyen felderíthető és elkülöníthető a javítások idejére, de közben törekedni kell a fogyasztók zavartalan ellátására, és ahol ez nem lehetséges, ott minél kevesebbeket érintsen. Minimalizálni kell azt az időt amíg az egyes fogyasztók áram nélkül maradnak.

A kapcsolások és sorrendjük meghatározásának számítógépes támogatása szerencsére nem igényli újabb eszközök telepítését a hálózaton. Természetesen a hatékonyság itt is javul extra információk rendelkezésre állásával.

A dolgozat további része ezzel a területtel foglalkozik.

1.2. Kapcsolások kérdése

Ha már a hibák keresése során és végig az üzemhelyreállítás folyamán segítjük a diszpécsereket, hogy a szükséges kapcsolásokat a legkevesebb fogyasztót érintve tudják elvégezni, akkor ritkábban, rövidebb ideig és kevesebbeket fog érinteni az áramkimaradás.

Alapvetően az a kérdés, hogy melyik kapcsolókat működtessük és milyen sorrendben. Sajnos a probléma optimális megoldása nem lehetséges determinisztikusan, mivel a lehetséges állapotok számának exponenciális függvénye a kapcsolók számának. Szerencsére sok megoldás létezik egy-egy esetre és megfelelő közülük egy *elég jó* is (nincs nagy eltérés az optimálistól), viszont további nehézség, hogy két közel egyformán jó megoldás lehet, hogy csak egy vagy két kapcsolásban egyezik meg egymással. Az *elég jó* megoldások közül egynek a megtalálása már könnyebb feladat.

Több irányban is folynak nemzetközi kutatások. Nagyrészük a következő elvek közül épít egyre vagy többre:

- heurisztikus módszerek
- szakértői rendszerek,
- neurális hálózatok,
- fuzzy rendszerek,
- evolúciós algoritmusok.

Minden területnek megvan a maga előnye és hátránya, sokszor kombinálhatók is a különböző módszerek.

Heurisztikus módszereknél nem valószínű az optimális megoldás megtalálása és nem garantált a megoldás használhatósága (túlterhelés, túl nagy feszültségesés előfordulhat), viszont gyorsabbak a többi módszernél.

Szakértői rendszerek esetében (amelyeket gyakran kombinálnak fuzzy eljárásokkal) nagyfokú specializáltság jelentkezik az adott hálózati jellemzőkre, nem alkalmazható minden kritérium benne és csak annyira jó, amennyire a megalkotott szabályok helytállnak az adott hálózatra, viszont minden futásra ugyanazt az eredményt adja, ami ráadásul nagy valószínűséggel használható is lesz. További előnye, hogy a működése közelebb áll az emberek gondolkodásához.

Neurális hálózatok esetén szintén a vizsgált hálózattól fog függeni a kész algoritmus, ráadásul minden változtatásnál újra kell tanítani és sajnos ez közel akkora feladat mint aminek a megoldására szánjuk; a kínált megoldás nem biztos hogy megvalósítható és a hiba okát sem tudjuk egyszerűen kiszűrni. Viszont ha egyszer működik, akkor gyors tud lenni és rejtett összefüggéseket is *megsejthet*. Vagyis olyan szabályszerűségek segítségével juthat jó megoldásokhoz, amikről nem is tudunk.

Fuzzy rendszerek nem használhatók jól magukban a problémára, inkább valamelyik másik módszer kiegészítéseként alkalmazzák.

Evolúciós algoritmusok esetén nem biztos, hogy minden futásra ugyanazt az eredményt kapjuk, ráadásul sok múlik a paraméterezésen, viszont könnyen alakítható, univerzálisabb, mivel a megadott célok alapján vizsgálja a jelölt hálózatokat, így nincs szükség különös tudásra a hálózat működését, kialakítását illetően. Természetesen amennyiben rendelkezünk információkkal, azokat az algoritmusba építve javíthatjuk hatékonyságát.

A lehetőségeket mérlegelve, a választás az evolúciós algoritmusokra, azon belül is a genetikus algoritmusokra esett.

1.3. Néhány megoldási módszer rövid elemzése

Sok kutatás létezik. Ezek közül csak néhányat, egy-egy részterületre mutatok be, lehetőleg friss eredményt. Sokan nem foglalkoznak a kapcsolások sorrendjével és sokan csak egy-egy részfeladatra adnak megoldást, például csak a NF/KöF transzformátor kiesésére, vagy csak a szomszédokba való átcsoportosításra stb...

Íme néhány átfogóbb megoldást kínáló módszer:

Fuzzy Petri-háló [1](2002): Fuzzy és szakértői rendszer kombinációja, melynek megvalósításához Petri-hálókat használnak a szerzők. A szabályok értelmezése ennek következtében párhuzamosan folyik és a bemeneti adatok bizonytalanságának kezelése is megoldott, a szabályok ábrázolása lehetséges grafikusan. Viszont sajnos a szakértői rendszerekre jellemző hátránnyal rendelkezik: nagyon sok múlik a megalkotott szabályokon.

Módosított Prim-algoritmus [2](2011): Ez egy heurisztikus módszer, a Prim-algoritmus módosításán alapul. Legrövidebb feszítő fákat keres a hálózaton, vagyis erdőt készít. Ennek következtében a módszer használhatósága erősen függ az egyes kapcsolókat jelképező élek súlyozásától, hiszen ebbe az egy paraméterbe kell a lehető legtöbb szempontot bele sűríteni. Erre javasol egy sémát is a szerző. Megoldás közben nem kezeli a hálózati feltételeket (túlterhelések, feszültségesések ...), ezért a kínált megol-

dás nem biztos, hogy használható lesz.

Neurális háló [3](2011): RBF neurális hálózattal megvalósított eljárás, mely fuzzy adatokkal dolgozik. A neurális hálózatot több módszerrel is optimalizálták, ebből a leghatékonyabb egy módosított differenciál evolúciós algoritmus volt. Nagy hangsúlyt kapott a betanítási adathalmaz is. Ebből is látszik az eljárás bonyolultsága, és még ezzel sem garantált minden körülmények között a jó működése.

Genetikus algoritmus [4](2009): Bináris génkódolással, egy célfüggvénnyel rendelkező genetikus algoritmus. A feltétel sértések büntetésként vannak implementálva, a kezdeti generációba heurisztikus módszerekkel kialakított egyedek kerülnek. Megvalósításra került a mutáció és keresztezés valószínűségére adaptív szabályzás is (a generáció fitneszértékein alapulva), ennek köszönhetően gyorsabb a konvergencia. Viszont sajnos az egyetlen fitneszérték (célfüggvény) miatt kevésbé könnyen kezelhető a különböző célok és feltételek összeegyeztethetősége.

Látható, hogy jelenleg is aktív kutatások folynak a témában, több irányból megközelítve a problémát. Nincsen általánosan elfogadott, legjobb módszer.

1.4. A genetikus algoritmusokról röviden

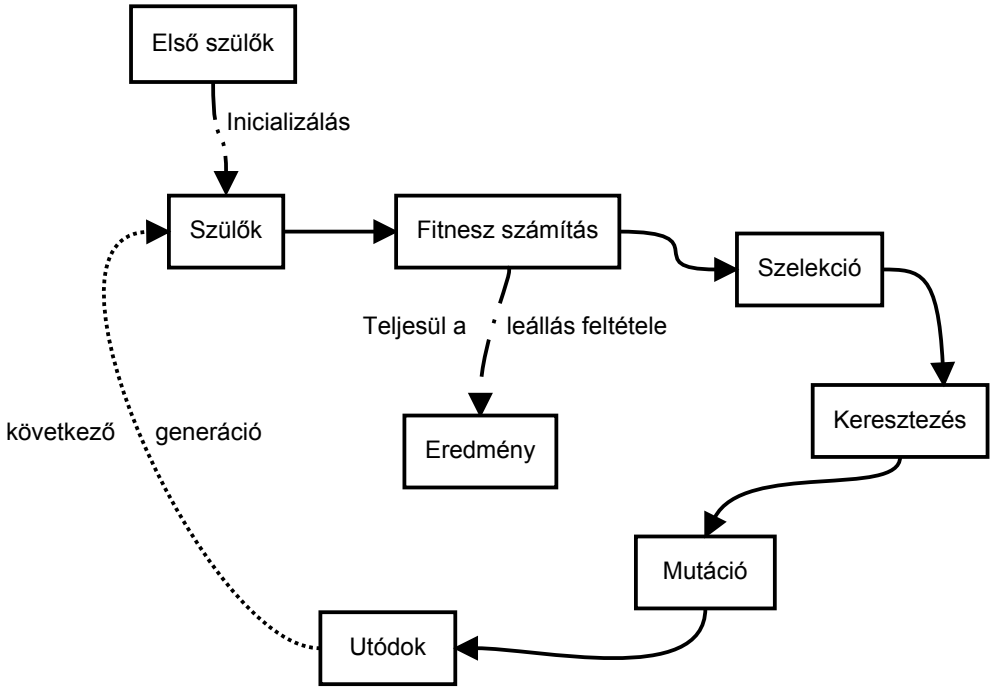
A genetikus algoritmusok a lágy számítási módszerek és azon belül is az evolúciós algoritmusok közé sorolhatók. Goldberg könyve [5] az egyik alapmű e téren, egy másik hasznos könyv [6], egy összefoglaló az evolúciós algoritmusokról, az újabb eredményeket is bemutatja.

Az evolúciós algoritmusok lényege, hogy megpróbálják a természetben meglévő törvényszerűségeket hatékony optimum kereső algoritmusokban kamatoztatni. Az evolúció során egyre jobban alkalmazkodnak az élőlények a környezetükhöz szelekció és reprodukció által - ez az ötlet vezérli a genetikus algoritmusokat. A kanonikus alakja egy céllal és kényszerek nélküli megoldáshalmazon keres, génjei binárisak.

A különböző megoldások kromoszómákban vannak tárolva, ezeket gének építik fel. A gén az információ legkisebb építőegysége. Minden egyed rendelkezik egy kromoszómával és egy "fittségét" (a megoldás minőségét) jellemző értékkel, ennek segítségével lehet a megoldások között válogatni, egy célfüggvény (fitneszfüggvény) segítségével lehet kiszámítani.

A keresés menete a következő (az 1.1. ábrán követhető): Egy kezdeti, véletlenszerű egyedekből (megoldásokból) álló szülő populáció a kiindulási állapot. Ezután ki kell számítani minden egyednek a fitnesz értékét és szelekcióval biztosítani kell, hogy a jobb szülőknek több utódja lehessen, a rosszabbaknak kevesebb. A legegyszerűbb szelekció a rulett kerékszelekció: Minden egyednek egy cikkely felel meg a rulett keréken, ennek nagysága arányos az adott egyed fitnesz értékével. Minden utód létrehozásánál egy forgatással választunk a szülők közül, mivel a jobb egyedeknek nagyobb a cikkelyük ezért nagyobb valószínűséggel lesz több utóduk.

Az utódok a szülőkből jönnek létre keresztezéssel és mutációval. Keresztezésnél két szülő jó tulajdonságainak egyesítését reméljük. A legegyszerűbb módszer az egy-pontos keresz-



1.1. ábra. *A kanonikus genetikus algoritmus működése*

tezés: mindkét egyedben egy azonos helyen lévő pont (véletlenszerű, hogy hol lesz) utáni rész az utódjában szerepel, a pont előtti rész a másik szülő pont előtti része lesz.

Mutációhoz egy egyed kell csak, egy pici véletlenszerű változtatás történik, legegyszerűbb alakja: Egy fix valószínűsége van annak, hogy mutálódik egy gén (ellentettjére változik az értéke). Ez a valószínűség nagyon picire van állítva, például 0.001-re és így sok egyedenként csak egyetlen gén változik. Lényegében alig változik a populáció a mutáció hatására, mégis szükség van rá, mivel csak így érhetők el azok a megoldások amiknek az építőegységei nem találhatók meg a kezdeti szülőkben. Az utód populáció létrejöttével a generáció végére értünk. A következő generáció szülői a jelenlegi generáció utódai lesznek. - A jó megoldások megtalálásához sok generációra van szükség.

Fontos szempont a keresés folyamán, hogy ne legyen korai konvergencia, vagyis ne forduljon elő, hogy az összes egyed a populációban megegyezik egy erős egyed kezdeti felbukkanása révén. Ennek oka az lehet, hogy a többi egyedhez képest lényegesen jobb ez az erős egyedhez tartozó megoldás, emiatt nagyon nagy lesz a cikkelyének a mérete, és pár generáció után már eltűnnek az egyéb megoldás jelöltek, lokális optimumba torkollhat a keresés.

A lokális optimumba torkollás elkerülésére (és ez nem csak korai konvergencia miatt történhet) fontos a populáció diverzitásának, sokszínűségének megőrzése, vagyis, hogy sok különböző egyed legyen jelen a populációban.

Nagyon sok múlik a paramétereken:

- Populáció nagysága, vagyis, hogy hány egyed legyen egy generációban.
- Generációk száma, vagyis hány generáción át folytatódjon a keresés.

- Keresztezés valószínűsége (mindig van esélye annak, hogy keresztezés nélkül, csupán másolással kerül át a két szülő az utód populációba).
- Mutáció valószínűsége, mennyire gyakran és mennyire változzanak reprodukciónál az egyedek.

Ezenkívül a megfelelő célfüggvény és génkódolás szerepe kiemelkedő. Génkódolásnál (a megoldás génekben való kifejezése) többek között a következő szempontokat érdemes fegyelembe venni:

- Mennyire költséges a dekódolás (a génekből a megoldás előállítása) folyamata.
- A génekben kifejezett távolság mennyire érvényes a megoldások között. Például egyetlen gén megváltoztatása a megoldásban is kicsi változással jár-e.
- A gének által reprezentált megoldások száma: egy megoldáshoz egy génsorozat (kromoszóma) tartozik-e, vagy esetleg több megoldás is tartozhat egy kromoszómához, esetleg egy megoldáshoz több kromoszóma tartozik.

Segíthető a keresés egy olyan kezdeti populációval, ami megfelelő eloszlással lefedi a keresési teret.

2. fejezet

A Megoldás kifejtése

Jelenleg kiindulásként feltételezzük, hogy a hiba helye (esetleg annak megtalálására egy hálózatrész) ismert és elkülönített (kapcsoló állások fixálásával lehet ezt jelezni). Tehát adottak a kiindulási feltételek:

- vezetékek r,x értékei,
- topológia,
- fogyasztások,
- betáplálások,
- kapcsolók helyei,
- a kiindulási és,
- a normál üzemhez tartozó állásuk(a kapcsolók állása rögzíthető mind nyitott mind zárt állapotban).

A feladat a megfelelő kapcsolások sorozatának megtalálása. A probléma két részre van bontva, az elsőben az eljárás megkeresi a kívánatos hálózati topológiát, a második lépésben pedig az ideális kapcsolási sorrendet. Ezzel egyszerűsödött a keresés. A két lépés egyazon algoritmusban egyszerre kezelve túl nagy feladat lenne a mostani módszerekkel. (lassú, nem megfelelő konvergencia, nem mindig ad jó eredményt)

2.1. Első rész: kívánatos topológia meghatározása

A bemenő adatok a kiindulási feltételek (lásd fentebb),a rögzített kapcsolók és állásaik, valamint az algoritmus paraméterei.

A hiba által érintett terület lesz az éppen vizsgált hálózatrész (általában egy leágazás vagy alállomási körzet), mivel a szomszédos részeken valószínűleg nem lesz szükség intézkedésekre (az aktuális hálózatrésszel való összeköttetést biztosító kapcsolókat leszámítva) ezért ott állandónak vesszük az elrendezést.

Az így kiragadott hálózatrészt a szomszédos hálózatrészekkel összekötő ágak kezelése a következő módon történik: A szomszédos zóna helyét egy fiktív betáplálás veszi át,

teljesítményét akkorára állítjuk, amit a szomszéd még tud szolgáltatni az aktuális zóna felé, feltételezve, hogy ott minden állandó marad a helyreállítás alatt. Egy egyszerű iterációval kiszámolható a használható maximális teljesítmény: egy kezdeti lépésnagysággal növeljük a keresztág helyére rakott fiktív betáplálás nagyságát (ez a szomszédos zónában fogyasztásként jelentkezik), minden ciklusban lefuttatva egy load flow-t (a megoldás során alkalmazott módszerről bővebben az F.1 mellékletben és a [7] cikkben lehet olvasni). Ha már nem teljesülnek a szabványok szerinti követelmények, akkor a lépésnagyságot felezve, egyet visszalépünk, majd még egyet felezve ismét növeljük. Ezt elvégezzük külön a hatásos és külön a meddő teljesítményre, minden keresztágra. Ezen kívül figyelembe kell venni, hogy az egyes ágakon a szomszédból szolgáltatható teljesítmények összegének az ottani alállomás(ok) terhelhetőségnek is meg kell felelnie. Az eredmény természetesen közelítő lesz, de ez elég támpontot ad már a szomszédokkal összekötő vonalak terhelhetőségéhez.

Az algoritmus kimenete egy kapcsolási kép, a kiesett fogyasztások és az adott állapotba jutáshoz szükséges átkapcsolások száma. Az utolsó generációban szereplő több lehetséges jelölt közüli kiválasztás történhet kizárólag valamelyik cél preferenciájával, vagy valamilyen súlyozásokat alkalmazva, de lehetőség van a második részből (kapcsolási sorrendek alapján) minden jelöltre másodlagos eredményt is számítani és ez alapján választani a megoldások közül. Utóbbinak előnyére egy példa: Két jelölt maradt, ezek közül kell választani. Az egyik 3 kapcsolásos megoldás ami 300 KW-nak megfelelő fogyasztás szüneteltetését jelenti, a másik 5 kapcsolásos de csak 150 KW-nak megfelelő fogyasztás felfüggesztését eredményezi. Ha minél gyorsabbak akarunk lenni akkor valószínűsíthetően a kevesebb kapcsolás gyorsabb lesz (persze sok múlik a kapcsolók távolságán is). Ám lehetséges, hogy az 5 kapcsolásos megoldás első pár kapcsolása után (ami nagyjából hasonló ideig tart mint a 3 kapcsolásosé) szintén 300 KW körüli lesz a kiesés, de ezután még további javulást érhetünk el.

A probléma megoldásához a *Nondominated Sorting Genetic Algorithm 2* (NSGA-2) [8] módszert választottam alapul, mivel viszonylag egyszerű, alkalmas többcélú, nem folytonos, feltételekkel behatárolt halmazon a keresésre. A hálózat reprezentációja gráfként történik.

Célok:

- minimális kapcsolás,
- minimális kiesés.

Feltételek, korlátok:

- mindig maradjon sugaras a hálózat,
- ne legyen túlterhelve sem a vezeték sem a betáplálás,
- a megengedett feszültséghatárokon belül maradjon minden fogyasztás[1].

[1] középfeszültségen 10 százalék feszültségesés a maximum (elosztói szabályzat 5.sz módosítás (aktuális) 99. oldal 12.3.3.7 pont)
`https://e-iroda.eon-eszakdunantul.com/download.php?url=download/Elo_szab_torzs_5_sz_mod_171_2011.pdf`

A sugaras hálózati topológia (gráfként véve a hálózatot egy fát kapunk) megtartása annyira szigorú, hogy eleve a gének kódolásánál figyelembe vesszük, vagyis minden egyed dekódolása után kapott hálózat meg fog felelni ennek a követelménynek.

2.1.1. Génkódolás

Több reprezentáció is lehetséges:

- Bináris: minden kapcsolónak van egy saját génje, ennek értéke (0 vagy 1) a kapcsoló állapotát mutatja.

  Előnyök: nincs költséges dekódolás, egyszerű mutáció és keresztezés.

  Hátrányok: nem lesz minden reprezentációhoz érvényes hálózat (hurkok lehetnek benne [2], vagy szembe lehet kapcsolva két alállomás[3]) ezért javító algoritmusra van szükség ami költséges lehet és torzítja az egyenletes eloszlást.

- Fa alapú: a génekben a fákat tároljuk, például szélességi bejárás során kapott sorrendben a pontokat.

  Előnye: a mutáció és keresztezés jobb eredményt ad.

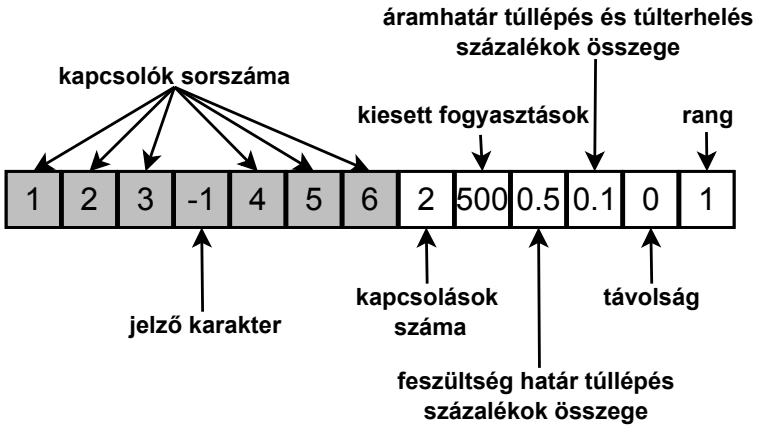
  Hátránya: bonyolult mutáció és keresztezés, változó génhossz.

- Permutáción alapuló: minden kapcsolóhoz rendelünk egy sorszámot, ezek egy permutációja van tárolva és a dekódolás során alakul ki az aktuális hálózat.

  Előnye: minden lehetséges permutáció érvényes hálózathoz vezet, könnyebb kezelni a sugaras követelményt.

  Hátránya: költséges dekódolás.

A választott génreprezentáció nem a szokásos bináris, hanem permutáción alapuló: A génekben a kapcsolók sorszáma szerepel, valamint egy jelző -1 érték (ezzel van lehetőség bizonyos hálózatrészek ellátásának mellőzésére).



2.1. ábra. *Az egyedek tárolásának módja a választott reprezentációval, a szürke hátterű rész a kromoszóma, a többi a társított értékek.*

[2]kör a gráfban

[3]Kerülendő, hogy hosszabb ideig párhuzamossá váljon a középfeszültségű hálózatrész a nagyfeszültségűvel.

A Dekódolás menete a következő:

1. Minden kapcsoló ki van kapcsolva.

2. Sorban veszi a géneket az algoritmus az első géntől kezdve minden génre megvizsgálja:

   Ha az adott kapcsoló bekapcsolása nem vezet hurokhoz vagy két betáplálás szembekapcsolásához akkor bekapcsolja az adott sorszámú kapcsolót (ezek ellenőrzéséhez szélességi fabejárás nyújt segítséget).

   Egyébként kikapcsolva marad.

3. Ha -1 van a génben akkor az utána következő kapcsolók már nem lesznek bekapcsolva.

A keresési tér kibővül az egyszerű bináris kódoláshoz képest (2^n helyett $n!$, n a kapcsolók száma), viszont sokkal könnyebben kezelhető a sugaras követelmény, összefüggőbb lesz a keresési tér, folyamatosabban elérhetők a különböző megoldás jelöltek egymásból (binárisban például ahhoz hogy egy hurkot máshol nyissunk, egyszerre kell megváltozni a két kapcsolónak az állását, ez itt nem jelentkezik), valamint egy-egy mutáció nem vezet feltétlen a kapcsolási kép megváltozásához (hiszen sok génsorhoz tartozhat ugyanaz a kapcsolási kép). Ráadásul minden mutáció és keresztezés eredménye sugaras lesz, így nincs szükség javító algoritmusra.

Ehhez a reprezentációhoz nem használhatók a szokásos mutációk és keresztezések.

2.1.2. Műveletek a géneken

A következő mutációk vannak implementálva jelenleg (zárójelben a paraméter a szabályozáshoz):

- Két tetszőleges helyen lévő gén felcserélése (valószínűség, hogy kiválasztásra kerül a vizsgált gén),
- két egymás melletti gén felcserélése (valószínűség, hogy kiválasztásra kerül a vizsgált gén),
- egy véletlenszerű kapcsoló beszúrása egy véletlenszerű helyre (hányszor történjen meg),
- egy véletlen hosszú génsor kivétele és beszúrása máshová szintén véletlenszerűek a helyek (a génsor hosszának határai),
- egy véletlen hosszú génsor kivétele és fordított sorrendben visszaírása (a génsor hosszának határai).

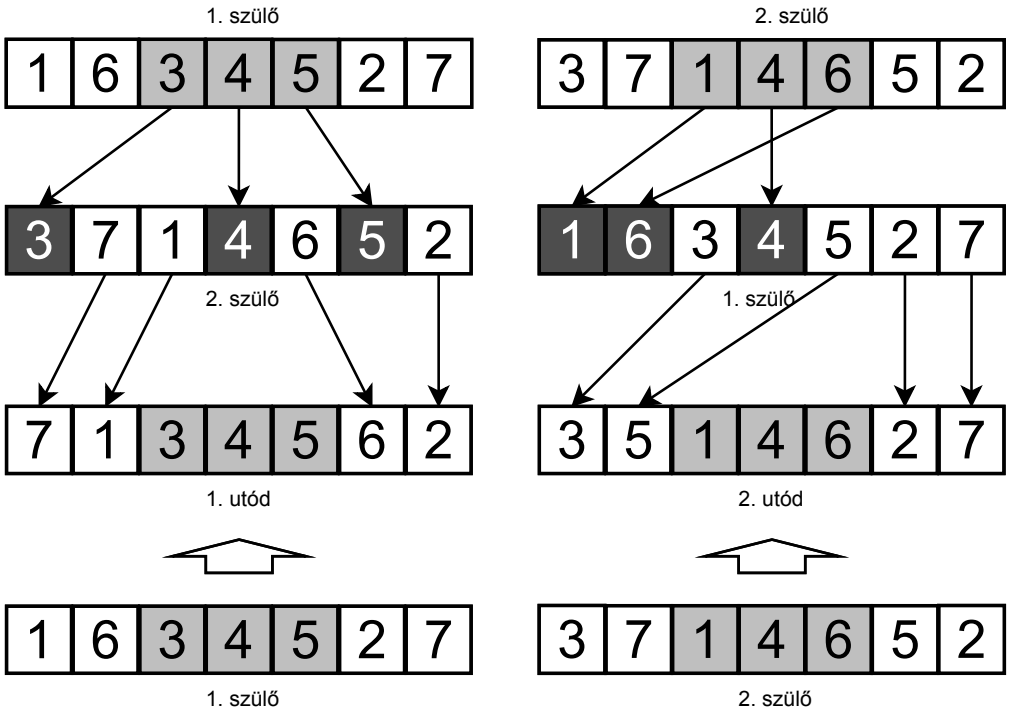
A mutációk erősségét és gyakoriságát valamilyen módon szabályozni kell. Az elitizmus révén nem fontos, hogy legyen esélye egy-egy egyednek változatlan formában megjelennie az utódok között, ezért mutációt a keresztezés után minden egyedre alkalmazunk. Mivel a kódolás robusztus még így is lesznek olyanok, amiknek a keresztezés utáni kapcsolási képük változatlan marad.

A leghatékonyabbnak(gyors konvergencia, megfelelő minőségű megoldások) az első módszer bizonyult, így ez került alkalmazásra az algoritmusban: Minden génen egyenként sorba megy az algoritmus, egy bizonyos valószínűsége van annak, hogy az adott génen mutációt végez. A valószínűséggel változtatható az erősség. Ha 1/génszám a valószínűség akkor minden egyeden átlagban egy pici változás lesz, ha több akkor nagyobb változás lehet (érdemes azért 0.5-nél kisebbre választani, hiszen e felett már véletlenszerű keresés folyik), ha kevesebb, akkor több egyedenként lesz csak egy változás.

Nagyobb változásokkal gyorsabban fedezzük fel a lehetséges megoldásokat, de azok finomhangolását megnehezíti. Ezzel szemben a kicsi változások segítik a finomhangolást, de lassúak a felfedezésben. Fixen tartva az erősséget valamilyen kompromisszumot kell vállalni a kettő között. Javítható a hatékonyság a következő módszerrel: az algoritmus előrehaladtával egyre csökkentjük a mutáció erősségét egy kiindulási erősebb értékről (0.5-ről ha az első pontban vázolt módszert alkalmazzuk): $mutp = 0.5 * (1 - \frac{gen}{generacio})$ ahol mutp az aktuális mutációs valószínűség lesz, gen az eddigi generációk száma, generacio pedig az összes megengedett generációk száma. Ezzel kezdetben gyorsabban halad a felfedezés, de később eltolódik a finomhangolás irányába és segíti a legjobb megoldások megtalálását.

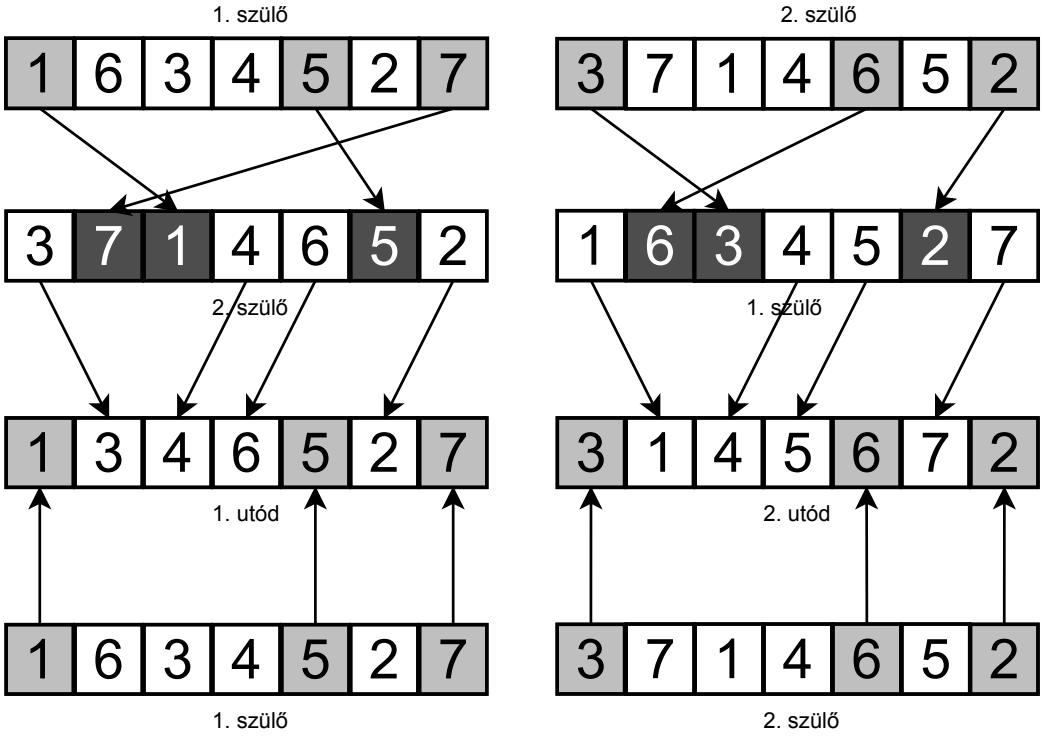
A következő keresztezések vannak jelenleg implementálva (ezek közül az első módszer bizonyult hatékonynak, ezt alkalmaztam az algoritmusban):

- Mindkét egyedben ugyanazon a helyen egy szakasz kiválasztása, ezek lesznek az utódok alapjai. Az egyik szülő szakaszában lévő elemek kivétele a másik szülő génjeiből ezután sorban a szabad helyek feltöltése ezekből az utódban (Lásd 2.2. ábrát).



2.2. ábra. *Keresztezés szemléltetése*

- Véletlenszerűen kiválasztásra kerülnek bizonyos gének, ezek a helyükön maradnak az utódban is, az üres helyeket a másik szülő génjeivel töltődnek fel, ismétlések ez esetben sem lesznek (Lásd 2.3. ábrát).



2.3. ábra. *Keresztezés szemléltetése*

2.1.3. Célok és feltételek számítása

Az ellátott fogyasztók és a kapcsolások száma könnyen meghatározható. A fabejárások során (géndekódoláskor) megjelölésre kerülnek a feszültség alatt lévő fogyasztók, egyszerűen csak összeszámoljuk a nem ellátott fogyasztásokat. Azért nincs figyelembe véve a feszültség az ellátott fogyasztóknál, mert eleve olyan jelölteket keresünk, amik a szabvány értékeket betartják, így ahol nem lenne elfogadható a feszültség érték, ott a feltétel sértés miatt hátrányban szenved eleve a jelölt.

Kapcsolások száma a kiindulási állapothoz képesti eltérések száma (fixált kapcsolók nem számítanak bele, ezeket így is-úgy is használni szeretnénk).

A korlátok megsértése nem számolható ilyen egyszerűen, load flow segítségével kerülnek meghatározásra. Az alkalmazott load flow (az F.1 mellékletben találhatók a részletek, valamint a [7] cikkben) alkalmas sugaras és gyengén hurkolt hálózatok számolására, csak egyetlen szabályozott feszültségű sín lehet benne (más lehetőség a sugaras struktúra és a hálózatüzemeltetési gyakorlat folytán nem adódik), viszont gyors, egyszerű. Ez sokat számít mivel nagyon sokszor kell lefuttatni: minden egyedre minden generációban.

Az egyedhez a követelmény sértések társítása: Mindenhol ahol nem felel meg a hátértékeknek az adott érték (áram vagy feszültség), ott kiszámításra kerül a túllépés százalékos

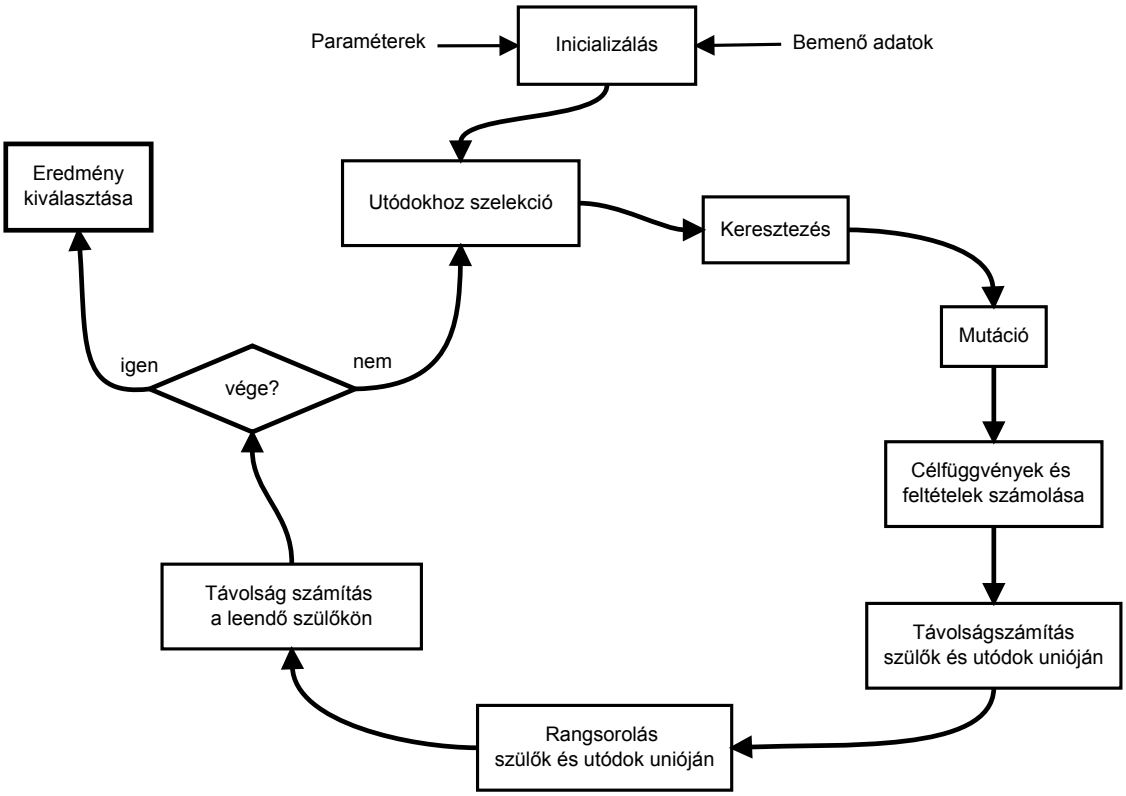
értéke, majd ezek összege kerül az egyedbe. Külön van kezelve a feszültség és az áram határ, utóbbiba számít az is, ha egy betáplálás túlterhelődik. Ha véletlenül nagyon nagy terhelés kerülne egy kis betáplálásra, és ennek hatására a feszültségesés nagyobb lenne mint a betáplálás feszültsége, akkor végtelen értéket kapnak a kihágások.

2.1.4. Az algoritmus

Egy áttekintő képet láthatunk lentebb (2.4. ábra) az algoritmus működésének alapjairól. A mutációval, keresztezéssel, célfüggvények és feltételek számításával foglalkoztunk már.

A szelekcióhoz szükség van rangsorolásra és távolság számításra is. A rangsorolás dominancián és távolságon alapul. A dominancia az egyedek jóságán (célfüggvény értékein) és használhatóságán (feltételeknek való megfelelés) alapul. A következőkben sorra vesszük a még nem tárgyalt részeket.

A távolság taglalásával érdemes kezdeni.



2.4. ábra. *Az algoritmus blokkvázlata*

Távolság

Távolság alatt a következőre kell gondolni:

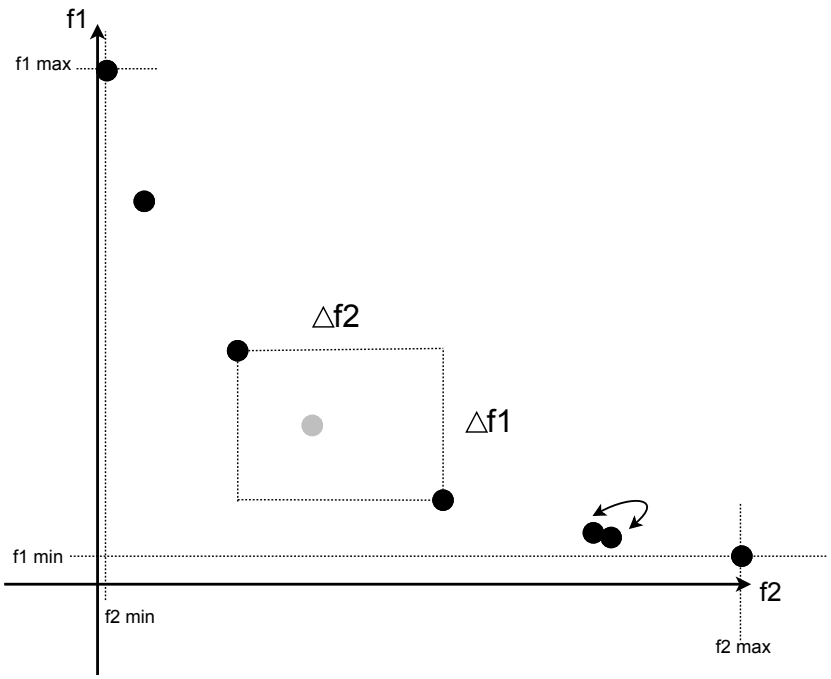
1. sorba rendezve az egyedeket az egyik célfüggvény érték szerint
2. az első és utolsó elem a sorban végtelen távolság értéket kap

3. a többiekre: az előtte és utána álló célfüggvény érték különbségét ($\triangle f_i$) normálva (a legnagyobb és legkisebb célfüggvény érték (az adott generációban) különbségével ($f_{max,i} - f_{min,i}$)),

4. 1-3-ig pontokat megismételni minden egyes célfüggvényre,

5. majd rendre összeadva a különböző célfüggvényekhez tartozó távolságokat, így kapható meg az egyetlen mérőszám.

Szemléltetésül a 2.5. ábrán követhető, hogy mit jelent ez egy két célfüggvényes esetben.



2.5. ábra. *A távolság elvének szemléltetése*

Tulajdonképpen az egyes egyedek különbözőségét tudjuk mérni a távolsággal (minden egyednek saját távolság értéke van). Ha kicsi egy egyed távolsága akkor valamely másik vagy több másik egyedhez nagyon hasonló megoldást hordoz. A szélsőértékeket mindig szeretnénk megtartani, ezzel segítjük az egyes célfüggvények legjobb értékeinek megtalálását. A legjobbat magától értetődő, hogy miért tartjuk meg, a legrosszabbat azért, mert valószínűleg az az egyed, amelyik az egyik dologban a legjobb, a másikban a legrosszabbat nyújthatja. Ezenkívül a számolás módszere miatt is szükséges (a szélsőértékeknek csak egy szomszédja van).

A nyíllal jelölt két pont nagyon hasonló egymáshoz a többi megoldás elhelyezkedéséhez képest. Az ilyen helyzeteket próbáljuk meg elkerülni a távolság használatával, vagyis megpróbáljuk egyenletesen eloszlatva megtalálni a megoldásokat a Pareto-front mentén.

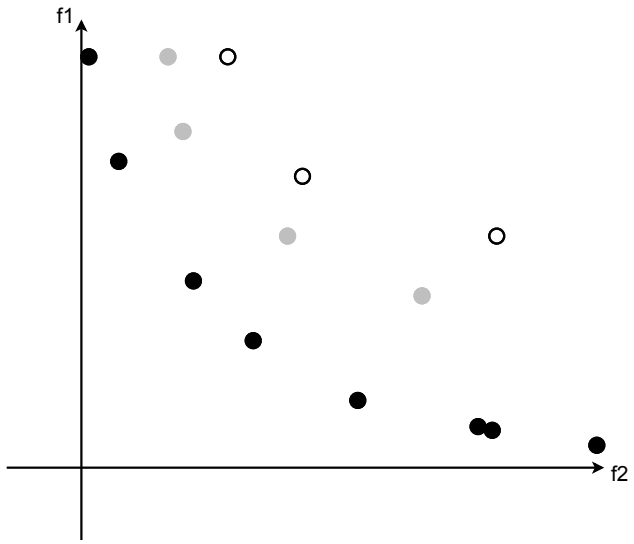
Pareto-front alatt azon megoldások halmazát értjük, ahol nem lehetséges az egyik szempontból javítani a megoldás jelöltet egy másik szempont rontása nélkül.

Dominancia, rangsorolás, leendő szülők

A Dominancia fogalma a Pareto-frontok kialakításánál játszik szerepet. Ennek segítségével érhető el, hogy a fejlődés egyre jobb és jobb egyedeket hozzon.

Egy egyed akkor domináns a másikkal szemben, ha minden célfüggvény érték tekintetében jobb nála, amennyiben egyik sem sért feltételt. Ha az egyikük feltételt sért akkor az a domináns amelyik nem sért. Ha mindkettő feltételt sért akkor ismét a célfüggvények a döntőek. A feltételek kezelésére léteznek más módszerek is ([8] cikkben is szerepel egy alternatív megoldás), de ez is megfelelő eredményt ad és mivel rengeteg összehasonlításra van szükség ezért a minél gyorsabb megoldást érdemes előnyben részesíteni.

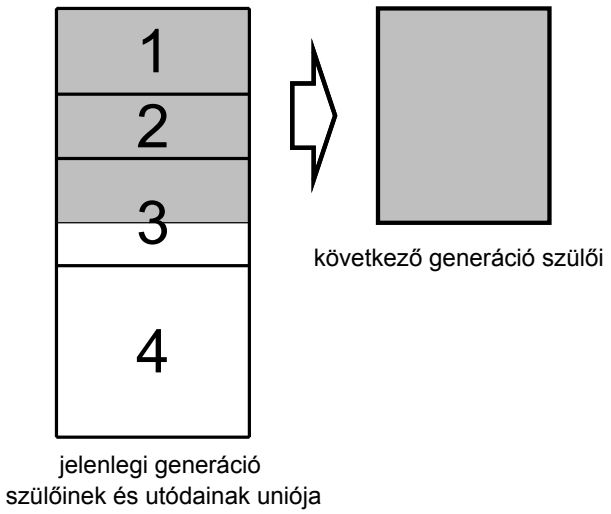
Rangsorolásnál az egyes frontok külön rangot adnak. A szülő és az utód egyedek unióján rangsorolunk, ennek köszönhető az elitizmus.



2.6. ábra. *A Pareto front és a rangsorolás (fekete pontok az 1. rangúak, szürkék a 2., üres karikák a 3.)*

Minden lehetséges párosításban az egyedek összehasonlításra kerülnek egymással a rangsorolásnál. Ennek során különböző rangokat kap az egyedek egy-egy csoportja. Egy ranghoz azon egyedek tartozhatnak akik nem dominánsak egymással szemben. Az első rangot azok kapják akik egymáson kívül a teljes populációval szemben dominánsak, a második rangot azok kapják akik egymáson és a jobb rangúakon kívül a teljes maradék populációt dominálják és így tovább amíg mindenki rangot nem kap. Ha kész van a rangok kiosztása akkor a következő generáció szülői a legjobb rangúak lesznek. Ha az egy rangba tartozók száma kisebb, mint amennyi még elfér a következő generáció szülőiben akkor a rangból mindenki

átkerül, ha már nem férne be mindenki akkor a legnagyobb távolságúak jutnak be, ezzel előnyben részesülnek azok a megoldások, amik különböznek a többségtől (lásd 2.7. ábra). Mivel a különböző célok szerint az esetek többségében több különböző egyed van jelen a populációban, amiket az elitizmus meg is tart, ezért sokkal kisebb az esélye a korai konvergenciának és nagyobb a diverzitás (sokszínűség). Ezáltal a lokális optimumba ragadásnak is csökken a valószínűsége.



2.7. ábra. *A következő generáció szülőinek kialakítása*

Szelekció és reprodukció

Véletlenszerű két egyed *mérkőzik meg* egymással a szülő populációból, az nyer akinek jobb a rangja vagy rangegyenlőség esetén a nagyobb távolságú. A nyertes az egyik utód alapja lesz. Ezután újabb két véletlenszerű egyed következik (egy szülő jelölt többször is használható, emiatt lesz több utóda a jobb egyedeknek). Addig folytatódik ez amíg a megfelelő számú utód-alap meglesz. Így a jobb egyedek valószínűleg több utóddal rendelkeznek majd, valamint itt is szempont marad az egyenletes lefedése a megoldáshalmaznak (a nagyobb távolság preferálása biztosítja). Az elitizmus miatt elég egy ilyen viszonylag egyszerűbb, de rugalmasabb módszer, amivel jobban kezelhető a több cél és feltétel, nincs szükség globális mutatókra.

A leendő utódok egyelőre mindössze még csak a szülők másolatai, a következő két lépésben nyerik el végső formájukat. Először véletlenszerű két egyed (az utód-alapokból) kereszteződik egymással egy bizonyos valószínűséggel, majd a következő véletlenszerű két egyed jön és így tovább. Itt már nem kerülhet sorra többször ugyanaz az utód-alap (tartalmilag persze egyezhetnek különböző utód-alapok, hiszen egy szülőnek több is lehet). Majd ha megvolt az összes keresztezés akkor minden egyedre lefuttatunk egy mutációt, ebben a

változó értéke azt határozza meg, hogy mennyire változtassa meg az egyedeket a mutáció.

Inicializálás

Itt történik a bemenő adatok beolvasása, átalakítása és a belőlük származtatott paraméterek kiszámítása. A szülők kezdeti populációjának kialakítása és belőlük történik az első utódok kialakítása már rangsorolással, szelekcióval, keresztezéssel, mutációval. Szülőknél két lehetőségünk van a kezdeti populáció kialakítására: Teljesen véletlenszerű minden egyed, vagy egy esetleg néhány egyed célzottan kerül a kezdők közé. Például az egyik egyed a normál üzemi állapotot hordozza magában. Ez ugyan némileg korlátozza a globális keresést, de ha a hiba kiterjedése kicsi, várhatóan úgyis néhány kapcsolásnyi távolságra keressük a megoldásokat a normál állapottól. Később a teszthálózaton végzett próbák során kiderült, hogy nagyobb hátránya van a normál üzemállapot megjelentetésének a kezdeti populációban súlyosabb hibák (nagyobb érintett hálózatrész) esetén: a lokális optimumban ragadás esélyének megnövekedése, emellett alig gyorsult a konvergencia.

Valamint itt történik még az első egyedek kiértékelése is és a feltétel beállítása a fő ciklus leállásához. Az esetek nagyrészében nem fog bekövetkezni, hogy a teljes populáció ugyanazokból a megoldásokból áll (mivel több cél szerinti legjobb egyedek kerülnek megtartásra) így a teljes konvergenciára nem támaszkodhatunk. Ezért az egyetlen feltétel a maximális generációszám elérése.

2.1.5. Skálázhatósági kérdések, robusztusság

Részenként haladva vizsgáljuk meg a teljesítményigényt egy generációra:

Jelölések: m: konstans, f: célfüggvények száma, n: egyedek száma, csp: csomópontok száma, kp: kapcsolók száma

- inicializálás: itt történik az adatok beolvasása, az első szülő és utód egyedek kialakítása és az első kiértékelése az egyedeknek. Ennek ideje elhanyagolható a többi részhez képest.
- távolságszámítás: minden célfüggvény szerint sorba kell rendezni és kiszámolni a távolságokat, $m * f * 2 * n * log(2 * n) + f * n$
- rangsorolás: [8] alapján $m * (2n)^2 + n * log(n)$ (a szelekció egy része)
- szelekció: $m * n$
- egyértelműen az $m*(2n)^2$ dominál a fentiekben, csökkenthető az igény, ha csak addig rangsorolunk, amíg megtaláljuk a megfelelő számú egyedet
- keresztezés: $m * n * (kp + 1)$
- mutáció: $m * n * (kp + 1)$
- kiértékelés:

  dekódolás:

- legrosszabb: $m * n * ((kp + 1) * csp + csp + loadflow)$
- átlagosan: $m * n * ((kp + 1) * csp/2 + csp + loadflow)$

loadflow:

- legrosszabb: $m * 10 * (3 * elszam)$
- átlagos: $m * 3 * (3 * elszam)$ (általában maximum három iteráció után konvergál)

Tapasztalataim szerint az egész futás idő nagy részét a dekódolás teszi ki (több nagyságrend a különbség, 8-12 egyeddel a kiértékelés és az algoritmus többi része között). A load flow a következő legnagyobb időt igénylő rész, futása egyébként értelemszerűen attól függ, mekkora aktív hálózatrészt kell kiszámolnia. Viszont még a nagyobb (148 csomópontos) hálózaton, 250 generációval és 10 egyeddel is mindössze 3 másodpercet tett ki a teljes időből (a dekódolás 35 másodpercet). - Egy gyors előkészítő és fabejáró algoritmus sokat gyorsíthat a futáson.

A generációk számától lineárisan függ a futásidő.

A megvalósítás matlabbal történt, nyilván sok helyen lehetne még optimalizálni, de elsőre egy prototípus megvalósítása volt a cél.

Tájékoztató jelleggel a gyorsaságról: A következő pontban tárgyalt mintahálózaton (36 csomópont, 28 kapcsoló, 38 él) 8.5s futásidő volt mérhető a matlab tic-toc utasítás párjával, egy másik teszthálózaton (148 csomópont, 103 kapcsoló,150 él) 38s volt mérhető. Az utóbbi a hazai gyakorlatban nagy leágazásnak számít. Tehát jól skálázódik, ráadásul nagyon jól párhuzamosítható így nagyobb hálózatok is kezelhetők, ezért több futásra is marad idő a végleges megoldás javaslatáig. Nem is beszélve egy alacsonyabb szintű nyelven (például C-ben) megírt megvalósításról, amivel akár nagyságrendben mérhető gyorsulás érhető el.

A többszöri futtatásnak az elv nem determinisztikus jellegéből adódóan van értelme. Tegyük fel, hogy az esetek nagy részében megtalálja a legjobb, vagy ehhez közel eső megoldást az algoritmus, akkor annak az esélye, hogy nem megfelelő minőségű megoldásokra jutunk többszöri ismétlés után már elenyésző lesz.

A robusztusságról:

A topológia értelemszerűen fontos, hogy pontos legyen (nem földrajzi értelemben, hanem összeköttetések szintjén), viszont elég csak egyszer kialakítani, a legtöbb helyen a SCADA rendszerekben amúgy is rendelkezésre áll, onnan könnyen kinyerhető. A hálózati paraméterek pontossága nem olyan kritikus, könnyen pótolhatók a hiányzó adatok.

A program megpróbálja a megoldásokat a megszabott feltételek betartásával megoldani. Így érzékeny lehet ezek kezelésénél. Például ha egy adott fogyasztási körzetet nem bír el egy betáplálás, akkor megpróbálja máshova átcsoportosítani egy részét, holott lehet, hogy rövidtávon túlterhelhető lenne. Ez orvosolható a határok bizonyos mértékű kitolásával a bemenő paraméterekben, attól függően, hogy rövid, vagy hosszú távú kiesésre számít a diszpécser. Esetleg egy későbbi fejlesztés során a feltételkezelés módszerének átalakítása is megvalósítható. Az első részen belül nem kezelhető könnyen az időbeliség, mivel az idő alapvetően csak a megoldás második részében szerepel.

Ezenkívül nagy bizonytalanság van a fogyasztások becslésében, ezen csak az okos mé-

rők terjedése és a fogyasztói magatartások jobb megismerése segíthet, alapvetően nem a program fejlesztésével oldható meg.

A fenti nehézségeket leszámítva megbízhatóan jó eredményeket szolgáltat az eljárás, kevés valósidejű adat is megfelelő (kapcsolók állása, a leágazás feszültsége és árama az alállomáson), ezek a mai irányítástechnikában amúgy is rendelkezésre állnak.

2.1.6. Tesztelés

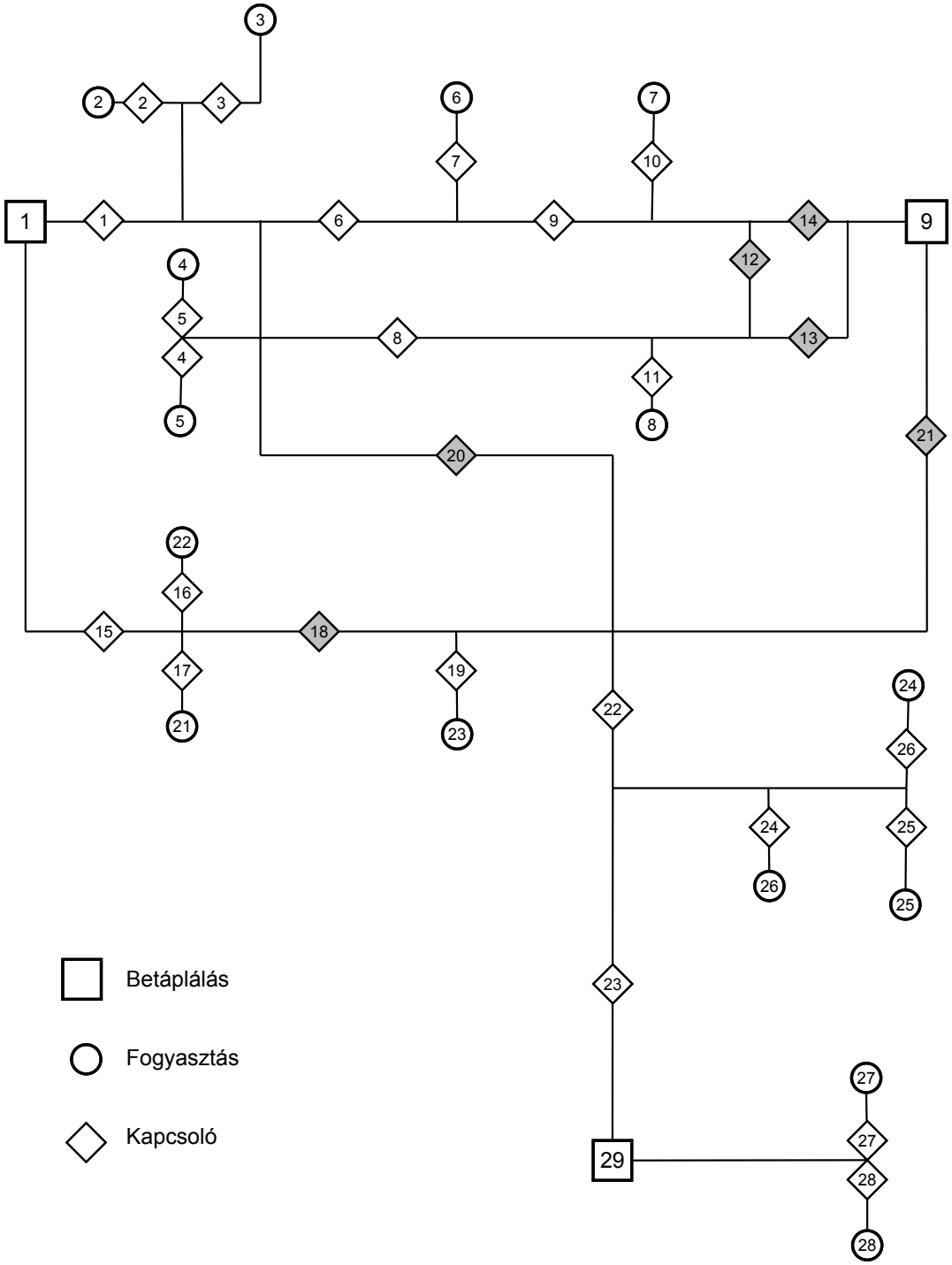
A 2.8. ábrán látható a fejlesztés közben alkalmazott teszthálózat. Ezzel jól vizsgálható a hurkok, több betáplálás, kapcsoló nélküli vezetékszakasz kezelése.

Egy jó kezdeti próba, hogy teljesen véletlenszerű kezdő szülő populációval indulva egy hiba nélküli hálózaton megtalálja-e az algoritmus a normál üzemi állapotot (0 kapcsolás és 0 kiesés). A fogyasztások és betáplálások teljesítményeinek változtatásával (vagy vezeték paraméterek változtatásával) elérhető, hogy fogyasztás átcsoportosítás legyen szükséges a hálózaton, ezt véghez viszi-e az algoritmus. Néhány példa:

- a 7-es számú fogyasztást egy nagyobb értékre állítva átkerült a 9-es betápláláshoz,
- az 1-es betáplálás kapacitásának csökkentésére a fogyasztások egy része átkerült a 9-es és 29-es betápláláshoz.
- az 1-es kapcsoló vezetékszakaszának ellenállását megnövelve a feszültségesés miatt a 7-es és 8-as fogyasztás átkerült a 9-es betáplálásra

Ha a normál hálózati körülmények között minden rendben ment, akkor következtek a zárlatok hatásának vizsgálata. A felkínált megoldások között szerepel-e túlterhelést vagy fogyasztás szüneteltetését okozó - ezek fontos indikátorai az egyes részek működésének. Hiszen néha a túlterhelések elkerülésére szükséges lehet fogyasztás csökkentés. Néhány példa hely: 29-es vagy 1-es betáplálás közvetlen közelében fellépő, vagy egy fogyasztó előtt bekövetkezett zárlat stb... Figyelhető, hogy a normál üzemi állapothoz tartozó egyed szerepeltetése a kezdeti populációban miként befolyásolja a konvergenciát, megoldások minőségét.

Ezenkívül ki lett próbálva egy nagyobb valóságos leágazáson alapuló hálózatra is (150 csomópont, 103 kapcsoló,148 él).



2.8. ábra. *Az alkalmazott teszthálózat, a szürkített kapcsolók alaphelyzetben nyitottak, a többi zárt*

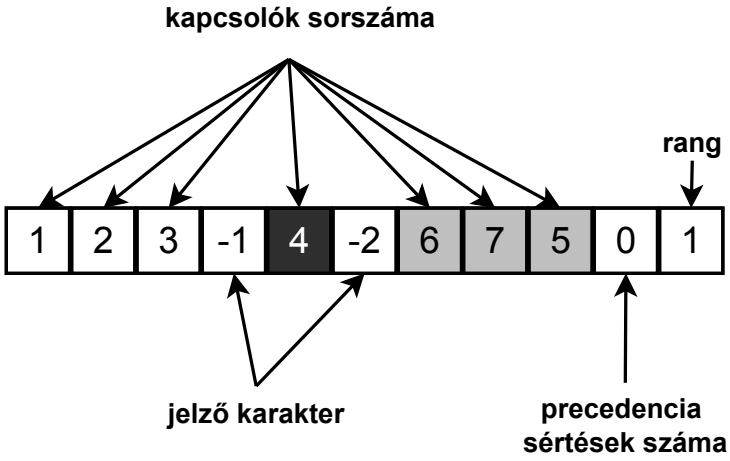
2.2. Második rész: kapcsolási sorrend

A feladat második része a kapott kapcsolások optimális sorrendben történő elvégzése. Ez egyszerűbb feladat már, mivel sokkal kisebb a keresési tér, tipikusan maximum 10 kapcsolásról van szó, viszont figyelembe kell venni a rendelkezésre álló csapatok számát is (hány különböző helyszínen lehet párhuzamosan dolgozni). Lényegében legrövidebb utat kell keresni, több csapattal párhuzamosan lehet haladni, valamint nem számít a visszatérés ideje sem. Az előző rész keresztezései és mutációi itt is alkalmazhatók, hiszen hasonló, permutáción alapuló génkódolás van itt is (lásd 2.9. ábra).

Bemenetként megkapja az algoritmus a kapcsolandó kapcsolók sorszámát, egy mátrixot amiben benne van, hogy mennyi idő alatt érhetők el egymástól és a telephelytől a kapcsolók, valamint, hogy hány csapattal dolgozunk.

Az egyetlen célfüggvény a kapcsolási idő és ezt szeretnénk minimalizálni. Feltétel itt is van: kapcsolási precedencia. Ha nem adunk meg mást (erre van lehetőség egy bemeneti paraméter fájlban) akkor azzal a közelítéssel működik az algoritmus, hogy minden kapcsolót először ki kell kapcsolni majd csak ezután következhetnek a bekapcsolások. Ezenkívül figyelembe veszi még, hogy a csapatok képesek várakozni is, ezáltal lehet, hogy összességében rövidebb idő alatt végeznek.

A kódolás tehát permutáción alapul (lásd 2.9. ábra), a génekben a kapcsolók sorszáma szerepel, valamint a csapatok számánál egyel kevesebb negatív szám (ezzel kijelöljük, hogy egy-egy csapathoz mely kapcsolások tartoznak). A fitnesz érték a kapcsolási sorozat teljes ideje. A feltétel sértés külön van tárolva és a kapcsolási precedencia be nem tartásainak számát tartalmazza.



2.9. ábra. *Az egyedek tárolása a választott reprezentációval,a különböző színű részek egy-egy csapat kapcsolásai*

A szelekcióhoz más módszerre van szükség, mint az első résznél. Itt is rangsoroláson alapul, de másképpen. Az alapul vett módszer: dynamic stochastic ranking [9]. A szülők és az utódok együttesén rangsorol az algoritmus, így elitizmus itt is van:

Buborék rendezéssel történik az egyedek rangsorolása, a fitnesz, a feltétel sértés, valamint

egy külön változó érték segítségével. Alapvetően a fitnesz alapján történik a rangsorolás, de ha feltétel sértés van, akkor nagy valószínűséggel rosszabb rangot kap a kihágó, viszont van egy bizonyos valószínűsége (ezt határozza meg a változó) annak, hogy ekkor is a fitnesz alapján történik a helycsere. Ez a valószínűség lineárisan csökken a generációk előrehaladtával, így biztosított, hogy mire vége az evolúciós folyamatnak, ne lehessen a legjobb rangú egyed használhatatlan megoldás. Viszont kezdetben a keresés segítségére van a precedencia megsértésével is jobb rang szerzésének lehetősége. Addig megy végig az egyedeken újra és újra a rendezés, amíg van helycsere, ha nem volt az utolsó körben akkor befejezettnek tekintett a rangsorolás, a legjobb rangúak lesznek a szülők.

Miután kiosztottuk a rangokat, egyszerű véletlenen alapuló párokban *mérkőznek meg* a szülők az utódhagyásért. Hasonló az eljárás mint az első résznél, csak itt nincs távolság és minden egyednek egyedi rangja van, tehát elég csak a rangok összehasonlításával dönteni. Szintén az erősebb, jobb szülőnek lehetősége van több utódot hagyni, míg a rosszabb rangúaknak erre kisebb esélyük van.

Ezután következik az utódok kiértékelése: Sorrendben minden génen végigmenve a génben található kapcsoló megkapja a működtetésének idejét. Precedencia sértésre úgy derül fény, hogy az egyik kapcsoló korábbi időpontot kap, mint az amelyiknek a működtetése után következhet. Ilyenkor várakozással próbálja feloldani a helyzetet az algoritmus. Lehetőség van többszöri várakozásra is. Ha ekkor sem kerülnek rendbe a sorrendek akkor feltételsértésként beírásra kerül az egyedbe a precedencia sértések száma. A kapcsolás teljes ideje a legnagyobb idővel rendelkező kapcsoló ideje lesz.

Tesztelés és sebesség

Ennek a résznek a tesztelése egyszerűbb az előzőénél. Elég különböző csapatszámokra néhány megoldásnak a kézi követése. A teszteléshez egy nyolc kapcsolóból álló példa szolgált, kettő kikapcsolás, a többi bekapcsolás volt.

Sebesség tekintetében rendkívül gyors, 1 másodperc körüli futásidőkkel 8 kapcsolóval és 1-7 csapattal próbálva. Ennél lényegesen több kapcsolás nem valószínű hogy előfordul hiba esetén (A teszthálózaton 2 külön helyen lévő hibával is csak maximum 9 kapcsolásos megoldás fordult elő). De van még idő sokkal több kapcsolás kezelésére is.

3. fejezet

Az eredmények értékelése, kitekintés, fejlesztések

3.1. Eredmények értékelése

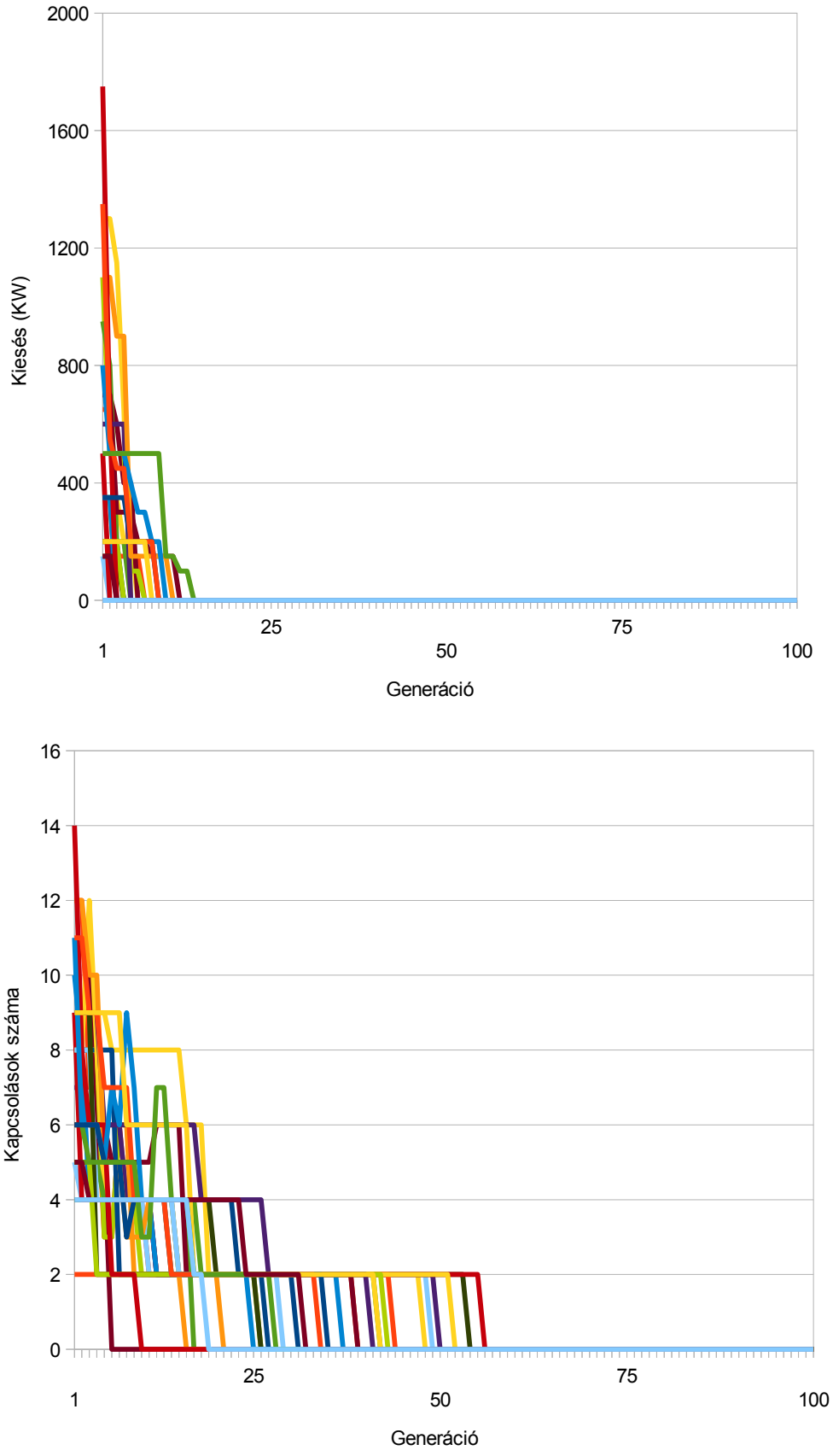
Az első rész során (kívánatos topológia kialakítása) a tesztek 250 generációval és 10 egyedes populációval futottak, a keresztezés valószínűsége 0.8 volt a mutáció pedig a bemutatott módon volt szabályozva. Több lehetőséget is kipróbálva ez ideális választásnak látszott. Minden eseteben a legkisebb kieséshez tartozó megoldásokat vizsgáltam. Mivel nem determinisztikus a módszer, így minden esetre 30 futtatás eredménye látható. Egy futás 8 másodperc körül volt a 2.8. ábrán látható hálózatra.

A normál üzemállapot megtalálása véletlenszerű kezdéssel a 3.1. ábrán látható. Gyors konvergencia látszik, valamint, hogy minden próbálkozás sikerrel zárult. A kiesett fogyasztások minimalizálása általában gyorsabban folyik mint a kapcsolások számáé, ez abból adódott, hogy több módon is lehetséges a fogyasztók ellátása.

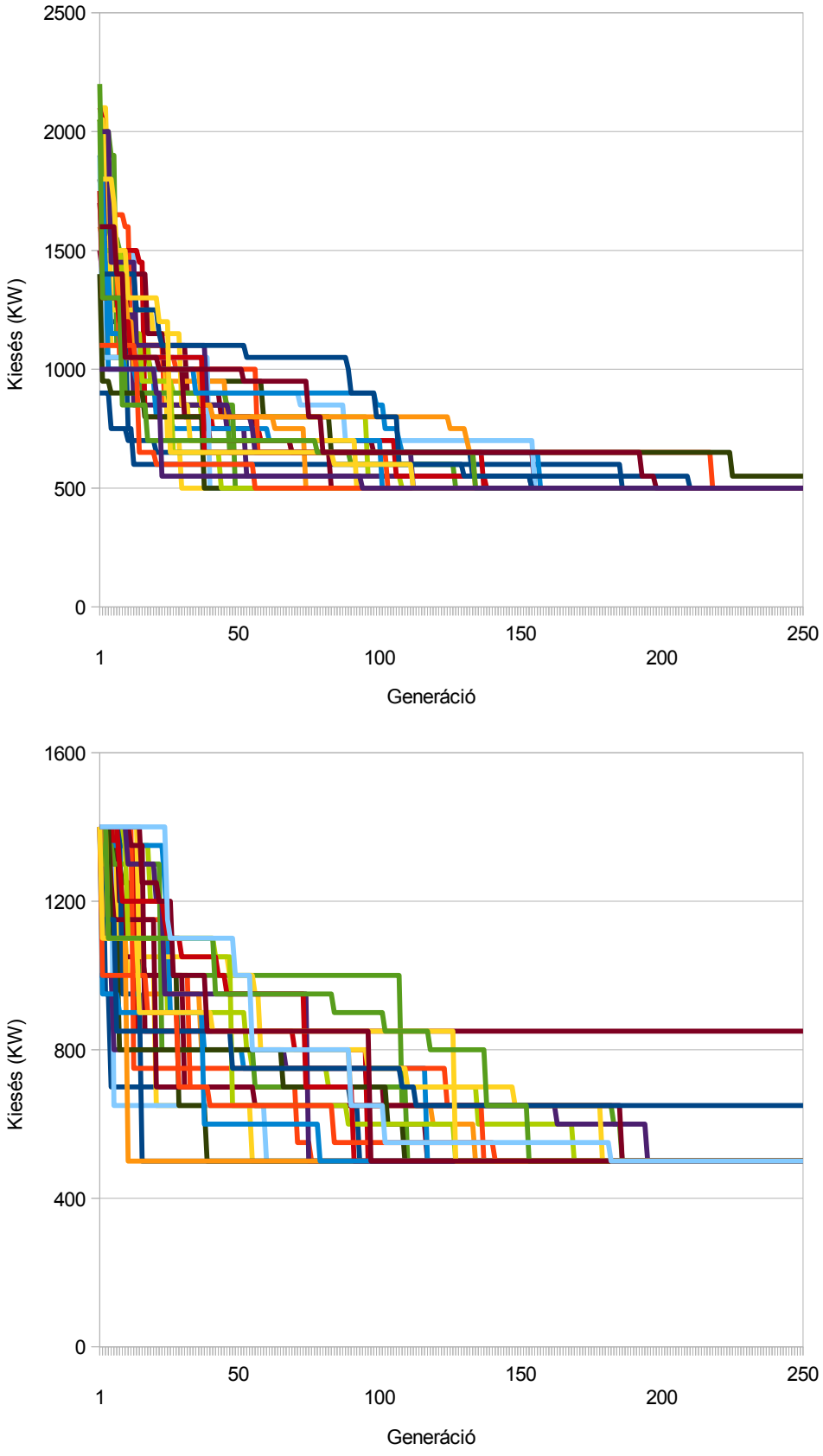
A 3.2. és 3.3. ábrán egyszerre látható egy súlyos hiba kezelése (1-es kapcsoló nyitása a 2.8. ábrán szereplő teszthálózaton), valamint annak a hatása ha a normál üzemállapot jelen van a kezdő populációban. Sajnos semmilyen előnnyel, sőt hátránnyal rendelkezik a második módszer: Lokális optimumba ragadás esélye megnövekszik. A legjobb megoldás 500 KW kiesés volt 7 kapcsolással. 28-szor jutott erre a megoldásra a véletlenszerű kiindulással az algoritmus. Ezenkívül előnyös, hogy a maradék két megoldás is nagyon közel van az optimálishoz: 500 KW kiesés 8 kapcsolással és 550 KW kiesés 8 kapcsolással.

Kevésbé volt jó a normál üzemállapot szerepeltetése kezdésben: 24-szer jutott az optimális megoldásra, 4-szer 500 KW kiesés 8 kapcsolással, 1-szer 4 kapcsolással 850 KW kiesés és 1-szer 650 KW kiesés 9 kapcsolással. Itt tehát az optimális megoldáshoz képest lényegesen rosszabb is előfordult.

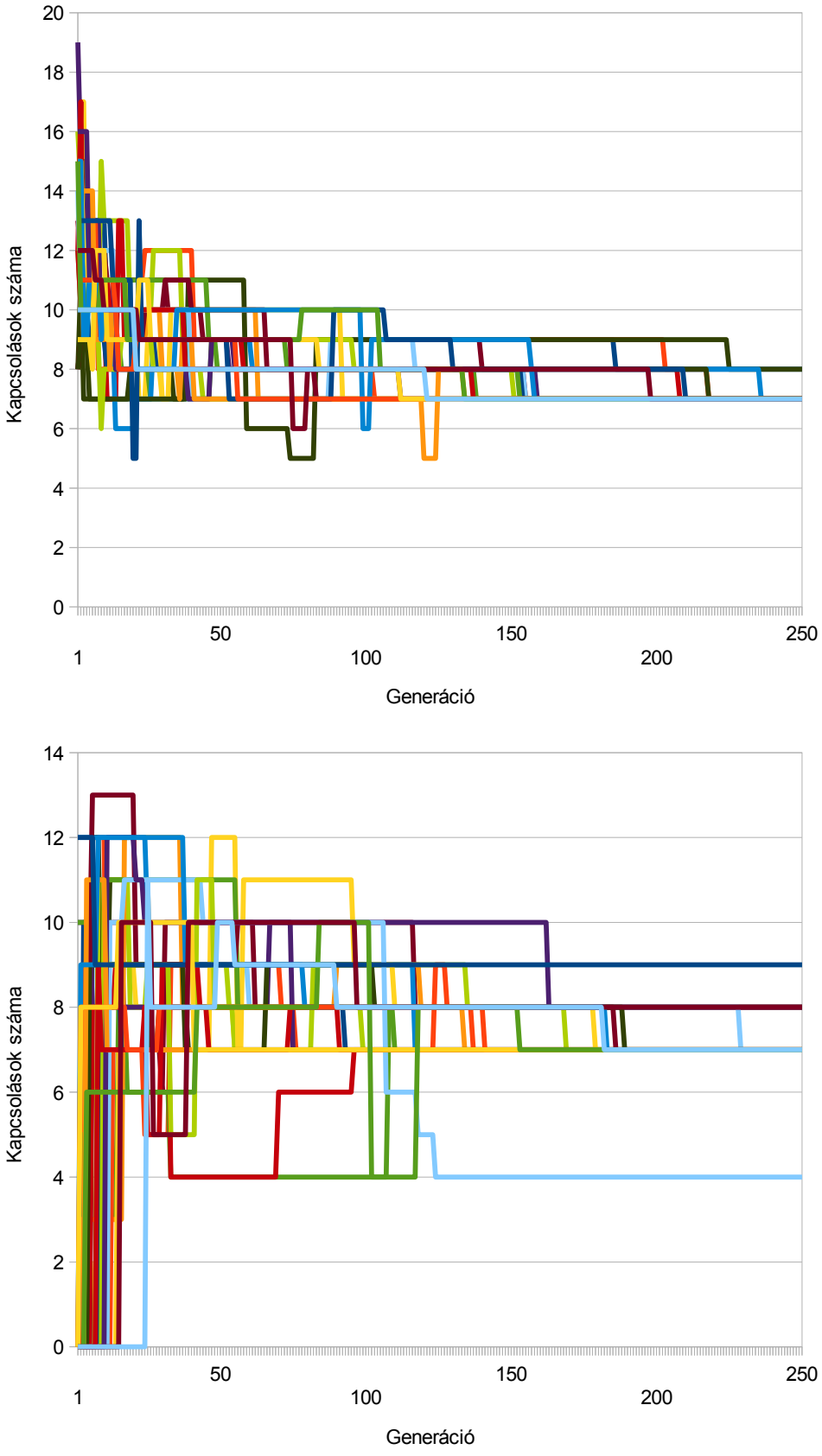
A következő példa (3.4. és 3.5. ábra) egy kisebb hiba esete (22-es kapcsoló nyitása a 2.8. ábrán szereplő teszthálózaton). A normál üzemállapot szerepeltetése a kezdő populációban ismét nem hozott jelentős előnyöket. Igaz, a kiesések jóval kisebb értékről indulnak, de lassabban találja meg a kevesebb kapcsolásos utat. Itt szerencsére mindig sikerült az optimális megoldást megtalálni: 1 kapcsolás, ellátatlan fogyasztó nélkül.



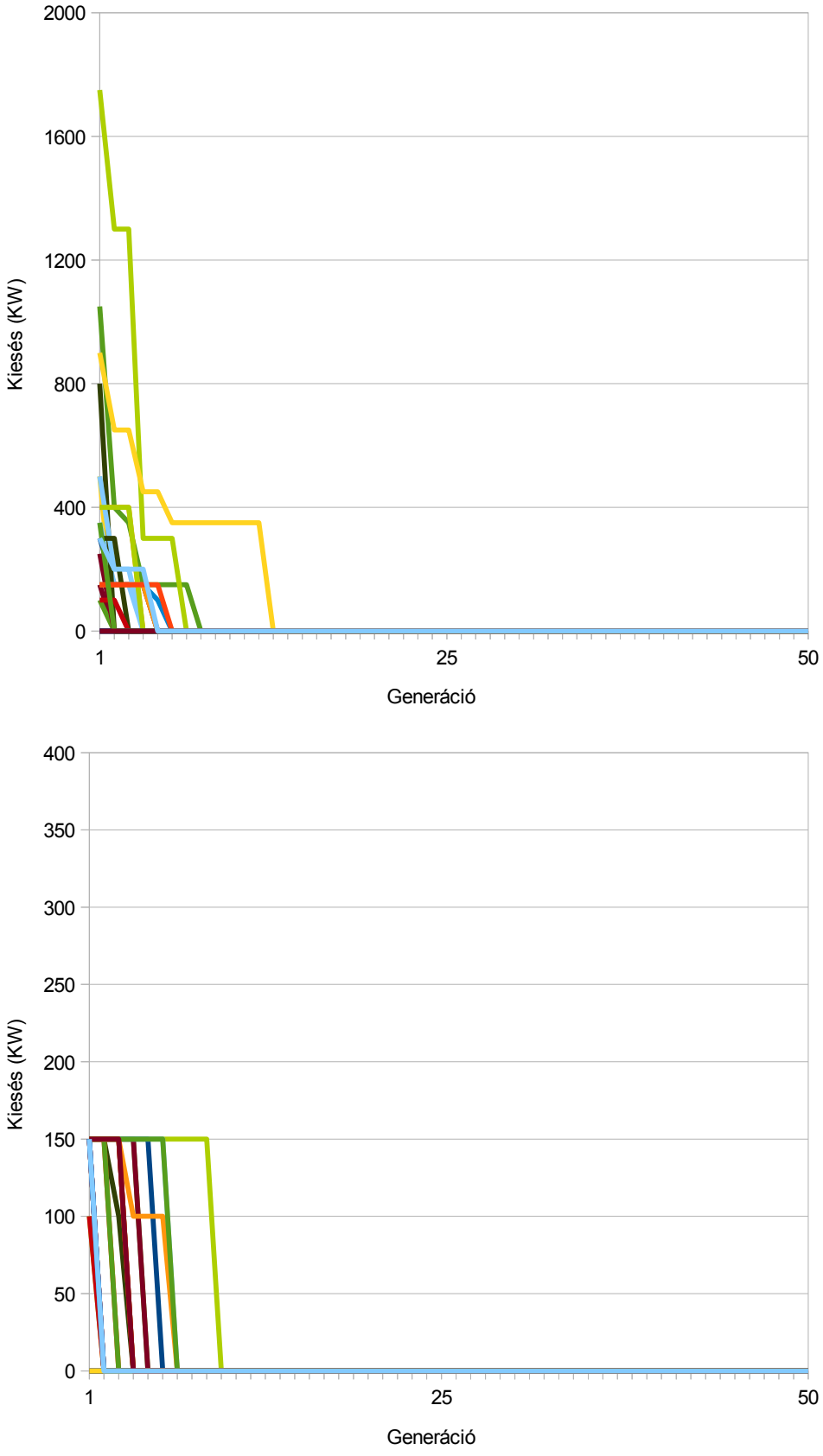
3.1. ábra. *Véletlenszerű kezdőállapotból a normál üzemállapot megtalálása.*



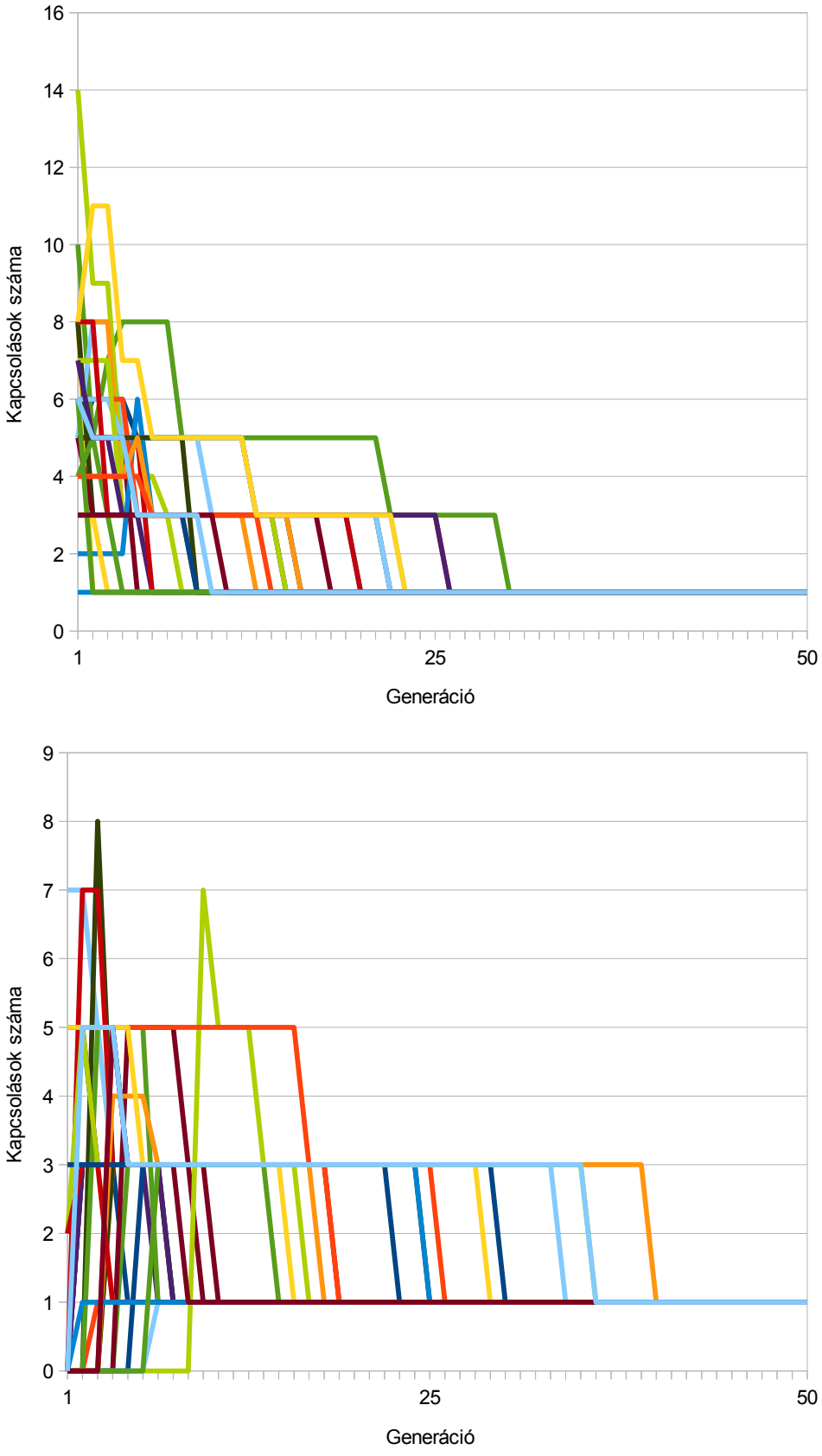
3.2. ábra. *Súlyos hiba megoldása, fent véletlenszerű kezdőállapotból, lent normál üzemi állapot szerepelt a kezdő populációban.*



3.3. ábra. *Súlyos hiba megoldása, fent véletlenszerű kezdőállapotból, lent normál üzemi állapot szerepelt a kezdő populációban.*



3.4. ábra. *Egyszerű hiba kezelése, fent véletlenszerű kezdőállapotból, lent normál üzemi állapot szerepelt a kezdő populációban.*



3.5. ábra. *Egyszerű hiba kezelése, fent véletlenszerű kezdőállapotból, lent normál üzemi állapot szerepelt a kezdő populációban.*

A 3.6. és 3.7. ábrán összehasonlítás látható a véletlenszerű indulás és a normál üzemállapottal indított esetek között a megoldások megtalálásának gyorsasága és minősége szempontjából (30 futtatás átlagai). Nem tapasztalható lényegi gyorsulás egyik indulási feltételtől sem. Emellett, mint fentebb láthattuk, a normál üzemi állapot szerepeltetése a kezdő populációban a korai konvergencia esélyét megnöveli. Elmondhatjuk tehát, hogy érdemes mindig teljesen véletlenszerű populációval indítani a keresést, legalábbis a vizsgált módszer a kezdő populáció megváltoztatására nem hatékony a példahálózati méretekkel rendelkező esetekben.

A program első része tehát rendkívül jó arányokkal rendelkezik az optimális megoldás megtalálásában (minden esetben 90 százalék feletti ezek aránya), ahol más lett a végeredmény, ott is legtöbbször nem áll túl távol az optimálistól a felkínált megoldás. Még a nagyobb, többszörös hibákat (például a 23-as és az 1-es kapcsoló nyitása) is jól kezelte az eljárás.

Nagyobb hálózaton értelemszerűen több generációra van szükség az optimális megoldás megtalálásához, de például a normál üzemállapot megtalálásához a 148 csomópontos hálózaton (igaz más típusú, mint a kisebb, itt együtt halad a kapcsolások száma és a kiesések a hibamentes hálózatnál) átlagban 90 generáció alatt eljut az optimumhoz, kis szórással. Tehát nincs szükség nagyságrendekkel több generációra nagyobb hálózatok esetén sem.

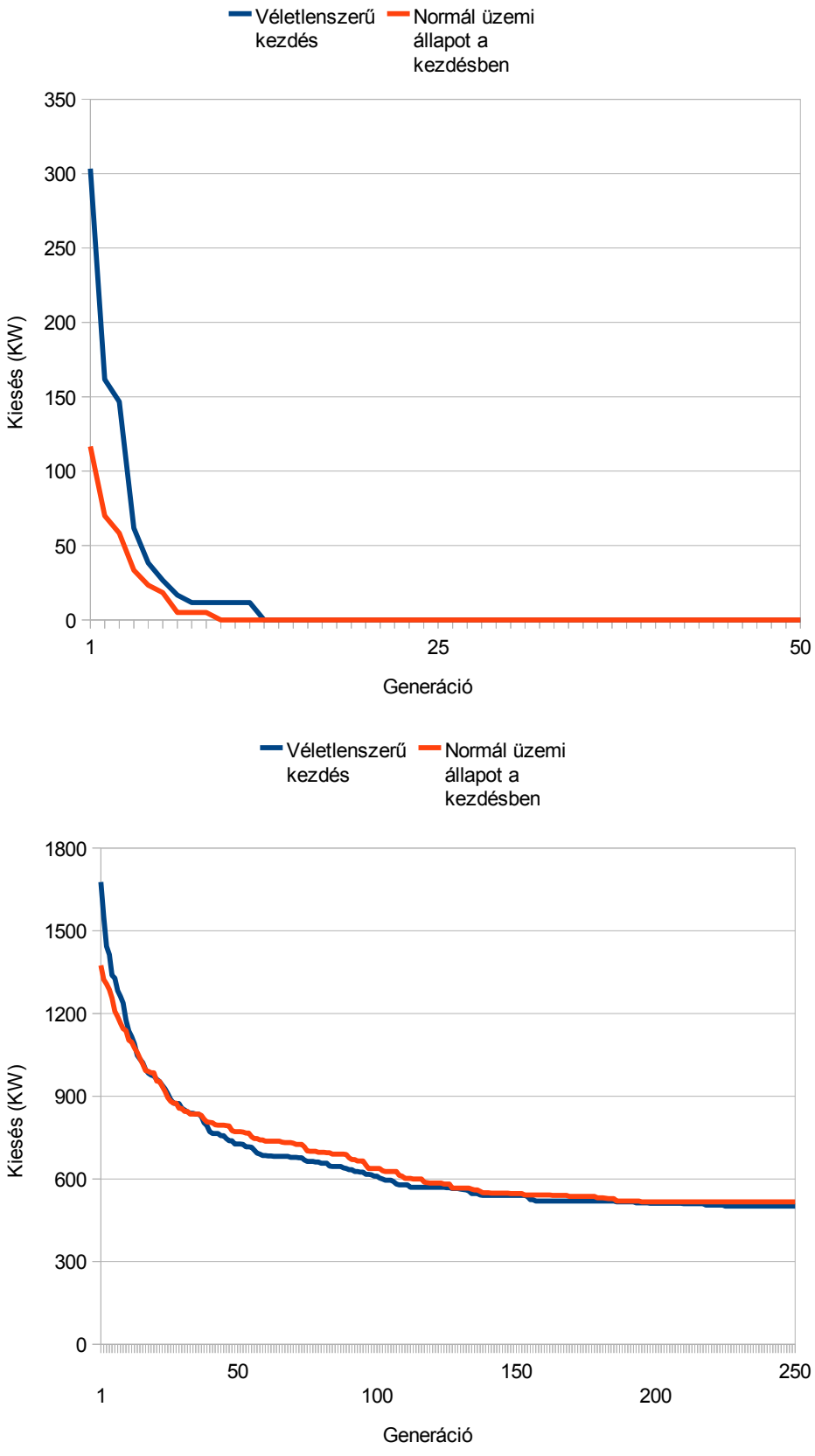
Természetesen minden hálózat más és más, ezért nehéz előrejelezni, hogy különböző esetekben hány generáció szükséges a megoldás megtalálásához.

Az eljárás már a mostani formájában is gyors. A genetikus algoritmus futása során 250*10 vagyis 2500-szor értékelt ki lehetséges hálózati képet ez kevesebb mint 2^{12}-en. Az összes lehetséges kapcsolóállás vizsgálatára 2^{28} kiértékelés kellene. Tehát nem lehetne rövid idő alatt minden lehetséges megoldást átnézni, nagyobb hálózatokon még reménytelenebb lenne.

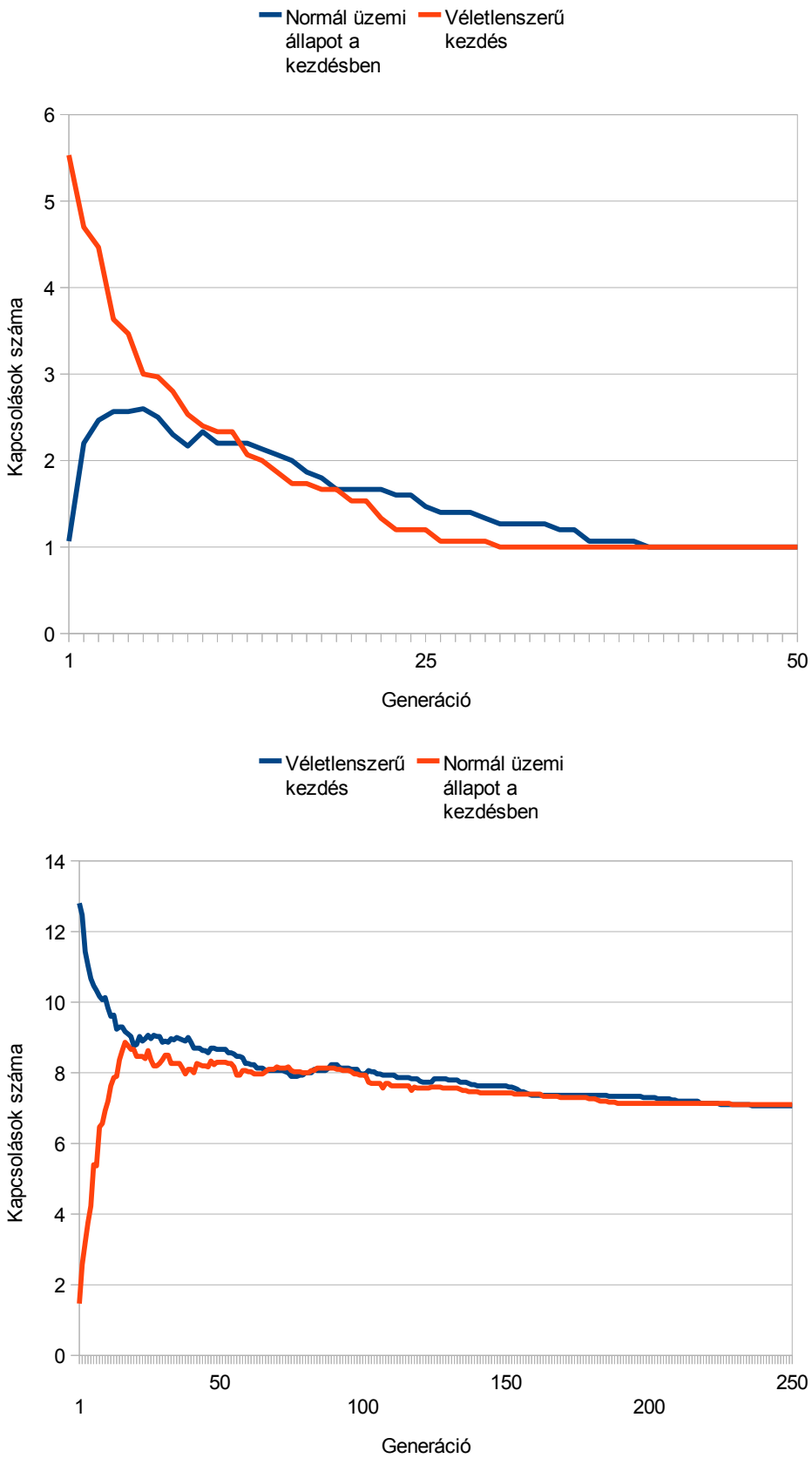
A második rész az első rész megoldásának kapcsolásait átvéve optimalizálja a kapcsolások sorrendjét. 200 generáció, 10 egyedes populáció, 0.8 keresztezési valószínűség és a vázolt mutáció szabályozás, lényegében hasonló alap paraméterekkel működik, mint az első rész. Itt most egy 8 kapcsolásból álló minta eset eredményei vannak.

A 3.8. ábrán ugyanaz a kapcsolási feladat látható egy és két csapat rendelkezésre állásánál. A gyors konvergencia itt is megfigyelhető, az esetek túlnyomó többségében itt is optimális megoldás az eredmény (28-29 eredmény a 30-ból), és ahol más eredmény adódott, az is közel volt az optimumhoz. A kiindulási értékekhez képesti 10-20 perc körüli javulás jelentősnek mondható. A javulás mértéke értelemszerűen függ a kapcsolások számától és azok elhelyezkedésétől. Az eredmények könnyen összehasonlíthatók a csapatok számának függvényében, ezáltal gyorsan mérlegelhető hány csapatra van szükség. A teszt feladatnál 1-8 csapatig rendre az optimális idők percben: 38, 21.3, 15.7, 14.7, 14.7, 14.5, 14.5, 10.7

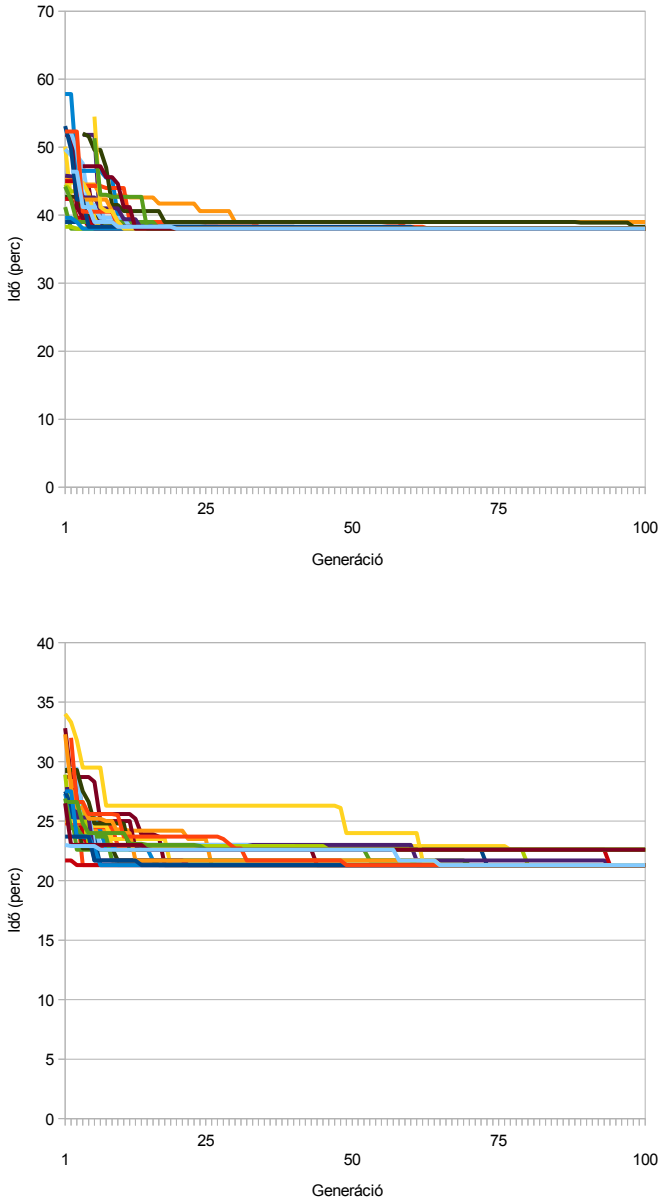
Ezen résznél is megéri genetikus algoritmust alkalmazni: a kiértékelt esetek száma: 2000 (2^{11}-nél kevesebb) A lehetséges esetek száma a csapatok számától függ. 2^8-tól 2^{15}-ig. Egy csapattal nyolc kapcsolásnál jobbnak látszik az összes lehetőség kipróbálása, de ne felejtsük el, hogy a legtöbb alkalommal 30 iteráción belül konvergált a megoldás ilyen esetben. Valamint előre nem tudjuk, hogy hány kapcsolás lesz majd szükséges és hány csapat áll



3.6. ábra. *Kiesések alakulása, fent egyszerű hiba, lent súlyos hiba.*



3.7. ábra. *Kapcsolások alakulása, fent egyszerű hiba, lent súlyos hiba.*



3.8. ábra. *Fent egy csapattal, lent két csapattal a kapcsolási idők alakulása a generációk előrehaladtával.*

rendelkezésre, így egyszerűbb mindig genetikus algoritmust használni. Az 1 másodperc körüli futások amúgy is a megoldás megalkotásának kisebbik részét teszik ki, a több időt igénylő rész mindig a kívánatos topológia kialakítása lesz.

3.2. Összegzés

Hazánkban jelenleg nem alkalmaznak tanácsadó rendszereket az optimális kapcsolások megtalálásához és külföldön is a jó módszerek megtalálásán fáradoznak.

Az eljárás segítségével gyorsítható az üzemzavar elhárítás és csökkenthető az érintett fogyasztások száma. Olyan szempontok is jobban kezelhetők mint a feszültségesés és túlterhelések kérdése, ezek számítógépes segítség nélkül nehezen becsülhetők. A szakértői rend-

szerekkel ellentétben nem szükséges szabályok felállítása a különböző hálózatokra (ami hosszadalmas és nagy feladat a karbantartása is), nem függ ezektől, így univerzálisabb. Adatigénye automatikusan kinyerhető a szolgáltatók adatbázisából. Nincs szükség a hálózaton változtatásokra ahhoz, hogy eredményesen használhassuk az üzemirányításban és tréningszimulátorokban. Nem determinisztikus jellege, mint láthattuk, nem okoz igazán nagy hátrányt, hiszen rendkívül nagy arányban optimális vagy ahhoz közeli megoldásokat kapunk.

A kidolgozott eljárás tehát jól működik, rugalmasan alakítható és elég gyors. Használata várhatóan javítja az villamos energia ellátás minőségét.

3.3. Kitekintés, fejlesztési lehetőségek

Az eljárás felhasználható tréningszimulátorban (például a diszpécserek képzésénél a véletlenszerű helyzetekben nyújtott önálló teljesítményük értékeléséhez nyerhetők közel objektív mutatók) vagy tanácsadó rendszerben.

A kidolgozott eljárás kevés módosítással alkalmas lehet más célok figyelembe vételére is: Például az első részben előnyben részesíthetjük a prioritásos fogyasztók ellátását is. Vagyis az új célfüggvény a nem ellátott prioritásos fogyasztók száma lenne, hogy illeszkedjen a többi minimalizálandó célhoz. Sőt a második részbe is belevehető, hogy ezen fogyasztók ellátásához szükséges kapcsolásokat a lehető legelőbb végezzük el.

Egy teljesen más felhasználás lehet a hálózati veszteségek minimalizálása. Ha más szempontokat is szeretnénk figyelembe venni e mellett, akkor az első rész az ideális alap, ha csak pusztán ez az egy cél van, akkor a második rész vázára érdemes építeni.

A második rész az idő minimalizálása helyett, használható a kiesett (nem szolgáltatott) energia minimalizálására, vagy a kiesett (nem ellátott) fogyasztók száma szerinti legkisebb kiesési időre a kapcsolások során.

Más problémára is használhatjuk az eredményeket: Ha egy adott hálózaton szeretnénk a távműködtethető kapcsolók optimális elhelyezését meghatározni akkor minden lehetséges zárlati helyre lefuttatva az algoritmust (az számít különböző helynek ami más kapcsolók működtetésével különíthető el) az eredményekből statisztikát készíthetünk, így láthatóvá válik, hogy melyik kapcsoló hányszor volt az optimális kapcsolási sorozat(ok)ban. Mivel tervezésről van szó a futásidő nem olyan kritikus. Itt érdemes nem csak a kiszemelt szempont szerinti legjobb megoldást, hanem a teljes utolsó szülő populáció megoldásait eltárolni.

Ez több megfontolásból is hasznos lehet: nem biztos, hogy mindig az adott szempont lesz az elsődleges. Lehet, hogy például ugyan 3 mellett 4 kapcsolásos megoldás is van az adott zárlati helyre, de ha a 4. kapcsoló távműködtetésű lesz az közel olyan gyors lehet, mint a 3 kapcsolásos ráadásul valószínűleg kisebb kiesett fogyasztást is eredményez, hiszen a késői generációkban akkor maradhat fent a több kapcsolós megoldás, ha más szempontból előnyösebb tulajdonságai vannak társainál. Ezenkívül egy másik helyzetben a legkisebb kapcsolással járó megoldásban is szerepelhet az a kapcsoló, így ott további gyorsulást eredményez.

Fejlesztési lehetőségek:

Egy adaptív szabályozás kialakítása az algoritmus saját paramétereire, amely futás közben dinamikusan alakítja ezeket. Konkrétan a mutációk és keresztezések valószínűségének, erősségének befolyásolására, valamint arra, hogy melyikek vezetnek gyorsabban célra az adott környezetben. Így egy mutáció és egy keresztezés operátor helyett lehetne sokkal többet alkalmazni egyszerre. Ez az egyik legnagyobb haszonnal kecsegtető lehetőség és egyben a legnehezebb feladat is. Sokat javulhatna a futásidő (kevesebb generáció is elegendő), a megoldások minősége is javulhatna, kevesebbszer maradhat lokális minimumokban az algoritmus.

Az inicializálás során további egyszerűsítése a vizsgálandó hálózatnak, ezáltal javulhat a teljesítmény.

Az algoritmus felkészítése az elosztott termelés tömegesebb megjelenésére.

A második rész kapcsolási precedenciájának finomítása, valamint bizonyos mértékű túlterhelések megengedése és kezelése a kapcsolások során.

A kapcsolási szám szétválasztása és külön kezelni a manuális és külön a távműködtethető kapcsolások számát.

A feltételek rugalmasabb kezelése: külön célként tekintve rájuk, vagy fuzzy módszerekkel kezelve.

Extrém üzemzavarok esetén tartósabban párhuzamos topológiát eredményező kapcsolások is szükségesek és indokoltak lehetnek a minél kisebb érintett fogyasztói szám érdekében. Ezek kezelésére felkészíteni az eljárást.

Köszönetnyilvánítás

A munka szakmai tartalma kapcsolódik a "Minőségorientált, összehangolt oktatási és K+F+I stratégia, valamint működési modell kidolgozása a Műegyetemen" c. projekt szakmai célkitűzéseinek megvalósításához. (ÚMFT TÁMOP-4.2.1/B-09/1/KMR-2010-0002 programja támogatja).

Irodalomjegyzék

[1] H.-T. Yang and C.-M. Huang. Distribution system service restoration using fuzzy petri net models. *International Journal of Electrical Power and Energy Systems*, 24(5):395–403, June 2002.

[2] S. Dimitrijevic and N. Rajakovic. An innovative approach for solving the restoration problem in distribution networks. *Electric Power Systems Research*, 81(10):1961–1972, October 2011.

[3] C.-M. Huang, C.-T. Hsieh, and Y.S. Wang. Evolution of radial basic function neural network for fast restoration of distribution systems with load variations. *International Journal of Electrical Power and Energy Systems*, 33(4):961–968, May 2011.

[4] S. Siqing, M. Zhigang, W. Jing, and G. Nan. distribution network fault restoration based on improved adaptive genetic algorithm. In *2009 Second International Conference on Intelligent Computation Technology and Automation*, pages 318–321, October 2009.

[5] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Kluwer Academic Publishers, 1989.

[6] Xinjie Yu and Mitsuo Gen. *Introduction to Evolutionary Algorithms.* Springer, 2010.

[7] D. Shirmoharmnadi, H. W. Hong, A. Semlyen, and G. X. Luo. A compensation-based power flow method for weakly meshed distribution and transmission networks. *IEEE Transactions on Power Systems*, 3(2):753–762, May 1988.

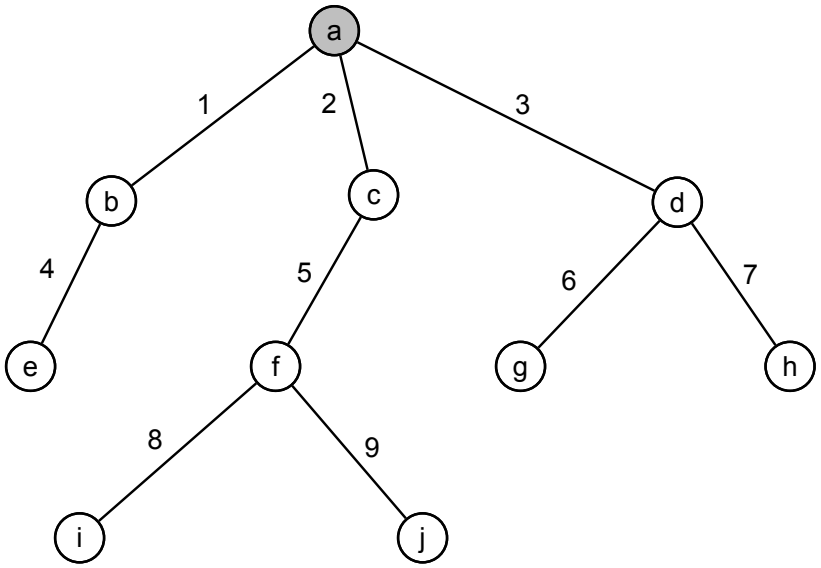
[8] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.*, 6(2):182–197, April 2002.

[9] M. Zhang, W. Luo, and X. Wang. Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences*, 178(15):3043–3074, January 2008.

Függelék

F.1. Az alkalmazott load flow rövid bemutatása

Az eljárás alkalmas gyengén hurkolt hálózatok számítására is (bővebben a [7] irodalomban), de itt csak az egyszerűbb sugaras hálózatra alkalmas implementációról írok, hiszen nem fordulnak elő hurkok az elosztó hálózaton. Az F.1.1. ábrán látható példahálózaton könnyebb követni a működést.



F.1.1. ábra. *Példahálózat a Load Flow bemutatásához*

A számítani kívánt hálózatot gráfként reprezentálva egy fába kell rendezni, amelynek a csúcsán a generátor van (a jelű csúcs az ábrán).

Minden csúcshoz:

- Tartozik valamekkora feszültség (V_k),
- ezenkívül tartozhat fogyasztás (wattos és meddő) (S_k).

Minden élhez:

- Tartozik ellenállás és reaktancia (R_l, X_l),
- és a rajta folyó áram értéke (I_l).

Kiinduláskénti szükségünk van a fogyasztásokra, a vezetékek paramétereire (ellenállás és reaktancia), és minden csúcshoz egy kezdeti feszültségre amiből kiindulunk (elég mindenhová ugyanakkora célszerűen a generátor feszültsége). Az eljárás szövegesen:

1. Minden élet sorszámozni kell, fentről indulva,

2. a hálózat végéről visszafelé haladva minden élre ki kell számolni a fogyasztás hatására kialakuló és a lentebbi ágakról érkező áramok összegét, például: $I_{5,i} = I_{8,i} + I_{9,i} + \frac{V_{f,i-1}}{S_f}$,

3. a hálózat tetejéről ki kell számolni a feszültségesések nyomán kialakuló feszültségeket minden csúcsra, például $V_{c,i} = V_{a,i} - (R_2 + jX_2) * I_{2,i}$,

4. minden csúcsnál ki kell számolni az aktuális feszültség és áram hatására létrejövő wattos és meddő teljesítmény értékét, például $S_{c,i} = (I_{2,i} - I_{5,i}) * V_{c,i}$

5. ezeket össze kell vetni a kívánt értékekkel, ha az eltérések egy előre meghatározott érték alatt (ϵ) maradnak akkor kész, ha nem akkor vissza a 2. pontra,

   $max(S_k - S_{k,i}) < \epsilon \qquad i = 1, 2, ..., 9 \;\; , \;\; k = b, c, ..., j.$

Sok esetben néhány iteráción belül (2-3) konvergál a megoldás egyszerű sugaras hálózatokon. A számítás rendkívül gyors, ezért különösen alkalmas a genetikus algoritmusok célfüggvény számításaihoz.

F.2. Fogalomtár

Adaptív szabályozás (evolúciós algoritmusokra) Az algoritmus futása közben az eddigi és jelenlegi populációk alapján a paraméterek dinamikus szabályozása.

Átviteli hálózat A villamos energia szállítására szolgáló nagyfeszültségű, erősen hurkolt hálózat. A nagytermelőktől az elosztó hálózatokig tart.

Buborék rendezés A rendezendő listában két egymás melletti elemet összehasonlítva a nagyobb/kisebb elem helyet cserél. Annyiszor kell végigmenni a listán ameddig van helycsere.

Diszpécser A hálózat üzemét felügyelő és szükség esetén intézkedő szakember.

Elitizmus A szelekciós folyamat során a populáció legjobb egyedei változatlanul kerülnek a következő generációba.

Elosztó hálózat A villamos energia elosztására szolgáló hálózat, főelosztó hálózat főleg a 120 KV-os feszültségszintű, az elosztó hálózat ez alatt egészen 400/230 V-ig. Sugaras alapvetően a topológia.

Evolúciós algoritmusok A módszer azt az evolúciót igyekszik megragadni, amellyel az élőlények környezetükhöz nagyszerűen alkalmazkodtak az elmúlt idők során. Kicsit átértelmezve a környezet a megoldandó feladat az élőlények pedig a lehetséges megoldások halmaza.

Feszültségesés A vezetéken folyó áram hatására azon feszültség esik, vagyis a két végén mért feszültség nem egyezik meg.

Fuzzy rendszerek A természetben, az emberi beszédben és mértékekben meglévő bizonytalanságok kezelésére alkalmas elmélet mellyel olyan kifejezésekkel számolhatunk mint a gyors vagy a lassú, a kicsi vagy a nagy. Nem szükséges egyetlen értékkel jellemezni ezeket a bizonytalan adatokat.

Heurisztikus módszerek Tapasztalatokon és sejtéseken alapuló megoldási módszer az olyan problémákra ahol egzakt, bizonyítottan legjobb megoldás megtalálása nem lehetséges valamilyen oknál fogva (Például futásidő miatt). Az így nyert eredmény valószínűleg nem a legjobb lesz, de jó módszer esetén közel lesz ahhoz.

Keresztág Egy-egy leágazás között vezet.

Kisfeszültségű hálózat Nálunk tipikusan a 400/230 V -os hálózat.

Középfeszültségű hálózat Nálunk tipikusan a 0.4 KV felett 35 KV-ig tartományba eső feszültségszintű villamos hálózat.

Lágy számítási módszerek Az olyan problémákra szánt megoldások gyűjtőfogalma amelyek nem oldhatók meg polinomiális időben. A pontos megoldás helyett közelítő eljárásokkal keresik a megoldást. Amely nagy valószínűséggel talál optimális megoldást is.

Leágazás Az alállomásról kiinduló ellátási körzet.

Nagyfeszültségű hálózat Nálunk tipikusan a 120 KV és a feletti feszültségszintű villamos hálózat.

Neurális hálózatok Az idegsejtek működésének ötlete vezérelte a kifejlesztésüket. Sok egymással kapcsolatban lévő mesterséges neuronból áll amik utánozzák a valós működést. Nem szükséges a vizsgált probléma, rendszer ismerete a megoldáshoz. Elegendő tanításukhoz a bemenetekhez tartozó kimenetek megléte.

NF/KöF Transzformátor Nagyfeszültségű és középfeszültségű hálózatrészt összekötő elem.

Okos mérők Olyan mérők a fogyasztóknál amik a hagyományos mechanikus teljesítménymérőkön túlmutató képességekkel rendelkeznek.

Petri háló Olyan hálózat (gráf) ahol csak események, állapotok és élek vannak. Él csak esemény és állapot vagy állapot és esemény között vezethet. Tokenek segítségével minden ütemben kiértékelhető a rendszer állapota.

Populáció A szülők vagy az utódok száma egy generációban.

Prim algoritmus A legkisebb súlyú feszítő fát keresi egy gráfban.

Prioritásos fogyasztók Például kórházak, rendőrség, vagy olyan fogyasztók amiknek a leállása vagy hosszabb szüneteltetése esetén komoly károkkal kell szembenézni például alumínium kohó.

r,x értékek Ellenállás és reaktancia értékek.

RBF Neurális hálózat Radial Basis Function neural network. Sugaras (radiális) bázisfüggvényű neurális hálózat, az egyes összeköttetések súlya a neuronok elhelyezkedésétől függnek.

SCADA Supervisory Control And Data Acquisition system rövidítése. Távvezérlésre és adatgyűjtésre szolgáló rendszer.

Smart grid A jövő villamos energia hálózata, amely nagyfokú automatizáltsággal, hibatűréssel, robusztussággal rendelkezik. Több definíció is létezik.

Smart metering Okos mérők és szenzorok használata a hálózaton.

Szakértői rendszerek Két részből áll: az értelmező rendszerből és a szabályokból. A vizsgált problémára a szabályok alapján próbál megoldást találni. Az emberi gondolkodás utánzása, szakértők tudásának formalizálása az alapja.

Tréningszimulátor A diszpécserek képzésére kialakított környezet, ahol az éles rendszerre jellemző reagálással de az azon való tréningeztetés veszélye nélkül lehet gyakorolni.