1. Üzenetek nagyon távolra

Valahol a világűrben egy űrhajó száguld a Mars felé. Kapcsolatban van a földi irányítóközponttal, ahol a nagy teljesítményű számítógépek hatalmas munkát végeznek, feldolgozzák a mérési eredményeket, pályakorrekciókat számolnak és így tovább. A kommunikáció viszont nagyon drága dolog, az üzeneteket kódolni kell, mindez időbe telik, sok energiát fogyaszt. Egyáltalán nem mindegy, hosszú vagy rövid üzeneteket váltanak-e a felek. Nagyon fontos kérdés, hogy miképpen lehet gazdaságosan kommunikálni; erről lesz szó az alábbiakban.

Nézzünk először egy nagyon egyszerű alapfeladatot ebből a témakörből. Az űrhajó fedélzeti számítógépének alapprogramját időről időre ellenőrizni kell a Földön, hiszen a kozmikus sugárzás kárt tehet benne. Maga a program lényegében egy hosszú, nullákból és egyesekből álló sorozat; elvben semmi akadálya, hogy ezt a 0–1 sorozatot sugározzák vissza a Földre, ahol aztán egybevethető az ott biztonságosan tárolt példánnyal. A gond csak az, hogy ez a sorozat rettenetesen hosszú – mondjuk 10^{20} bitből áll – a teljes sorozat visszaküldése tehát gyakorlatilag lehetetlen.

Az első gondolatunk az lehet, hogy annak a természetes számnak a felhasználásával próbáljuk tömöríteni az információt, amelynek ez a 0–1 sorozat a kettes számrendszerbeli alakja – például úgy, hogy prímtényezőkre bontjuk és csak az ügyesen kódolt prímfelbontást sugározzuk vissza. Meg lehet azonban gondolni, hogy ezzel lényegében nem nyerhetünk, 10^{20} bitnél olcsóbban nem oldható meg a teljes információ visszaküldése. Hogy ne legyen az üzenet ilyen hosszú, valamiben engedni kell. Legyen ez a biztonság. Tegyük fel, hogy csak 99%-os biztonsággal szeretnénk tudni, nem sérült-e meg a program. Ezt persze nem úgy érdemes csinálni, hogy a sorozat egy véletlenszerűen választott jegyét – vagy jegyeit – küldjük vissza ellenőrzésre, hisz ekkor jegyenként $1/10^{20}$ a valószínűsége, hogy hibás jegyre akadunk.

Jelölje az űrhajó programjának megfelelő számot X, a Földön tárolt szám pedig – aminek tehát a valódi program a kettes számrendszerbeli alakja – legyen Y. A módszer legyen a következő.

A Földön választanak egy korlátot, legyen ez N, majd később eldöntjük, mekkora legyen – és ezután véletlenszerűen kiválasztanak egy p prímszámot 2 és N között. Ezt elküldik az űrhajónak, ahol kiszámolják az X maradékát p-vel osztva – legyen ez X_1 – és ezt visszaküldik a Földre, ahol egybevetik az ott kiszámolt Y_1-gyel. Ha a két maradék nem egyenlő, akkor a program biztosan rossz. Elképzelhető azonban, hogy a maradékok egyenlők, és a program mégis rossz. Ez úgy lehetséges, ha a véletlenszerűen választott p prímszám osztója az $X-Y$ különbségnek. Próbáljuk megbecsülni, hogy mekkora lehet ennek a valószínűsége. Ha az N-nél nem nagyobb prímek számát szokás szerint $\Pi(N)$-nel jelöljük, akkor a szóban forgó valószínűség éppen

$$P_{\text{nem vesszük észre a hibát}} = \frac{|X-Y| \text{prímosztóinak a száma}}{\Pi(N)}. \tag{1}$$

A kérdés úgy hangzik, hogy mekkorára válasszuk az N értékét ahhoz, hogy a fenti valószínűség $0,01$-nél kisebb legyen.

A számlálóban X és Y legfeljebb 10^{20} bitből álló számok, így

$$|X-Y| < 2^{10^{20}}$$

Ha $|X-Y|$ prímtényezős felbontása $p_1^{\alpha_1}p_2^{\alpha_2}\dots p_k^{\alpha_k}$, akkor (1) számlálójában éppen ez a k, az $|X-Y|$ prímtényezőinek a száma áll. Ha most a felbontásban minden k_i helyére kettőt, minden α_i kitevő helyére pedig 1-et írunk, akkor egy – elég durva – alsó becslést kapunk $|X-Y|$-ra, ahonnan

$$2^k < 2^{10^{20}},$$

tehát az $|X-Y|$ különbségnek legfeljebb 10^{20} prímosztója lehet – ami persze még mindig elég nagy.

A hiba valószínűségére innen a

$$\frac{10^{20}}{\Pi(N)}$$

felső becslést kapjuk. Ahhoz, hogy ez kisebb legyen $\frac{1}{100}$-nál,

$$10^{22} < \Pi(N) \tag{2}$$

szükséges, tehát az, hogy 1 és N között legalább 10^{22} prímszám forduljon elő. Egy igen mély számelméleti eredmény, az úgynevezett *prímszámtétel* szerint az N nagy értékeire $\Pi(N)$ „körülbelül” $N/\log_e N$. Innen az adódik, hogy (2) teljesül, ha N körülbelül 24 jegyű szám, ami kettes számrendszerben körülbelül 80 jegy. Ez azt jelenti, hogy 80 bit információ visszaküldésével 99%-os biztonsággal eldönthető, nem sérült-e meg az űrhajó programja, ez pedig óriási nyereség a teljes 10^{20} bithez képest. Az is látható, hogy a biztonság növelése – 99,99% például – a kitevőt növeli (2)-ben – most például 2-vel – ez pedig a prímszámtételből következően csak néhány bitnyi hossznövekedést jelent.

2. Üzenetek nagyon közelre

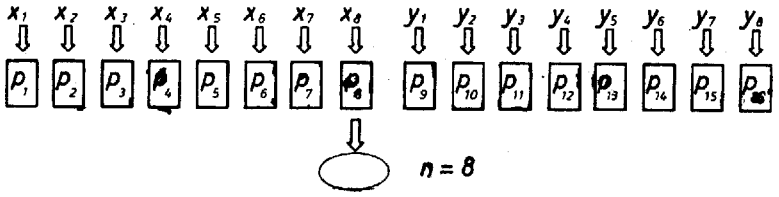
Az űrhajó kommunikációs feladata az egyik fő forrása ezeknek a kérdéseknek. Egy másik ilyen forrás a nagyméretű integrált áramkörök vizsgálata.

Ma már ott tart a technológia, hogy egy-egy chipen irdatlan mennyiségű áramköri elemet tudnak elhelyezni. Ezek között adott típusú huzalozást kell építeni. Kiderült, hogy az áramkör méretének a csökkentése során – a működés sebességének növeléséhez erre szükség van, hisz elvi korlát az áram terjedésének a sebessége – ezeknek a huzalozásoknak a tervezése az egyik legnagyobb probléma. Az áram nyomán ugyanis hő fejlődik, és így kis térfogatra nem lehet túl hosszú huzalrendszert bezsúfolni.

Nézzük a következő, a korábbira némileg emlékeztető feladatot. Olyan áramkört kell terveznünk, amely két n-hosszúságú bitsorozatról eldönti, egyenlők-e, vagy sem – tehát a földi ellenőrzés feladatát valósítja meg.

Az áramköri egységekről feltesszük, hogy rendkívül egyszerű „szerkentyűk", egy-vagy két bejövő bitet tudnak kezelni, ezekből kiszámolnak valamit, és továbbküldik egy következő processzornak.

Az 1. ábrán látható egyszerű berendezés például a következőképpen oldja meg a feladatot. Kezdetben minden processzor „alszik", kivéve p_{n+1}-et.



1. ábra

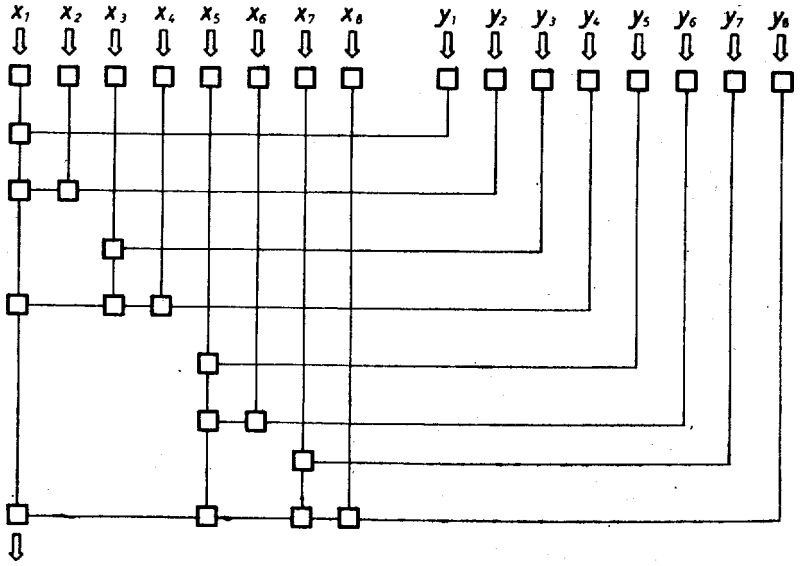
Az első taktusra a p_{n+1} processzor – az ábrán p_9 – leolvassa az y_1 bitet, elküldi balra és egyidejűleg jobbra küld egy „ébresztő" szignált. A következő lépésben a felébresztett p_{n+2} leolvassa y_2-t, balra küldi p_{n+1}-nek és felébreszti p_{n+3}-at. Általában a processzorok a jobbról érkező jeleket a következő taktusban balra küldik tovább, ezen kívül balról érkező szignál hatására a második n darab processzor mindegyike a következő lépésben a bejövő bitet balra küldi, és jobbra küld egy szignált. Ennél bonyolultabb működést nincs okunk feltételezni a processzorainkról. Ennek hatására a beérkező y jelek „egyes térközzel" masíroznak balra. Amikor a p_1 megkapja y_1-et, leolvassa x_1-et és összehasonlítja őket; jobbra küld egy 1-est, ha ezek egyenlők és 0-át, ha nem. Ez a b_1 jel épp akkor érkezik p_2-be, amikor az jobbról megkapja y_2-t. Ez a processzor kiszámolja az $x_2 = y_2$ összehasonlítás b_2 eredményét és jobbra küldi a $b_1 \cdot b_2$ szorzatot. Ez akkor és csak akkor lesz 1, ha $x_1 = y_1$ és $x_2 = y_2$. Nyilvánvaló, hogy a p_n processzor tudja majd a teljes választ: ott ismerjük meg a $b_1 b_2 \dots b_n$ szorzat értékét, ami pontosan akkor 1, ha a két sorozat bitről bitre azonos. A dolog tényleg igen egyszerű, egyetlen hátulütője, hogy kicsit lassú. Elég sokáig vándorolgatnak a jelek, látható, hogy éppen

$$T = 2n$$

taktusra van szükség. Eközben a huzalok összhossza nyilván

$$A = 2n - 1.$$

A 2. ábrán látható elrendezés a feladatot jóval gyorsabban, viszont lényegesen hosszabb huzalrendszer felhasználásával oldja meg.



2. ábra

Az egyszerűség kedvéért tegyük fel, hogy az n kettőhatvány – a gyakorlatban úgyis ez a helyzet. Ekkor a 2. ábra első átlójában levő 2^k processzor mindegyike összehasonlítja a megfelelő x_i-t és y_i-t – egyetlen taktusban szimultán – majd az átlóként feleződő számú processzorok ezeket a részeredményeket dolgozzák fel, és végül a $(k+1)$-edik taktusban a bal alsó processzor a teljes választ ismeri.

A feldolgozáshoz szükséges idő így $T = 1 + \log_2 n = \log_2 2n$-re rövidült, ennek viszont az lett az ára, hogy a huzalrendszer összhossza megnőtt. Könnyű számolási gyakorlat, hogy a terület – így hívják a drótok hosszának összegét – most körülbelül $A = 2n^2$. Felvetődik a kérdés, készíthető-e jobb áramkör ennek a feladatnak a megoldására.

*

Feladat. Tervezzünk olyan áramkört a fenti feladat megoldására, ahol az idő körülbelül $\log_2 n$ (itt valami konstans szorzó megengedett), a terület pedig körülbelül $n^2 \log_2 n$.

Ami számunkra érdekes és általában igaz, az egy bizonyos „osztozkodási" tétel, ami azt mondja ki, hogy az adott feladat megoldására tervezett áramköröknél ez a két mennyiség csak egymás rovására növelhető.

Tétel. Ha egy A területű áramkör T idő alatt dönti el két n-hosszúságú 0–1 sorozat egyenlőségét, akkor

$$AT \geqq n^2.$$

Bizonyítás. Képzeljünk el egy áramkört, és vágjuk függőlegesen két részre a chipet: az egyik rész tartalmazza az x_i, a másik pedig az y_i jeleket fogadó processzorokat. Mivel a két n hosszúságú sorozatot valahogy „össze kell hozni", a döntés meghozataláig ezen a vágáson legalább n bitnek át kell jutnia. Erre összesen T taktusnyi idő áll rendelkezésre, van tehát olyan lépés, amikor egyszerre $\frac{T}{n}$ bit jut át a vágáson, amihez persze legalább $\frac{T}{n}$ darab drótra van szükség. Ez persze csak annyit ad, hogy $AT \geqq n$. De most menjünk tovább. Vágjuk most el a chipet az első vágástól 1-gyel balra. Ekkor x_1 és y_1 persze egy oldalra kerülhetnek, de az így szétvágott chipben a két rész egyike sem tud semmit a „másik fél" $(n-1)$ hosszúságú sorozatáról $(x_2 x_3 \ldots x_n)$, ill. $(y_2 y_3 \ldots y_n)$, így az előzőek szerint ezt a vágást legalább $\frac{n-1}{T}$ darab drótnak kell kereszteznie ahhoz, hogy T taktus után megszülethessen a döntés. Ugyanez persze arra a vágásra is igaz, amit az elsőtől 1-gyel jobbra húzunk. Ha most egyesével tekintjük jobbra és balra a vágásokat, akkor a drótok összhosszára – és itt csak a függőlegesen átvágott drótokat számoltuk – a következő alsó becslés adódik:

$$A \geqq \frac{n}{T} + 2\frac{n-1}{T} + 2\frac{n-2}{T} + \ldots = \frac{n^2}{T},$$

és éppen ezt akartuk bizonyítani.

A megadott két áramkör tehát ebben az értelemben optimális: AT értéke mindkét esetben $c \cdot n^2$. Valamilyen értelemben a két szélsőséget jelentik: legalább $2n$ drótra már a $2n$ darab bemenő jel miatt is szükség van, másfelől – és ennek igazolása egy picit nehezebb – még a legbonyolultabb huzalozással sem vihető a döntéshez szükséges idő $\log_2 n$ alá.

Jegyezzük még meg, hogy nyilvánvalónak vettük, hogy a két rész között legalább n bit kommunikációjára van szükség. Itt kapcsolódik a kérdés az űrhajó problémájához. Ott láttuk, hogy ha nem törekszünk teljes biztonságra, akkor már lényegében $\log_2 n$ bit is elegendő – ez a megoldás azonban az integrált áramkörök tervezése során elesik, mivel a korábban leírt véletlen módszer alkalmazása jóval idő- és eszközigényesebb, mint ami az integrált áramkörök tervezésekor megengedhető.

3. A feladat általános megfogalmazása

Sokszor segít, ha az adott problémát megpróbáljuk viszonylag általános módon megfogalmazni. A fenti feladatok két szereplője – az űrhajó és a Föld, illetve a chipnek az x és y sorozatokat „ismerő" részei – külön-külön bizonyos, a másik fél által nem ismert információ birtokában „szeretnének" valami együttes döntést hozni. Az általánosság megszorítása nélkül föltehető, hogy a két fél – legyenek ők A és B – egy-egy n hosszúságú 0–1 sorozatot ismer – legyenek ezek X és Y – és ebből valamilyen $C(X,Y)$-nal jelölt mennyiséget szeretnének kiszámítani. Egy játéknak is tekinthető a dolog a két fél, A és B között. Ilyen például a bridzs, amikor a két partner ismeri a saját lapját és a licit során az adott szabályok szerint kommunikálva el szeretnék dönteni, hogy kettejük lapjában benne van-e például a nagy szlemm. (A példa persze nem egész pontos, hiszen a másik két játékos licitálását is hallják).

Az is feltehető, hogy ez a $C(X,Y)$ egyetlen bit – az űrhajó feladatában például 1, ha $X = Y$ és 0 egyébként. Ezt az esetet jellemezhetjük egy táblázattal. A táblázat sorai az A, oszlopai pedig a B lehetséges 0–1 sorozatainak felelnek meg – esetünkben tehát 2^n darab sora és ugyanennyi oszlopa van a táblázatnak – az i-edik sor j-edik mezőjében pedig az a $C(X_i, Y_j)$ érték áll, amelyet a két félnek valamilyen pár beszéd során meg kell határoznia, ha az A az X_i, a B pedig az Y_j sorozatot ismeri. A teljes táblázatot – mátrixot – persze mindketten ismerik, csak azt nem tudják a beszélgetés kezdetén, hogy a másik fél számára melyik sor, illetve oszlop van megadva.

Az űrhajófeladat mátrixa igen egyszerű szerkezetű: az átlóban 1-esek állnak, másutt pedig 0, hiszen föltehető, hogy a játék kezdete előtt A és B megállapodnak abban, hogy ugyanabban a sorrendben sorolják föl a sorozataikat. Látható, hogy n bit elegendő a feladat megoldásához – például úgy, hogy A elküldi a sorának a számát – ez B-nek elegendő, hisz magát a mátrixot mindketten ismerik.

Annak bizonyítására, hogy ennél kevesebb bit nem elegendő $C(X,\ Y)$ kiszámításához, „földönkívüli" segédeszközhöz folyamodunk. Az űrhajónál maradva tegyük fel, hogy egy szuperlény, egy ET is belehallgat a beszélgetésbe, átlátja a helyzetet és szeretne segíteni a résztvevőknek a beszélgetés lerövidítésében. Nos, ha a két program nem azonos, akkor ET-nek könnyű dolga van. Ránéz a földi programra, ránéz a rakéta programjára és rögvest feltűnik neki, hogy a 97. bit nem azonos a két programban. Közli az űrhajóval, hogy küldjék el ezt a bitet a földi központnak, hogy a két érintett fél is pillanatok alatt meggyőződhessen a hibáról. Ehhez lényegében a hibás bit sorszámát kell közölnie az űrhajóval, tehát a teljes n hosszúságú bitsorozat helyett egy legfeljebb n-jegyű szám nevét, ami körülbelül $\log_2 n$ bit. Például ha $n = 10^{20}$, akkor ez kb. $60 - 70$ bit. A hibás bit sorszáma alapján aztán az űrhajó és a földi központ már ennek az egyetlen további bitnek az elküldésével meggyőződhet arról, hogy a program valóban megsérült.

Mi a helyzet azonban akkor, ha a két program egyenlő. Tud-e az ET most is segíteni? Ha hisznek neki, akkor persze tud. A dolog azonban nemcsak bizalom kérdése. Mindkét félnek bizonyítékra van szüksége, ugyanúgy, mint az előbb. Azt állítom, hogy ilyenkor még az ET természetfölötti adottságai sem elegendők e mindkét fél számára meggyőző kommunikáció lerövidítéséhez.

Vizsgáljuk a kérdést tetszőleges C mátrixra. Az ET segítsége azt jelenti, hogy a két játékos, A és B bármely X sorához és Y oszlopához van az ET-nek olyan Z üzenete, amelyik az X sor és az Y oszlop metszéspontjában álló $C(X,Y)$ bit bizonyítéka mind az A, mind pedig a B számára. ET természetesen a mátrix mindegyik 1-esére bizonyítékkal tud szolgálni a feleknek. Egy adott Z üzenetére vannak a táblázatnak olyan sorai, illetve oszlopai, hogy A és B ezek bármelyike esetén a metszéspontban álló 1-es bizonyítékának fogadja el a Z üzenetet. Ekkor persze ezen sorok és oszlopok valamennyi metszéspontjában – a mátrix ún. részmátrixában – egyesek állnak. Ha ugyanis az A valamelyik, a Z-hez tartozó X sorát a B egyik Z-hez tartozó oszlopa 0-ban, a másik pedig 1-ben metszené, akkor amennyiben az A-nak ez az X a sora, akkor számára az ET üzenete nem bizonyíték.

Ha most az ET minden olyan X-re és Y-ra, amelyre $C(X,Y) = 1$, legfeljebb t bit hosszúságú bizonyítékkal tud szolgálni a felek számára, akkor az ő 2^t darab lehetséges üzenete alapján a fentiek szerint kijelölhető a C mátrixnak legfeljebb 2^t darab olyan részmátrixa, amelyek minden eleme 1-es, és ezek a mátrix minden 1-esét tartalmazzák.

Ha most az űrhajófeladat mátrixát tekintjük, akkor nyilvánvaló, hogy ebben a $2^n \times 2^n$-es mátrixban, ahol a főátlóban 1-esek, máshol pedig 0-ák állnak, az 1-esek nem fedhetők le 2^n-nél kevesebb „csupaegy" téglalappal, ET sem képes tehát n bitnél rövidebb bizonyítékkal szolgálni egyenlőség esetén a feleknek. Az pedig nyilvánvaló, hogy ekkor ennél rövidebben saját erőből sem boldogulhatnak, hiszen ha ez lehetséges volna, akkor az ET ezt a beszélgetést közölhetné A-val és B-vel, akiknek a táblázat ismeretében a saját párbeszédükként elismerve ezt már az egyenlőség bizonyítékaként kellene elfogadniuk.

4. Egy fölső korlát

ET eddig is jó szolgálatot tett, a továbbiakban se váljunk el tőle. Láttuk, hogy az „egyenlőség-probléma" megválaszolásakor egyenlőtlenség esetén komoly segítséget nyújthat: n hosszúságú bitsorozatokra $\log_2 n$ hosszúságú bizonyítékkal szolgálhat a felek számára akkor, ha a két sorozat nem egyenlő.

Nézzük a dolgot általában. Legyen megint a két játékos A és B, és játsszanak a 0-ákból és 1-esekből álló C mátrixon. A megkapja a mátrix egy sorát, B a mátrix egy oszlopát és el kell dönteniük, mi áll a metszéspontban. Három mennyiséget vezetünk be.

Legyen $\varkappa(C)$ a minimális hosszúságú bitsorozat, amellyel a felek bármely sor és oszlop esetén eldönthetik, hogy mi áll a metszéspontban. (Az űrhajófeladatban, mint láttuk, $\varkappa(C) = n$, a triviális megoldásnál jobb nincs – még ET számára sem.)

Jelölje $\varkappa_0(C)$ az ET által adható bizonyíték minimális hosszát akkor, ha a válasz 0 és hasonlóan értelmezzük 1 válasz esetén a $\varkappa_1(C)$ mennyiséget.

Az űrhajófeladatban $\varkappa_1(C) = n$ volt. A különböző kommunikációs feladatokban ez a két szám nagyon eltérő lehet. Nyilvánvaló, hogy $\max(\varkappa_0, \varkappa_1)$ alsó korlát $\varkappa$-ra is.

Egy meglepő tétel szerint viszont $(\varkappa_0+2)(\varkappa_1+2)$ felső korlát, azaz ha ET így vagy úgy be tudja nekik bizonyítani a választ, akkor ők egymás között is elboldogulnak lényegében e két mennyiség szorzatával. Ezt a tételt nem bizonyítjuk.

5. Játék egy fán

Az alábbi példában a játék speciális szerkezetű halmazon folyik: egy n pontú gráfon, mégpedig egy fán, tehát egy olyan gráfon, amelyik összefüggő és nem tartalmaz kört. Mindkét játékos ismeri a fát. Mindketten kapnak egy-egy részfát a gráfból, A az F_A-t, B pedig az F_B-t. A másik részfáját egyikük sem ismeri. Az a feladatuk, hogy eldöntsék, van-e a két részfának közös pontja.

Nézzük először azt a változatot, amikor ET is résztvesz a játékban. Ha a két részfának van közös pontja, akkor nincs más dolga, mint ennek a pontnak a nevét közölni A-val és B-vel, akik ellenőrizhetik, hogy a pont benne van a részfájukban. A gráf pontjainak száma n, így egy pont megnevezéséhez $\log_2 n$ bit szükséges. (Valójában $[\log_2 n] + 1$, de a továbbiakban az egészrész jelét elhagyjuk.) Így $\varkappa_1 \leqq \log_2 n$.

Akkor is könnyen boldogul ET, ha a két részfa nem metszi egymást. Ekkor ugyanis van olyan él – hacsak nem üres a két részfa valamelyike, – amelyiket elhagyva az F fa két összefüggő részre – komponensre – esik szét, melyek egyike az F_A-t, a másikuk pedig az F_B-t tartalmazza. Ha ET ezt az élt közli a felekkel, mégpedig irányítással, például úgy, hogy az elhagyott él az A részfája felé mutasson, akkor ennek alapján A és B ellenőrizhetik, hogy részfáik különböző

komponensekben vannak, és így ET közleménye számukra a fa ismeretében bizonyíték. Ekkor ET-nek összesen $2n$ közleménye lehetséges: a fa $(n-1)$ darab megfelelően irányított éléhez $2(n-1)$ és még kettő arra az esetre, ha F_A vagy F_B üres. Ez megfelelő kódolással $\log_2 2n$ bitet jelent, azaz

$$\varkappa_0 \leqq \log_2 2n.$$

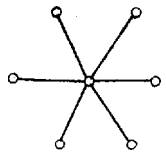
Mit tegyenek a játékosok az ET segítsége nélkül? Nos, legrosszabb esetben egyikük pl. A – elmondhatja B-nek, hogy melyik az ő részfája (az űrhajófeladatban ennek felel meg a teljes program visszaküldése). Az a helyzet, hogy egy n pontú fának általában nagyon sok – n-ben exponenciálisan sok – részfája van. Ezzel kapcsolatos az alábbi feladat.

Feladat. Bizonyítsuk be, hogy ha egy n pontú fában nincsen másodfokú pont, akkor a fának legalább $2^{n/2}$ darab részfája van.

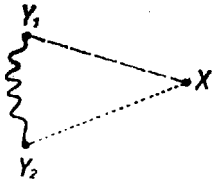
*

Ennek a ténynek az a következménye, hogy az ilyen fák esetén az a triviális eljárás, hogy A valamilyen kódolással megmondja B-nek az F_A részfáját, a legrosszabb esetben legalább $\frac{n}{2}$ bitet igényel. n bit persze minden fa esetében elegendő, hiszen ez 2^n darab közleményt tesz lehetővé, tehát például így a gráf szögpontjainak valamennyi részhalmaza felsorolható, vagyis az összes részfa is. Van olyan F fa, ahol a triviális eljárás során erre szükség is van, mert itt a csúcsok majdnem minden részhalmaza részfát jelöl ki. Ilyen az ún. csillag (3. ábra), amelynek $2^{n-1}+n$ darab részfája van.

Most azonban a triviális eljárásnál jóval hatékonyabb módszer is megadható. Válasszon az A egy X pontot az ő F_A részfájából és küldje ezt el B-nek. Ha $X \in F_B$, akkor B ezt látja és egyetlen további bitben megüzeni A-nak. Ha nem, akkor mivel az F gráf összefüggő, X-ből az F_B bármely pontjába vezet út – és csak egy, mert az F fa. Minden ilyen úton tekintsük a legelső F_B-beli pontot. Azt állítjuk, hogy ez a pont egyértelmű: Valóban, ha Y_1 és Y_2 két ilyen „belépési pont", akkor mivel F_B fa, Y_1 és Y_2 között az F_B-ben is vezet út (4. ábra) és így F-ben van kör, ami nem lehet.

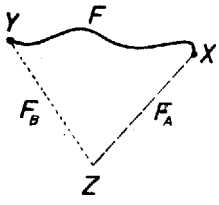


3. ábra



4. ábra

Küldje tehát vissza B ezt az X-ből az ő részfájába vezető úton egyértelműen meghatározott Y „belépési pontot" A-nak. Ha $Y \in F_A$, akkor megint készen vagyunk, A tudja a választ. De A akkor is tudja a választ, ha $Y \notin F_A$. Ekkor ugyanis a két részfának nem lehet közös pontja. Ha ugyanis volna ilyen Z közös pont (5. ábra), akkor ez az Y megválasztása miatt nem lehet rajta az X és Y között haladó F-beli úton – Y volt a belépési pont – másfelől F_A és F_B is összefüggők, így létezik F_A-beli XZ és F_B-beli ZY út, vagyis az F gráfban van kör. Ez pedig ellentmondás.



5. ábra

Az Y visszaküldése után tehát az A már tudja a választ, amit egyetlen további bittel megüzenhet a B-nek.

Az üzenetváltás során a játékosok két pont nevét közölték egymással. Ez n pont esetén az ismert módon $2\log_2 n$ bittel megoldható, így ebben a feladatban $\varkappa \leqq 2\log_2 n$.

Azt állítom másfelől, hogy most

$$\log_2 n \leqq \varkappa.$$

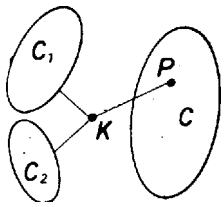
Ez nyilvánvaló, ha arra a speciális esetre gondolunk, amikor F_A és F_B egyetlen pontból állnak. Az ilyen gráfok is részfák, és a kérdés megválaszolása során az A-nak és B-nek azt kell eldöntenie, hogy a két pont egyenlő-e vagy sem. Ez pedig nem más, mint az egyenlőségfeladat egy n-elemű halmaz esetén, és erről beláttuk, hogy $\log_2 n$ bitnél olcsóbban nem oldható meg.

A $\varkappa$-ra kapott korlátok most közelíthetők. A fenti eljárás ügyesebb megszervezésével egy $\log_2 n$-hez közeli felső korláthoz juthatunk. Ehhez némi előkészítő munkálatok szükségesek. Arról van szó, hogy a fa csúcsait nem akárhogyan számozzuk meg, hanem az adott gráf szerkezetével összhangban. Hogy ez pontosan mit jelent, azt az alábbi segédtétel mondja el az $1, 2, \ldots, n$ számokkal.

Segédtétel. Minden n pontú F fa csúcsai megszámozhatók úgy, hogy ha F-ből elhagyjuk az $1, 2, \ldots, k-1$ számú csúcsokat, – azaz „felvágjuk" a fát – akkor a kapott körmentes gráf – ún. erdő – komponenseinek legfeljebb $\frac{2n}{k}$ csúcsa van.

A fenti számozás szerinti vágások tehát „egyenletesen" vágják szét a fát.

A bizonyításhoz megmutatjuk, hogy minden m pontú fában van olyan pont – hívjuk ezt a fa „középpontjának" – amelyet elhagyva a fából az legfeljebb $\frac{m}{2}$ pontból álló komponensekre esik szét.



6. ábra

Tekintsük ugyanis a fában azt a M pontot, amelyet elhagyva a kapott komponensek elemszámának maximuma minimális (6. ábra). Ha C egy ilyen maximális elemű komponens, akkor azt állítjuk, hogy C-nek legfeljebb $\frac{m}{2}$ pontja van. Tegyük fel, hogy ez nem igaz és a C-ben több, mint $\frac{m}{2}$ pont van. Ekkor a további $C_1, C_2, \ldots$ komponenseknek együttvéve még az M ponttal együtt is $\frac{m}{2}$-nél kevesebb pontja van. Vegyük most vissza a gráfba az elhagyott M pontot, és hagyjuk el az M-ből a C-be vezető él másik, P végpontját. Ekkor a $C_1, C_2, \ldots$ komponensek az M-mel együtt egyetlen, $\frac{m}{2}$-nél kevesebb pontú C_0 komponenssé állnak össze. A C komponensből a P elhagyása után egy, a C-nél eggyel kevesebb pontú gráf lesz, és így akár összefüggő marad, akár nem, az M helyett a P-t elhagyva csökken a komponensek elemszámának maximuma. Ez ellentmond az M kiválasztásának, tehát a C-ben valóban nem lehet $\frac{m}{2}$-nél több pont.

Ezután megadjuk a Segédtételben leírt számozást. Kapja az F egy középpontja az 1-es sorszámot. A 2-es sorszámú pont az 1-es elhagyása után keletkező legnagyobb komponens középpontja legyen, és így tovább, ha már megvannak az $1, 2, \ldots, k-1$ sorszámú csúcsok, akkor a k sorszámot a már megszámozott pontok elhagyásával keletkező legnagyobb komponens középpontja kapja.

Feladat. Bizonyítsuk be, hogy ez a számozás megfelelő.

*

Nézzük most meg, hogyan segíthet A-nak és B-nek ez a számozás. Az eljárás kezdetén az F gráffal együtt rögzítsenek egy ilyen számozást, majd külön-külön válasszák ki az F_A és az F_B részfájukat.

Az első lépésben A kiválasztja a részfájába eső legkisebb – mondjuk k – sorszámú csúcsot és ezt a k értéket elküldi B-nek. (Ha az F_A üres, akkor küldjön 0-át: ekkor készen vannak.)

A k ismeretében a B tudja, hogy F_A nem tartalmazza a k-nál kisebb sorszámú pontokat. Ezeket tehát elhagyhatja az F fából, F_A benne lesz valamelyik komponensben. Mivel B az F_A egyik csúcsát is ismeri – ez a k sorszámú pont – így azt is tudja, hogy melyik ez a bizonyos F_0 komponens. F_0-t természetesen az A is ismeri.

Ha most az F_B részfa nem metszi az F_0 komponenst, akkor F_A-nak és F_B-nek nincsen közös csúcsa, és ezt B egyetlen további 0 bittel jelezheti A-nak.

Ha az F_B metszi F_0-t, akkor legyen $F_B' = F_B \cap F_0$, az F_B-nek az F_0-ba eső része. F_B' is fa, és így az eredeti eljárás első lépése utáni helyzet állt elő, csak a játék fájának a mérete csökkent és a B fája módosult: F_A és F_B' most a segédtétel szerint legfeljebb $\frac{2n}{k}$ csúcsú F_0 fa részfái, és ezt A és B mindketten tudják. Ha B visszaküldi az előző eljárásban megadott Y belépési pontot A-nak, akkor az ott leírtak szerint ebből A már tudni fogja a választ.

A fenti számozás most már lehetővé teszi, hogy B ne az Y eredeti, F-beli sorszámát küldje vissza, hisz az ehhez szükséges $\log_2 n$ bit nem hasznosítja azt a mindkét fél számára rendelkezésre álló információt, hogy F_A és F'_B is a „kicsi" F_0-ban vannak. Ha az F_0-beli pontok sorszámai $k_1 < k_2 < \ldots < k_m$, és az Y pont ezek közül nagyság szerint az i-edik, akkor A-nak csak az i értékére van szüksége ahhoz, hogy az eredeti számozás alapján megkeresse az F_0 komponensben az Y pontot. X-et és Y-t ismerve ezután már az előző eljáráshoz hasonlóan dönteni is tud.

Vizsgáljuk meg az üzenetváltás hosszát. Az A a k értékét $\log_2 k$ bitben küldi el B-nek, aki $1 \leqq i \leqq m \leqq \dfrac{2n}{k}$ miatt $\log_2 \dfrac{2n}{k} = 1 + \log_2 n - \log_2 k$ bitben küldheti vissza az i értékét A-nak. Ez pedig összesen mindössze $1 + \log_2 n$ bit.

Egy elvi megjegyzést szeretnék még fűzni a fentiekhez. Az elhangzott példákban a felek a párbeszédek során mindig jóelőre megállapodtak az egyes üzenetek hosszában, ami elég természetes követelmény. Ez ugyan olykor látszólag gazdaságtalan megoldást jelent, hiszen ha például 100 jegyű számokat kell üzengetni, akkor a kicsi 1-est ugyanúgy 100 jegyű sorozat kódolja, mint a nagy számokat. A legutolsó példában viszont, ahol A első üzenete a k értéke, a játékosok nem tudnak előre megállapodni az üzenet hosszában, hisz a k értékét nem ismerik. Ha $n = 127$, $k = 3$, akkor A nem teheti meg, hogy csak az 11 bitsorozatot küldi el B-nek és vár. B ugyanis ekkor nem tudja, valóban vége van-e A üzenetének. Egy hosszú csend itt nem megoldás, hiszen az tulajdonképpen egy extra jel, A és B pedig csak a 0-t és az 1-est ismeri el jelként. Nem megoldás az sem, hogy valamilyen „üzenet vége" speciális bitsorozatban állapodnak meg, hisz előre nem látható hosszúságú üzenetnél nem lehet tudni, hogy az érkező jelek nem a valódi üzenethez tartoznak-e.

A persze elküldhetné az n értéke alapján félreérthetetlen 000011 bitsorozatot, hisz n nagyságrendjét mindketten ismerik, de ekkor $\log_2 k$ helyett $\log_2 n$ lesz az első üzenetének hossza. Ha viszont A egy – a mindkét fél által ismert és előre rögzített $\log_2 \log_2 n$ hosszúságú – sorozattal kezdi az üzenetét, amelyben közli a k méretét, akkor B az első $\log_2 \log_2 n$ jegyből álló számból már tudja, hogy a közlemény második, a k-t tartalmazó része milyen hosszú. A példában tehát az A üzenete a $k = 3$-ra 1011. A második lépésben B már nyugodtan használhat $\log \dfrac{2n}{k}$ méretű bitsorozatokat minden kommentár nélkül az i kódolására, hiszen a párbeszédnek ebben a fázisában már mindketten ismerik a k értékét. Ez annyit jelent, hogy a párbeszéd hossza $\log_2 \log_2 n$ bittel nő, ami azonban a $\log_2 n$ taghoz képest kicsi.

Végül azt kaptuk tehát, hogy a feladatra

$$\varkappa \leqq +1 \log_2 n + \log_2 \log_2 n,$$

ami valóban alig nagyobb, mint az alsó korlátként kapott $\log_2 n$.

*

A kommunikáció bonyolultságának megismerésében még csak a kezdet kezdetén járunk. Nagyon keveset tudunk például arról, hogy mi történik 2-nél több résztvevő esetén. Azt sem értjük igazán, hogy miért segít a véletlen használata az űrhajós példában, és miért nem segít más, hasonló feladatok esetén. A kommunikáció, hírközlés korunk egyik fő jelensége, és bizonyos, hogy még sok izgalmas matematikai problémával fog szolgálni.