Hibadiagnosztika elosztott rendszerekben

Bevezető

A mérés során megvalósítandó feladatok két csoportba oszthatók: determinisztikus és valószínűségi diagnosztikai algoritmusokra.

- A **determinisztikus algoritmusok** adott feltételek teljesülése esetén képesek teljes és konzisztens diagnosztikai képet előállítani. *Teljesnek* akkor nevezünk egy diagnosztikai eredményt, ha a rendszer minden egysége kapott valamilyen (hibátlan/hibás) minősítést, *konzisztensnek* pedig akkor, ha minden egység valós és az algoritmus által feltárt hibaállapota megegyezik. A teljes és konzisztens diagnosztika eléréséhez teljesítendő további feltétel(eke)re példa a mérési segédletben tárgyalt, a hibás egységek számára adott felső korlát, az ún. t-korlát.
- A **valószínűségi algoritmusok** nem garantálnak biztosan teljes és konzisztens megoldást, de nagy valószínűséggel helyes eredményt szolgáltatnak. A tévedés lehetőségért cserébe ezek a módszerek csak a teszteredményekből kinyert információt használják fel, azaz nem igényelnek a rendszerre nézve olyan további előírásokat, mint a t-korlát a determinisztikus módszereknél.

  A valószínűségi algoritmusok lehetséges diagnosztikai tévedései három csoportba sorolhatók:

  1. Az *ismeretlen egység* egy olyan processzort jelent, amelynek hibaállapotát az algoritmus nem tudta meghatározni. Ebben az esetben a diagnosztika nem teljes.
  2. *Jóindulatú tévedés* esetén egy hibátlan processzort hibásnak minősítünk. Így csökkentjük ugyan a rendelkezésre álló erőforrások számát, de a biztonság javára tévedünk.
  3. *Rosszindulatú tévedés* esetén egy hibás processzort hibátlannak minősítünk. Ekkor a rendszerből nem távolítjuk el a hibás komponenst, ami hibás adatokkal és hibaterjesztéssel az egész folyamat hibáját okozhatja. Az erőforrások számát nem csökkentjük.

A mérési feladat

A mérési feladat a következő pontban leírt valószínűségi diagnosztikai algoritmus megvalósítása egy C nyelven megírt diagnosztikai szubrutin formájában. A szubrutint a szimulációs környezetbe ágyazva le kell fordítani, majd az algoritmus helyességét próbafuttatásokkal ellenőrizni. (A futtatások eredményét statisztikailag értékelve a mérési jegyzőkönyvben is fel kell tüntetni!)

A mérés kiértékelése

A valószínűségi algoritmusok „tévedhetnek”. A jó- vagy rosszindulatú tévedések száma az adott valószínűségi algoritmus fontos jellemzője. Ezért a valószínűségi módszerek kiértékelésekor az algoritmus értékeléséhez több pontban elvégzett mérések alapján (adott rendszerméret esetén növekvő hibaszám, vagy adott számú hiba mellett növekvő rendszerméret, stb.) adjon statisztikát (táblázatos és diagram alakban) a jóindulatú és rosszindulatú diagnosztikai tévedések (és ha van, az ismeretlen egységek) számáról!

Szorgalmi feladat: ha tud, adjon ötleteket az adott algoritmus továbbfejlesztéséhez (és ha van idő, valósítsa is meg)! Cél, hogy a diagnosztikai tévedések száma csökkenjen.

A megvalósítandó algoritmus leírása

Somani és Agarwal valószínűségi diagnosztikai algoritmusa (a Dahbura algoritmushoz hasonlóan) a szomszédok által végzett teszteredményeket számlálja és szintén iterációkat használ. Vezessük be a következő jelöléseket: *LG* jelölje a *valószínűleg hibátlan* és *LF* a *valószínűleg hibás* egységek halmazát, valamint

$$\begin{aligned} K_0(u_i) &= \{u_j : u_j \in \Gamma^{-1}(u_i) \cap LG, t_{j,i} = 0\} \\ K_1(u_i) &= \{u_j : u_j \in \Gamma^{-1}(u_i) \cap LG, t_{j,i} = 1\} \end{aligned}$$

ahol a $\Gamma^{-1}(u_i)$ halmaz az u_i egység tesztelőinek halmazát jelöli.

Az algoritmus működése a következő:

1. Első lépésben az algoritmus minden egységhez egy konfidencia-szintet rendel. Egy processzor „valószínűleg hibátlan", ha őt a tesztelőinek legalább fele jónak tartotta:

   $$|\{u_j : u_j \in \Gamma^{-1}(u_i), t_{j,i} = 0\}| > |\Gamma^{-1}(u_i)|/2$$

   egyébként „valószínűleg hibás".

2. Második körben az algoritmus már csak a valószínűleg hibátlan egységeket veszi figyelembe. Hibátlannak minősíti azokat az egységeket, amelyekre

   $$|K_0(u_i)| - |K_1(u_i)| \geq t/2.$$

3. Az így keletkezett hibátlan minősítésű egységek szomszédait a lokális teszteredmények alapján a hibásak vagy hibátlanok közé sorolja. Itt kétféle következtetési séma alapján járja be és minősíti a szomszédokat az algoritmus:

   - Előre következtetés: a teszteredmények alapján a hibátlan egység minősíti az általa tesztelt processzorokat.
   - Vissza következtetés: a hibátlan egységet hibás teszteredménnyel tesztelő processzorokat hibásnak minősíti.

4. Ellentmondás-mentesség ellenőrzése: ha a hibátlan egységből kiindulva két külön következtetési láncon egy adott processzora két különböző minősítés adódik, akkor *ellentmondásra* jutottunk. Ez azt jelenti, hogy a kiinduló feltételezésünk (hogy az adott egység hibátlan) nem állja meg a helyét, azaz ez a processzor *biztosan hibás*. Ennek következményeit vissza kell vetíteni a diagnosztikára.

Az eljárást addig folytatja, amíg minden processzorhoz ellentmondás-mentesen diagnosztikai címkét nem rendel. Végül a megmaradó „valószínűleg hibás" és „valószínűleg hibátlan" címkéjű egységeket rendre hibásnak és hibátlannak diagnosztizálja.
Valósítsa meg a Somani, Agarwal algoritmust!