# MATEMATICI DISCRETE APLICATE

**Compilație realizată de**
**Gheorghe M.Panaitescu**

**Catedra Automatică si calculatoare**
**Universitatea "Petrol-Gaze" Ploiesti**
**2007**

# I N T R O D U C E R E

Prezenta compilatie este în principal o traducere cu unele adaptări minore a cursului de Matematici discrete tinut în primăvara anului 2005 la University of California, Berkeley de **Michael J.Clancy** si **David Wagner**.

Am adăugat un capitol bogat despre grafuri preluat din cursul predat de **Marc Pomplun** la University of Massachusetts at Boston.

În multe puncte, pentru întelegerea unor detalii si transpunerea lor corectă si inteligibilă în limba română am apelat la cursul de Matematici discrete predat de **David A.Santos** la Community College of Philadelphia.

Toate aceste surse primare sunt accesibile publicului pe site-urile universitătilor mentionate.

Ar fi nedrept a fi ignorată pregătirea matematică a autorului acestei compilatii: materialul rezultat este accesibil inginerilor poate si datorită pregătirii duble, de inginer si matematician a autorului.

Aceste lectii sunt acompaniate de un volum separat de Probleme si aplicatii, de regulă gata rezolvate, preluate în general din aceleasi surse.

**C U P R I N S**

# Lecția 1

## Scopul cursului

Cursul prezent cuprinde mai putine subiecte decât ar putea cuprinde, dar toate cele retinute pentru predare se asociază la necesităti reale ale calculului ingineresc. Există speranta că acest curs va fi interesant pentru studentii de la masterat în Automatizare si informatică si va aduce o întelegere mai adâncă si mai durabilă a matematicii implicate în automatizare si în utilizarea calculatoarelor.

Obiectivele predării acestui curs sunt constau în a dezvolta:

- *O gândire robustă si precisă*: aceste cunostinte permit studentilor utilizarea si dezvoltarea unor idei mai subtile si mai complexe în domeniul lor, mult dincolo de tratarea primară a problemelor si fac posibilă evitarea erorilor adesea elementare comise si descoperite la unele din testele finale din aria automatizare-calculatoare.
- *Abilitatea de a enunta si a dovedi fapte neuzuale, în particular relativ la programele de calcul*: capacitatea de a scrie programe corecte, care la rândul lor să se constituie în blocuri de calcul solid construite pentru a intra în componenta unor sisteme din ce în ce mai complexe dar încă fiabile.
- *Perceptia adecvată a fundamentelor matematice utilizate în ingineria automatizării si calculului*: familiarizarea cu logica, cu structurile definite inductiv, cu aritmetica întregilor si cu algebra polinoamelor, cu calculul probabilistic – concepte care se regăsesc în toate cursurile avansate din domeniul automatizării si calculului.

Pe scurt, continutul cursului face referiri la:

- Propozitii si demonstratii
- Inductia matematică: recursivitatea
- Logica propozitiilor: demonstrarea si rezolvarea automată de probleme
- Algoritmi aritmetici: cel mai mare divizor comun, cel mai mic multiplu comun, verificarea calitătii unui număr de a fi prim, sistemele de criptare RSA
- Polinoamele si aplicatiile lor: coduri corectoare de erori, secrete partajate
- Probabilităti si algoritmi probabilistici: echilibrarea încărcării mai multor procesoare, *hashing*, constructii probabilistice, probabilităti conditionate, inferenta bayesiană
- Grafuri, aplicatiile lor, imposibilitatea ocazională de a face un calcul

**Propozitii si demonstratii**

Multi dintre cei instruiti partial în problemele rationamentelor matematice cred că demonstratiile sunt exercitii formale fără un scop precis, similare mai curând ghicirii unui traseu printr-un labirint pentru a găsi o comoară veche de 2400 de ani sau mai mult.
Această gândire este într-un fel departe de realitate. Oricui îi place să spună lucruri de genul:

> "Acest sistem de criptare nu poate fi spart"
> "Programele mele lucrează eficient în toate împrejurările"
> "Nu există situatii în care eu să mint autoritătile"
> "Este de neconceput ca sistemul nostru legal să ducă la pedepsirea unei persoane inocente"

si altele asemenea. Relativ putini dintre oameni optează a sustine lucruri care se dovedesc a fi false. Demonstratia înseamnă a crea premisa de a nu spune vreodată "regret, scuze" – ea este mijlocul de a garanta o afirmatie o dată pentru totdeauna si pentru toti.
Dar ce se poate spune despre demonstratii false, incorecte? O demonstratie incorectă nu este o demonstratie, la fel cum iarba artificială nu este iarbă. În continuare aceste concepte vor fi redefinite în contururi mai precise. Multe cursuri de matematici discrete trec imediat la demonstratii; nu-i o idee rea în întregime, dar se face un salt poate nepermis peste unele concepte importante.
Demonstratiile în matematici si în tehnicile de automatizare si calcul (spre deosebire de justitie si politică, poate) cer a dovedi o propozitie precis enuntată.

O **propozitie** este o exprimare care este fie adevărată, fie falsă. Exprimările care nu sunt propozitii includ adesea întrebări si comenzi – acestea nu pot fi adevărate sau false, desi pot fi inteligibile sau absurde.

Se vorbeste adesea de teoreme. Simplu, o **teoremă** este o propozitie al cărui adevăr este garantat de o demonstratie.

Câteva exemple de propozitii:
(1a) Unele mamifere ouă
(1b) Unele mamifere au fulgi
(2) Acceleratia unui corp rigid este proportională cu forta aplicată asupra acelui corp
(3) Unghiurile unui triunghi au suma egală cu 180 de grade
(4) Pentru orice întreg nenegativ $n$, $n^2 + n + 41$ este un număr prim
(5) Pentru orice întreg $n$, dacă $n > 2$ nu există întregi pozitivi $a$, $b$, $c$ astfel încât $a^n + b^n = c^n$.
(6) Pentru orice întreg par $n > 2$, există două numere prime $a$ si $b$ astfel încât suma $a + b = n$.

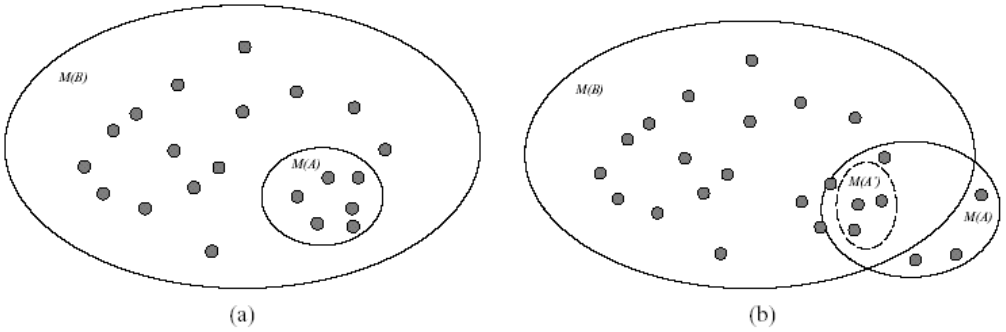
Este foarte important a observa că fiecare propozitie este adevărată sau falsă *în raport cu o lume posibilă*. Reciproc, o lume (sau un *model* în terminologia logicii) este ceva în raport cu care o propozitie de interes este fie adevărată, fie falsă – adică este complet specificată (definită).
Pentru fizică, chimie, biologie etc., care sunt stiinte *empirice*, suntem uzual interesati de adevăr în raport cu lumea *reală*, lumea în care de fapt trăim. Dar, desigur, nu stim care este aceea. De pildă, propozitia (1a) se întâmplă să fie adevărată în lumea reală, dar ar putea fi cu usurintă si altfel; propozitia (1b) poate fi si ea adevărată sau nu.
Propozitia (2) este una din legile lui Newton. S-a presupus ani si ani de zile a fi incontestabil adevărată si multe explicatii au fost date de ce n-ar fi posibil să fie altfel; dar ea este de fapt falsă în lumea reală. Unii declară că legile lui Newton au fost desfiintate de Einstein. Adevărul este altul: Einstein a propus mai curând legi care sunt inconsistente cu legile lui Newton. Legile empirice de genul legilor lui Newton pot fi contrazise numai prin observatii sau prin evidentierea unei inconsistente interioare.
De secole matematicienii, fizicienii si inginerii au demonstrat si au utilizat teoreme din mecanica newtoniană si continuă să o facă. Din punct de vedere fizic, multe din aceste teoreme sunt false în lumea reală. Ele sunt *totusi teoreme* ale mecanicii newtoniene. O demonstratie incorectă nu este o demonstratie, dar o teoremă falsă poate fi totusi o teoremă *dat fiind că decurge logic dintr-un set de* **axiome** *precizat*. O axiomă este o propozitie care este presupusă a fi adevărată fără vreo demonstratie. În mecanica newtoniană legile lui Newton sunt luate drept axiome.
Asadar, demonstratia este un mijloc de a arăta că o teoremă decurge logic dintr-un set de axiome. Trebuie explicat acum ce înseamnă "a decurge logic".
O propozitie $B$ decurge logic dintr-o altă propozitie $A$ dacă $B$ este adevărată în orice lume în care $A$ este adevărată.
Această relatie este scrisă adesea sub forma $A \models B$ ceea ce se poate citi ca "$A$ implică logic pe $B$". Într-o reprezentare grafică, multimea de lumi în care $A$ este adevărată este continută în multimea de lumi în care $B$ este adevărată ori de câte ori $A \models B$. În limbaj matematic mai riguros, dacă $M(A)$ este multimea de lumi în care $A$ este adevărată si $M(B)$ este multimea de lumi în care $B$ este adevărată (numim aceste lumi **modele** ale lui $A$ si $B$), atunci

$$A \models B \text{ dacă si numai dacă } M(A) \subseteq M(B)$$

Această relatie este ilustrată în partea din figura următoare marcată cu (a). Sensul semnului $\subseteq$ poate fi întrucâtva contraintuitiv: normal, gândim că $A$ este "mai tare" sau "mai mare" decât $B$ dacă "$A$ implică logic pe $B$". O cale de a tempera intuitia constă în a observa că fiecare axiomă adăugată lui $A$ *reduce* multimea de lumi posibile în care $A$ este adevărată, ceea ce face *mai* posibilă continerea acestei multimi în multimea de lumi pentru $B$, conform cu a doua parte a figurii care urmează, marcată cu (b).
S-a spus mai devreme că o demonstratie garantează o propozitie. Mai precis, făcând uz de definitia implicatiei logice, se vede că o demonstratie garantează adevărul unei teoreme în măsura în care axiomele însesi sunt adevărate. Vom

vedea în sectiunea următoare unele metode prin care se asigură această garantare.

Să revenim la propozitiile din lista de mai sus. Propozitia (3), "Suma unghiurilor unui triunghi este egală cu 180 de grade" este una din axiomele lui Euclid. Simplu, se postulează a fi adevărată. În realitate, ea este adevărată numai în geometria plană. Un triunghi asezat pe suprafata unei sfere poate face axioma evident falsă; relativitatea generală spune că spatiul însusi este curbat, astfel că geometriile care violează axioma în discutie sunt în realitate mult mai realiste. Ca si în cazul mecanicii newtoniene, teoremele dezvoltate de Euclid (si după el utilizate de multe generatii de elevi) sunt adevărate numai într-un univers idealizat, "plat".




Explicatii la figură: (a) *A* implică *B* dacă si numai dacă *B* este adevărată în orice lume în care *A* este adevărată. (b) Adăugând axiome la *A* pentru a deveni *A'* se *reduce* multimea de lumi posibile; figura arată un caz în care aceasta permite lui *A'* să implice pe *B*.

Când ne ocupăm de propozitii pur matematice, situatia este putin diferită: axiomele matematice sunt *definitorii* pentru lumea în discutie si nu o încercare de a o *descrie*. De pildă, axiomele lui Peano definesc ce este acela un număr natural (un întreg nenegativ):

0 este un număr natural

Dacă *n* este un număr natural, *s*(*n*) este un număr natural

Aici *s*(.) este *functia succesor*. Scriem uzual *s*(0) ca 1, *s*(*s*(0)) ca 2 s.a.m.d. Din acest început simplu se pot definii multe alte notiuni – adunarea, multiplicarea, scăderea, divizarea, numerele prime s.a.m.d. Teoremele din matematică sunt adevărate deoarece axiomele din matematică sunt (uzual) adevărate prin definitie (am adăugat "uzual" deoarece uneori un set de axiome propus pentru un domeniu al matematicii se pot dovedi inconsistente – adică contradictorii între ele – si de aceea *nu pot fi* adevărate). Textele matematice vorbesc uzual de demonstrarea adevărului sau falsitătii unei propozitii. Exprimarea este adesea o prescurtare din care lipseste "urmare a axiomelor standard" sau "inconsistentă cu axiomele standard". Vom face acelasi lucru dar cu încercarea de a avea grijă a mentiona axiomele necesare. De ce? Mai întâi aceasta este o practică bună deoarece elimină unele erori care apar accidental când se invocă o axiomă falsă;

în al doilea rând se relevă posibilitatea de a demonstra teoreme mai generale deoarece unele axiome pot să nu fie necesare în demomstratie; apoi, în al treilea rând, axiomele pe care se bazează demonstratia pot deveni mai târziu inconsistente si e bine a fi tinută o evidentă a premiselor utilizate (această posibilitate este destul de improbabilă în cele mai multe demonstratii care urmează).

**Conectori logici**

Din propozitii existente se pot forma alte propozitii noi, în mai multe moduri. Iată câteva din ele. Fie *P*, *Q* două propozitii arbitrare.

**Negarea**: $\neg P$ este propozitia "*non P*" si este adevărată dacă *P* este falsă.

**Disjunctia**: $P \vee Q$ este propozitia "*P* sau *Q*" si este adevărată dacă cel putin una din propozitiile *P*, *Q* este adevărată.

**Conjunctia**: $P \wedge Q$ este propozitia "*P* si *Q*" si este adevărată dacă ambele propozitii *P*, *Q* sunt adevărate.

**Implicatia**: $P \Rightarrow Q$ este propozitia "*P* implică *Q*" si este adevărată dacă propozitia *P* este falsă sau dacă propozitia *Q* este adevărată.

A se vedea tabelul de mai jos pentru definitiile acestor operatori.

| $P$ | $Q$ | $\neg P$ | $P \wedge Q$ | $P \vee Q$ | $P \Rightarrow Q$ | $P \Leftrightarrow Q$ |
|---|---|---|---|---|---|---|
| F | F | T | F | F | T | T |
| F | T | T | F | T | T | F |
| T | F | F | F | T | F | F |
| T | T | F | T | T | T | T |

Tabel de adevăr cu definitiile operatorilor logici standard

De retinut că acesti operatori pot fi combinati de mai multe ori pentru a construi propozitii compuse. De pildă, $P \vee \neg (P \Rightarrow Q)$ este o propozitie validă dacă *P* si *Q* sunt valide si este adevărată dacă si numai dacă *P* este adevărată.

**Predicate**

Se lucrează uneori cu colectii de propozitii. Fie de pildă lista de propozitii:

$A$ = "$0^2 + 0 + 41$ este număr prim"
$B$ = "$1^2 + 1 + 41$ este număr prim"
$C$ = "$2^2 + 2 + 41$ este număr prim"
$D$ = "$3^2 + 3 + 41$ este număr prim"

. . . . . .

Listarea în continuare a acestei familii de propozitii devine repede o povară, nemaivorbind de isprăvirea la un moment dat a literelor alfabetului. Asadar, este util a avea notiunea de "o functie care fiind dat un număr natural *n* produce o propozitie care spune ceva despre *n*". Un nume standard pentru aceasta este cel de *predicat*.

Un **predicat** *P* este o functie care aplică fiecare valoare *n* pe o propozitie $P(n)$ care depinde de *n* într-un mod anume.
De pildă, în exemplul de mai sus se poate defini un predicat *P* ca

$$P(n) = \text{“}n^2 + n + 41 \text{ este număr prim”}$$

$P(17)$ actionează ca o sciere prescurtată pentru: “$17^2 + 17 + 41$ este un număr prim”. Este o modalitate eficace de a grupa o colectie numeroasă, chiar infinită de propozitii.
De observat că dacă *P* este un predicat, dacă el este adevărat sau fals depinde de valoarea lui *n*. $P(17)$ ar putea fi adevărată dar $P(18)$ ar putea fi falsă. Pentru orice valoare particulară *n*, $P(n)$ este o propozitie care este fie adevărată fie falsă.
Desigur, putem aplica si predicatelor operatorii logici standard. De pildă, dacă $P(n)$ si $Q(n)$ sunt predicate, atunci $P(n) \vee Q(n)$ notează un predicat care este adevărat pentru acele valori ale lui *n* pentru care fie $P(n)$ este adevărat, fie $Q(n)$ este adevărat.

**Cuantificatori**

Introducem acum încă vreo câteva notatii. Să rescriem propozitia (4) sub forma

(4) $\forall\ n\ .\ n^2 + n + 41$ este număr prim

Simbolul $\forall$ este **cuantificatorul universal**; aici, el se referă la variabila *n* si înseamnă “pentru orice *n*…”. Strict vorbind, propozitia ar trebui scrisă

(4) $\forall\ n \in \mathbf{N}\ .\ n^2 + n + 41$ este număr prim

cu **N** multimea numerelor naturale. Această clarificare suplimentară este adesea omisă când contextul o face cât de cât evidentă.
În general, dacă *P* este un predicat, atunci $\forall\ n \in \mathbf{N}\ .\ P(n)$ este o propozitie care este fie adevărată, fie falsă.
Cum se poate demonstra o afirmatie cuantificată universal? Se poate constata că este adevărată pentru multe cazuri: $n = 0$, $n = 1$ s.a.m.d. până la $n = 39$. Constiuie asta o demonstartie? Desigur, nu! Propozitia e falsă deoarece $40^2 + 40 + 41$ nu este număr prim. Altfel spus, $P(40)$ este falsă. Cazul $n = 40$ este numit un **contraexemplu** pentru această propozitie.
Propozitia (5) de mai sus este ultima dintre teoremele lui Fermat. Este cunoscută de 300 de ani si tot de 300 de ani este numită teoremă deoarece Fermat a revendicat o demonstratie si nu a fost găsit nici un contraexemplu. Dar numai recent ea a devenit o teoremă propriu-zisă când Andrew Wiles a dezvoltat o demonstratie reală (lungă de câteva sute de pagini).

Propozitia (6) este **conjectura** lui Goldbach. O conjectură este o propozitie care nu a fost nici demonstrată, nici contrazisă. Utilizând notatia cu cuantificatori, ea se scrie

$\forall\ n$ . dacă $n$ este par atunci $\exists\ a, b$ astfel încât $a$ si $b$ sunt prime si $a + b = n$

Aici, $\exists$ este **cuantificatorul existential**, care înseamnă "pentru un…" sau "există…". Pentru orice $n$ particular, afirmatia cuantificată existential poate fi demonstrată simplu prin găsirea unor numere $a$ si $b$ particulare, dar, desigur, cuantificarea universală pentru $n$ nu permite demonstrarea prin generarea de exemple. Ea poate fi însă infirmată prin producerea unui singur contraexemplu, dar nici unul nu a fost găsit în ciuda verificărilor până la valori enorme ale lui $n$; conjectura e suspectă de a fi adevărată.

Cineva ar putea spune: "Sigur, ceva atât de simplu s-ar cădea a fi usor de demonstrat!" Dar ultima teoremă a lui Fermat s-a dovedit (deocamdată) a avea o demonstratie foarte complicată; teorema faimoasă a lui Kurt Gödel despre incompletitudine a arătat că există propozitii adevărate care *nu* au demonstratii în aritmetică. Din nefericire nu există vreo cale de a spune că acea conjectură a lui Goldbach este una de acest gen (teorema incompletitudinii a lui Gödel nu este în general o scuză valabilă pentru inabilitatea de a face o demonstratie relativ la o problemă din temele de casă…).

În genral, dacă $P$ este un predicat, atunci propozitia $\exists\ n \in \mathbf{N}.P(n)$ poate fi demonstrată printr-un unic exemplu de $n$ care face ca $P(n)$ să fie adevărată, dar infirmarea ei reclamă a arăta că $P(n)$ este falsă pentru toate numerele $n$. Comparativ, demonstrarea că $\forall\ n \in \mathbf{N}.P(n)$ este adevărată cere a arăta că $P(n)$ este adevărată pentru orice $n$, pe când infirmarea ei poate fi făcută prin găsirea unui singur contraexemplu, adică a unui $n$ pentru care $P(n)$ este falsă.

O propozitie cuantificată universal este foarte asemănătoare unei conjunctii mai lungi. De pildă, se defineste o multime $\mathbf{S} = \{1, 2, 3, 4\}$. Atunci, propozitia $\forall\ x \in \mathbf{S}.P(x)$ este echivalentă cu propozitia $P(1) \wedge P(2) \wedge P(3) \wedge P(4)$. Analog, propozitiile cuantificate existential sunt mai asemănătoare unei disjunctii mai lungi si $\exists\ x \mathbf{S} \in .P(x)$ echivalează cu $P(1) \vee P(2) \vee P(3) \vee P(4)$.

S-a vorbit deja prea mult despre propozitii si demonstratii. Să trecem la modul cum se pot verifica, demonstra propozitiile.

**Demonstratia prin enumerare**

Începem cu o metodă foarte simplă de a demonstra propozitii bazată pe definitia urmării, consecintei logice (*logical entailment*). Se consideră următorul exemplu extrem de simplu:

> Se dă: trandafirii sunt rosii şi violetele sunt albastre
> Demonstratie: trandafirii sunt rosii

Acest fapt este "evident corect" din cauza semnificatiei lui "şi". O propozitie "$P$ şi $Q$", care în notatie matematică se scrie $P \wedge Q$, este adevărată numai dacă $P$ este adevărată si $Q$ este adevărată. Astfel, semnul "$\wedge$" poate fi privit ca un

operator care construieste din două propozitii simple o propozitie compusă numită **conjunctie**. Acest operator este definit prin tabelul de adevăr al propozitiei compuse pentru toate valorile de adevăr posibile ale propozitiilor constitutive, cum se poate vedea într-un tabel mai sus.

Pentru a dovedi formal că trandafirii sunt rosii ($P$), stiind că trandafirii sunt rosii *şi* violetele sunt albastre ($P \wedge Q$), se identifică mai întâi toate cazurile posibile în care $P \wedge Q$ este adevărată. Un singur caz se identifică si acesta este cel din linia a patra a tabelului. Si, desigur, în acel caz "trandafirii sunt rosii", ($P$) este adevărată. Astfel demonstratia se încheie.

Demonstratia pare de la ficilă la prostească, dar ilustrează foarte bine conceptul fundamental al demonstrării formale prin enumerarea cazurilor posibile. Mai departe se va vedea că aceeasi idee de bază poate fi instanțiată într-un program care execută sarcini deductive mult dincolo de puterile fiintelor umane.

Demonstrația prin enumerare este plictisitoare dincolo de orice închipuire, de aceea vom mai examina numai încă un caz:

Se dă: dacă A nu este ales, atunci eu îmi mănânc pălăria

A nu este ales

Demonstratie: eu îmi mănânc pălăria

Aici prima propozitie are forma unei **implicatii**. În notatia matematică asta se scrie $P \Rightarrow Q$. Tabelul de adevăr pentru "$\Rightarrow$" este cuprins în acelasi tabel de mai sus. De observat că implicatia $P \Rightarrow Q$ este falsă numai când **antecedenta** $P$ este adevărată si **consecinta** ei $Q$ este falsă. Implicatia este *adevărată* în acele cazuri în care antecedenta este *falsă*. Asta-i bine deoarece în aceste cazuri implicatia, simplu, nu se aplică. Pentru facilitarea lecturii, se reia aici tabelul de adevăr de mai sus.

| $P$ | $Q$ | $\neg P$ | $P \wedge Q$ | $P \vee Q$ | $P \Rightarrow Q$ | $P \Leftrightarrow Q$ |
|---|---|---|---|---|---|---|
| F | F | T | F | F | T | T |
| F | T | T | F | T | T | F |
| T | F | F | F | T | F | F |
| T | T | F | T | T | T | T |

**Demonstratie**: Mai întâi identificăm toate acele cazuri în care atât $P \Rightarrow Q$ cât si $P$ sunt adevărate. Există numai un astfel de caz, si anume linia a patra a tabelului. Si, desigur, în acest caz "îmi voi mânca pălăria" ($Q$) este de asemenea adevărată.

□

De notat că semnul □ marchează finalul demonstratiei (în zile mai bune, oamenii scriau în loc QED, ceea ce însemna în latineste *quod erat demonstrandum* – ceea ce era de demonstrat). Unor oameni le place să citească textele urmărind ceea ce este scris între "Demonstratie:" si semnul □ si privind textul obisnuit ca pe o inutilă umplutură. Altii fac exact opusul, citesc privind textul demonstratiei ca pe un detaliu plictisitor. Puteti ghici cărei categorii apartine o persoană sau alta.

**Demonstratii prin aplicarea de reguli de inferență**

De notat că odată făcută o demonstratie prin enumerare, se poate extrage un *pattern* general denumit o **regulă de inferență**. Un *pattern* în secțiunea precedentă era:

> Pentru orice propozitii $P$ si $Q$, propozitia $P$ poate fi inferată din propozitia $P \wedge Q$.

Aceasta se poate scrie si utilizând următoarea notatie:

$$\frac{P \wedge Q}{P} \text{ (eliminare de şi)}$$

Aceasta este o **regulă de inferență**. Propozitiile de deasupra liniei sunt ipotezele. Dacă toate aceste ipoteze sunt cunoscute a fi adevărate, atunci regula de inferență spune că se poate conchide în sigurantă că propozitiile de sub linie sunt de asemenea adevărate. Regulile de inferență sunt crise adesea cu o versiune scurtă a numelui lor alături de ele; cea de mai sus este numită **eliminare-de-şi**.
Un alt *pattern* din sectiunea precedentă era:

> Pentru orice propozitii $P$ si $Q$, propozitia $Q$ poate fi inferată din propozitia $P$ si din $P \Rightarrow Q$.

Aceasta poate fi scrisă ca următoarea regulă de inferentă:

$$\frac{P, P \Rightarrow Q}{Q} \text{ (modus ponens)}$$

si se numeste modus ponens (în latină "afirmarea antecedentului"). Este unul din pasii cei mai comuni utilizati în demonstratii. De observat că această regulă particulară are două ipoteze (una este $P$, celalaltă este $P \Rightarrow Q$); atentie, acestea trebuie să fie satisfăcute ambele înainte de a aplica regula de inferentă pentru a conchide $Q$. Mai sunt si alte reguli de acest gen. Acestea sunt listate mai jos. Oricine se poate convinge de corectitudinea acestor reguli de inferentă prin demonstratie enumerativă.

$$\frac{P \wedge Q}{P} \text{ (eliminare-de-si)} \qquad \frac{P}{P \vee Q} \text{ (introducere-de-sau)}$$

$$\frac{P \wedge Q}{Q} \text{ (eliminare-de-si)} \qquad \frac{Q}{P \vee Q} \text{ (introducere-de-sau)}$$

$$\frac{P, Q}{P \wedge Q} \text{ (introducere-de-si)} \qquad \frac{}{P \vee \neg P} \text{ (mediul exclus)}$$

$\frac{P \vee Q, \neg P}{Q}$ (eliminare-de-sau) $\frac{P \vee Q, \neg Q}{P}$ (eliminare-de-sau)

$\frac{P \Rightarrow Q, P}{Q}$ (modus ponens) $\frac{P, \neg P}{Q}$ (contrapunere)

$\frac{P \Rightarrow Q, \neg Q}{\neg P}$ (modus tollens) $\frac{P \Rightarrow Q, R \Rightarrow S}{P \vee R \Rightarrow Q \vee S}$

$\frac{P \Rightarrow Q, Q \Rightarrow R}{P \Rightarrow R}$ (trazitivitate)

Regulile pot fi înlăntuite. De pildă, dacă se cunoaste că $A \wedge B$ si $B \Rightarrow C$, se poate aplica eliminarea-de-şi pentru a obtine $B$ si apoi modus ponens pentru a obtine $C$. O demonstratie cu un tabel de adevăr complet ar necesita opt linii pentru a trata cele trei propozitii.

Există de asemenea un număr de echivalențe care sunt utile în manipularea expresiilor logice. De pildă, propozitia $P$ este echivalentă cu $\neg(\neg P)$, prin care se consemnează faptul că $P$ este adevărată dacă si numai dacă $\neg(\neg P)$ este adevărată. Această echivalență se scrie concis $P \equiv \neg(\neg P)$. Mai jos este dată o listă întreagă de echivalențe utile. Se poate înlocui liber o expresie logică prin orice echivalentă a ei. O demonstrație prin enumerare este deplin convingătoare pentru corectitudinea lor. Fiecare echivalență duce la o regulă de inferență: de pildă, echivalența $P \equiv \neg(\neg P)$ produce regulile de inferență $\frac{P}{\neg(\neg P)}$ si $\frac{\neg(\neg P)}{P}$.

**Demonstrații prin contrapunere**

Se consideră propozitiile

> Dacă Ion este la muncă, el este logat
> Dacă Ion nu este logat, el nu este la muncă

Este limpede, fiecare din aceste propozitii poate fi demonstrată de cealaltă. Ele sunt **echivalente logic**, adică valorile lor de adevăr sunt aceleasi în toate lumile posibile. Scriind pe prima ca $P \Rightarrow Q$ si pe cealaltă ca $\neg Q \Rightarrow \neg P$, este usor de verificat echivalența utilizând tabelele de adevăr. (Multe asemenea echivalențe vor fi expuse mai departe.).

Propozitia $\neg Q \Rightarrow \neg P$ este **contrapusa** propozitiei $P \Rightarrow Q$. (A nu se confunda cu reciproca: care este $Q \Rightarrow P$ si *nu este* echivalentă). Demonstrația prin contrapunere (denumită si demonstrație indirectă) înseamnă demonstrarea propozitiei $P \Rightarrow Q$ prin demonstarea propozitiei $\neg Q \Rightarrow \neg P$. Deoarece cele două sunt echivalente, demonstrând una din ele se stabileste automat cealaltă. Iată un exemplu simplu:

**Teorema 1.1**: Pentru orice întreg $n$, dacă $n^2$ este par atunci $n$ este par.

**Demonstratie**: Se va demonstra contrapusa: dacă $n$ este impar atunci $n^2$ este impar.

1. Dacă $n$ este impar, atunci prin definitie $n = 2k + 1$ cu $k$ un întreg
2. Pentru orice numere $x$, $y$, se stie că $x = y \Rightarrow x^2 = y^2$. Asadar,
$$n^2 = (2a + 1)^2 = 4a^2 + 4a + 1 = 2(2a^2 + 2a) + 1$$
3. Deoarece $a$ este întreg, $(2a^2 + 2a)$ este întreg
4. Asadar, prin definitia imparității, $n^2$ este un număr impar

□

Echivalențe logice utile

$$P \equiv \neg(\neg P)$$
$$P \wedge P \equiv P$$
$$P \vee P \equiv P$$
$$P \wedge Q \equiv Q \wedge P$$
$$P \vee Q \equiv Q \vee P$$
$$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$$
$$(P \vee Q) \vee R \equiv P \vee (Q \vee R)$$
$$(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$$
$$(P \vee Q) \wedge R \equiv (P \wedge R) \vee (Q \wedge R)$$
$$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$$
$$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$$
$$P \vee (P \wedge Q) \equiv P$$
$$P \wedge (P \vee Q) \equiv P$$
$$P \Rightarrow Q \equiv \neg P \vee Q$$
$$P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$$
$$\neg \forall x \in S.P(x) \equiv \exists x \in S.\neg P(x)$$
$$\neg \exists x \in S.P(x) \equiv \forall x \in S.\neg P(x)$$

**False demonstrații**

Lipsa de suport justificativ pentru fiecare pas poate conduce cu usurință la false demonstrații. Iată un exemplu:

**Teorema 1.2**: $1 = -1$

"**Demonstratie**": $1 = \sqrt{1} = \sqrt{(-1)(-1)} = \sqrt{-1}\sqrt{-1} = (\sqrt{-1})^2 = -1$

□

Este de presupus că unul din pasii demonstratiei este fals. Dar fiecare pas pare autorului acestei demonstratii a fi "rezonabil". Scriind toate axiomele justificatoare pentru fiecare pas se evidentiază repede că una din axiome este falsă: este pur si simplu neadevărat că pentru oricare două numere $x$ si $y$, $\sqrt{xy} = \sqrt{x}\sqrt{y}$.

Alte erori clasice:

- Împărtirea ambelor părti ale unei ecuatii cu o variabilă. De exemplu:

$$ax = bx \text{ asadar } a = b$$

"Axioma" la care acest pas apelează implicit este falsă deoarece dacă $x = 0$ afirmatia $a = b$ nu mai rezultă. Este necesar un efort suplimentar pentru a arăta că $x$ nu este nul.

- Împărtirea ambelor părti ale unei inegalităti cu o variabilă. De exemplu:

$$ax < bx \text{ asadar } a < b$$

Aici afirmatia $a < b$ este falsă dacă $x < 0$ si neverificată dacă $x = 0$.

**Demonstratia prin cazuri**

Uneori nu se stie care dintr-o multime de cazuri sunt adevărate, dar se stie că cel putin unul din cazuri este adevărat. Dacă rezultatul se poate dovedi în fiecare din cazuri, atunci avem o demonstratie. Propozitia din limba engleză "damned if you do and damned if you don't" (blestemat de faci si blestemat de nu faci) cuprinde, încapsulează această metodă de a demonstra anumite adevăruri. Iată un exemplu simpatic:

**Teorema 1.3**: Există numere irationale $x$ si $y$ pentru care $x^y$ este rational.

**Demonstratie**:

1. Deoarece teorema este cuantificată existential, este necesar să dovedim existenta (a cel putin) unui exemplu. Fie cazul $x = \sqrt{2}$ si $y = \sqrt{2}$. Trebuie să fie adevărat sau că (*a*) $\sqrt{2}^{\sqrt{2}}$ este rational, sau că (*b*) $\sqrt{2}^{\sqrt{2}}$ este irational.
2. În cazul (*a*) am arătat existenta a două numere irationale $x$ si $y$ astfel încât puterea $x^y$ să fie un număr rational.
3. În cazul (*b*) se consideră valorile $x = \sqrt{2}^{\sqrt{2}}$ si $y = \sqrt{2}$. Avem

$$x^y = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\sqrt{2}} = \sqrt{2}^2 = 2$$

cu a doua egalitate bazată pe axioma $(x^y)^z = x^{yz}$. Asadar, am arătat că există numere irationale $x$ si $y$ astfel încât $x^y$ este rational.

4. Deoarece unul din cazurile (a) si (b) trebuie să fie adevărat, urmează că pentru unele numere irationale $x$ si $y$, $x^y$ este rational.

□

De observat că si după demonstratie nu stim încă dacă unul sau altul din cele două cazuri este adevărat, astfel că nu putem produce efectiv numere irationale care să satisfacă teorema. Este aici un exemplu de demonstratie **nonconstructivă**, o demonstratie în care o teoremă existentială este dovedită fără a construi un exemplu.

**Demonstratie prin contradictie**

Se mai numeste si *reductio ad absurdum* (reducere la absurd). Demonstratia prin contradictie este înrudită strâns cu demonstratia prin contrapunere. Ideia

este a presupune opusul, contrarul a ceea ce este de demonstrat si apoi a arăta că în combinatie cu axiomele aceasta duce la o contradictie.
Se consideră exemplul următor. Un **număr rational** este un număr care poate fi exprimat ca raportul a doi întregi. De exemplu, 2/3, 3/5 si 9/16 sunt rationale. **Forma redusă** a unui număr rational este o fractie în care numărătorul si numitorul nu au alt factor comun în afară de 1. De exemplu, forma redusă a lui 3/6 este 1/2. De notat că orice număr cu o reprezentare zecimală finită sau recursivă (periodică) este rational.

**Teorema 1.4**: $\sqrt{2}$ este irational.

**Demonstratie**:

1. Se presupune că $\sqrt{2}$ este rational.
2. Prin definitia unui număr rational, există întregi $a$ si $b$ fără alt factor comun decât 1, astfel încât $\sqrt{2} = a/b$.
3. Pentru orice numere $x$ si $y$, se stie că $x = y \Rightarrow x^2 = y^2$. Asadar, $2 = a^2/b^2$.
4. Prin multiplicarea ambelor părti cu $b^2$ se obtine $a^2 = 2b^2$.
5. $b$ este întreg deci $b^2$ este un întreg si, mai departe, $a^2$ este par (prin definitia paritătii).
6. Asadar, prin teorema prezentată mai devreme, $a$ este par.
7. Asadar, prin definitia paritătii, există un intreg $c$ astfel încât $a = 2c$.
8. Apoi $2b^2 = 4c^2$, de unde $b^2 = 2c^2$.
9. Deoarece $c$ este întreg, $c^2$ este întreg, asadar $b^2$ este par.
10. Asadar, prin teorema prezentată mai devreme, $b$ este par.
11. Asadar, $a$ si $b$ au ca factor comun pe 2, ceea ce contrazice pasul 2.
12. Asadar, pasul 2 este fals, adică $\sqrt{2}$ este irational.

□

**Stil si substanță în demonstratii**

Demonstratiile de mai sus justifică fiecare pas prin apel la o definitie sau la o axiomă generală. Adâncimea cu care se face aceasta în practică este o problemă de gust. Un pas poate fi descompus în pasi încă mai multi, de pildă pasul "Deoarece $a$ este întreg, $(2a^2 + 2a)$ este întreg" (Exercitiu: care sunt acei pasi?). O justificare poate fi declarată fără demonstratie dacă si numai dacă este absolut sigur că (1) este corectă si (2) cititorul este de acord automat că ea este corectă.
De observat că în demonstrarea faptului că $\sqrt{2}$ este irational s-a folosit un rezultat mai vechi, "Pentru orice întreg $n$, dacă $n^2$ este par atunci $n$ este par", si chiar de două ori. Asta sugerează că acel rezultat poate fi util în multe demonstratii. Un rezultat subsidiar care poate fi util într-o demonstratie mai complexă este denumit **lemă**. Uneori este o idee bună a diviza o demonstratie lungă în mai multe leme. Este ceva similar modului în care taskurile de programare mari sunt divizate în subrutine. Mai mult, este bine ca fiecare lemă (ca si fiecare subrutină) să fie făcută cât mai generală posibil pentru a fi reutilizată si în alte ocazii.

Linia de separare dintre leme si teoreme nu este foarte clară. Uzual, când se scrie un articol, teoremele sunt acele propozitii pe care autorul le "exportă" din articol către restul lumii, pe când lemele sunt propozitii utilizate în demonstratiile teoremelor. Există totusi unele leme (de pildă lema pompării – *Pumping Lemma* – si lema ridicării – *Lifting Lemma*) care sunt probabil mai faimoase si mai importante decât teoremele pentru demonstrarea cărora sunt sunt utilizate.

Asupra demonstratiilor si stilului în demonstratii se pot citi mai multe în "Social processes and proofs of theorems and programs" de De Millo, Lipton si Perlis, în *Communication of the ACM*, mai 1979, vol 22, p. 271. O lucrare de consultat în această privintă este si *Mathematical Writing*, de Don Knuth, Mathematical Association of America, 1989.

# Lecția 2

Această lectie acoperă subiectul inductiei, o tehnică dintre cele mai obisnuite pentru demonstrarea propozitiilor cu cuantificare universală peste multimea numerelor naturale si peste alte multimi discrete de obiecte (liste, arbori, asezări gen tigle pe acoperis, pavaje, grafuri, programe etc.). Inductia sau ceva de acest gen este necesară deoarece orice tentativă de a demonstra prin enumerarea unor lumi posibile este sortită esecului – cu elemente infinit de multe este posibil a defini infinit de multe lumi.

**Principiul inductiei**

Principiul inductiei se asează pe o axiomă a numerelor naturale. Să ne reamintim din prima lectie că numerele naturale satisfac axiomele lui Peano (pentru claritate, cu scrierea $n + 1$ în loc de $s(n)$):

> 0 este un număr natural
> Dacă $n$ este un număr natural, atunci $n + 1$ este un număr natural

Aceste axiome nu sunt tocmai definitorii pentru numerele naturale deoarece nu includ o declaratie de forma "… si în legătură cu aceasta, nu există alte numere naturale", ceea ce face imposibil uzul exclusiv al acestor axiome pentru a demonstra că o proprietate se mentine pentru toate numerele naturale $n$.
Principiul inductiei în esentă completează lipsa aceasta. Informal acest principiu spune că:

> Dacă se poate dovedi că o anumită proprietate este adevărată pentru 0 si se poate dovedi că acea proprietate se mentine pentru $n + 1$ dacă este adevărată pentru $n$, atunci s-a demonstrat că acea proprietate se mentine pentru toate numerele naturale.

Pentru a formula acest principiu formal, fie $P(n)$ o propozitie arbitrară despre numărul natural $n$ (de pildă, $P(n)$ ar putea fi "$2^n > n$"). Principiul inductiei este după cum urmează:

**Axioma 2.1** (**Inductia**): Pentru orice proprietate $P$, dacă $P(0)$ si $\forall\ n \in \mathbf{N}\ (P(n) \Rightarrow P(n + 1))$, atunci $\forall\ n \in \mathbf{N}\ P(n)$.

Axioma spune că dacă $P(0)$ este adevărată, si $P(0) \Rightarrow P(1)$, si $P(1) \Rightarrow P(2)$ s.a.m.d. atunci $P(n)$ trebuie să fie adevărată pentru orice $n$. Acest fapt pare destul de rational. (Mai departe se va vedea că axioma are o variantă din care acelasi fapt pare încă mai rational).

**Demonstratii inductive**

Primul exemplu, aproape standard constă în verificarea binecunoscutei formule pentru suma primilor $n$ întregi pozitivi:

**Teorema 2.1**: $\forall\ n \in \mathbf{N}\ \sum_{i=1}^{n} i = n(n+1)/2$.

[Notă: $\sum_{i=1}^{n} i$ înseamnă $1 + 2 + \ldots + n$ într-o scriere mai precisă si mai concisă. Dacă $n = 0$, suma nu are nici un termen si este nulă; pentru $n = 1$ suma are numai termenul $i = 1$ si este egală cu 1].
În multe cazuri de demonstrare prin inductie (dar nu în toate), proprietatea $P$ utilizată în inductie este exact aceea care trebuie demonstrată. Acesta esta cazul aici: $P(n)$ este propozitia $\sum_{i=1}^{n} i = n(n+1)/2$.
Pentru a construi o demonstratie inductivă, se stabileste mai întâi **cazul de bază** $P(0)$, apoi e necesar a se demonstra că $P(n + 1)$ decurge din $P(n)$. Demonstratia aceasta este **pasul inductiv** si implică uzual întregul efort. Ideia cheie a inductiei, totusi, este că pasul inductiv este mai usor decât demonstrarea de la zero a întregii propozitii universale deoarece este de presupus că $P(n)$ este adevărată când se demonstrează $P(n + 1)$. Această presupunere este denumită **ipoteza inductivă**.
**Demonstratie**: Demonstratia este prin inductie pe numerele naturale.

- Cazul de bază: verificarea propozitiei $P(0)$.

$P(0)$ este propozitia $\sum_{i=1}^{0} i = 0(0 + 1)/2$. Prin definitie suma este nulă si $0 = 0$ ceea ce este adevărat.

- Pasul inductiv: demonstrarea propozitiei $P(n) \Rightarrow P(n + 1)$, pentru orice $n \in \mathbf{N}$.

    1. Ipoteza inductivă este $\sum_{i=1}^{n} i = n(n+1)/2$
    2. De demonstrat: $\sum_{i=1}^{n+1} i = (n+1)(n+2)/2$
    3. Re-exprimarea sumei pentru $n + 1$ în functie de suma pentru $n$ (*pentru care există formula de la punctul* 1)

    $$\sum_{i=1}^{n+1} i = \sum_{i=1}^{n} i + (n+1) = n(n+1)/2 + 2(n+1)/2 = (n+1)(n+2)/2$$

Asadar, prin principiul inductiei, $\forall\ n \in \mathbf{N}\ \sum_{i=1}^{n} i = n(n+1)/2$.

□

Acest exemplu ilustrează deplin principiul inductiei care trebuie respectat aproape cu religiozitate. Exemplul ilustrează tehnica obisnuită pentru toate demonstratiile prin inductie: propozitia $P(n + 1)$ este reformulată în fragmente care provin direct din $P(n)$, pentru care există deja un răspuns, si un supliment care este combinat cu răspunsul pentru $P(n)$ pentru a da un răspuns relativ la $P(n + 1)$.

**Demonstratii exemplu**

**Teorema 2.2**: $\forall\ n \in \mathbf{N}$, $n^3 - n$ este *divizibil* cu 3.
Este util a defini mai precis sintagma "divizibil cu". Se scrie $a|b$ ($a$ divide pe $b$, sau $b$ este divizibil cu $a$) si definitia este
**Definitia 2.1** (**divizibilitate**): Pentru întregii $a$ si $b$, $a|b$ dacă si numai dacă pentru un anume întreg $q$, $b = aq$.
Ca si în exemplul precedent, definitia lui $P$ se prelevă din teoremă; $P(n)$ este propozitia $3|(n^3 - n)$.
**Demonstratie**: Demonstratia este prin inductie pe numerele naturale.

- Cazul de bază: demonstrarea propozitiei $P(0)$.

$P(0)$ este propozitia $3|(0^3 - 0)$ sau $3|0$, care este adevărată prin însăsi definitia divizibilitătii, cu $q = 0$

- Pasul inductiv: se demonstrează $P(n) \Rightarrow P(n + 1)$ pentru orice $n \in \mathbf{N}$
    1. Ipoteza inductivă este $3|(n^3 - n)$ sau $n^3 - n = 3q$, cu $q$ întreg.
    2. Demonstrarea propozitiei $3|((n + 1)^3 - (n + 1))$, ceea ce se face arătând că $(n + 1)^3 - (n + 1) = 3r$ pentru un întreg $r$.
    3. Se re-exprimă cantitatea pentru $n + 1$ în functie de cantitatea pentru $n$ (pentru care există o formulă simplă) si se obtine succesiv
    $$(n + 1)^3 - (n + 1) = n^3 + 3n^2 + 3n + 1 - (n + 1) =$$
    $$= (n^3 - n) + 3n^2 + 3n = 3q + 3(n^2 + n) = 3(q + n^2 + n) = 3r$$
    4. Asadar, deoarece $q$ si $n$ sunt întregi, $r$ este întreg si $3|((n + 1)^3 - (n + 1))$.

Concluzia prin principiul inductiei: $\forall\ n \in \mathbf{N}$, $3|(n^3 - n)$.

□

Din nou, trucul este a reduce $P(n + 1)$ la $P(n)$ si un fapt suplimentar este acela că $3|3(n^2 + n)$. Uneori acest pas poate fi executat prin simpla manipulare algebrică constând în a extrage $P(n)$ din $P(n + 1)$; alteori trebuie înteles ceea ce se produce în mod real la un nivel mai profund.
Exemplul următor demonstrează o inegalitate între două functii de $n$; astfel de inegalităti sunt utile în stiinta calculatoarelor când se arată că un algoritm este mult mai eficient fată de altul. De observat că aici cazul de bază este $P(2)$ si nu $P(0)$. Este o variantă evidentă a principiului inductiei în forma standard; se poate porni cu orice întreg fixat pentru a arăta că toti întregii următori satisfac

aceeasi proprietate. (Pentru a demonstra $P(n)$ pentru *toate* numerele întregi, inclusiv cele negative, trebuie făcute două rationamente inductive – unul în sensul crescător de la, să spunem, 0, altul în sens scăzător de la 0).
Există o modalitate de a evita "varianta" aceasta nouă? Putem face ca $P(n)$ să fie $n > 1 \Rightarrow n! < n^n$ si să se demonstreze cazul $P(0)$ la modul trivial deoarece premisa $0 > 1$ este falsă. Desigur, când se demonstrează pasul inductiv, va fi necesar a demonstra $P(0) \Rightarrow P(1)$ si $P(1) \Rightarrow P(2)$ simplu, demonstrând separat $P(1)$ si $P(2)$ si apoi demonstrând $P(n) \Rightarrow P(n + 1)$ pentru orice $n > 1$. Munca este aceeasi dar demonstratia arată mai încurcată.
**Teorema 2.3**: $\forall n \in \mathbf{N}, n > 1 \Rightarrow n! < n^n$.
**Demonstratie**: Demonstratia se face prin inductie pe numerele naturale mai mari decât 1.

- Cazul de bază: verificarea propozitiei $P(2)$.

P(2) înseamnă $2! < 2^2$ sau $2 < 4$, ceea ce este adevărat.

- Pasul inductiv: demonstrarea propozitieie $P(n) \Rightarrow P(n + 1)$ pentru orice $n > 1$.
    1. Ipoteza inductivă este $n! < n^n$
    2. De demonstrat: $(n + 1)! < (n + 1)^{(n + 1)}$
    3. Prin re-exprimarea numerelor comparate pentru $n + 1$ în functie de numerele pentru $n$ (*pentru care este cunoscută o inegalitate simplă*) se obtine:

$$(n + 1)! = (n + 1)n! < (n + 1)n^n < (n + 1)(n + 1)^n = (n + 1)^{(n + 1)}$$

tinând seama de definitia factorialului, de ipoteza inductivă, de faptul că

$$\forall x, y, a \; a > 0 \wedge x < y \Rightarrow ax < ay$$

si încă de faptul că

$$\forall x, y, a \; x > 0 \wedge a > 0 \wedge x < y \Rightarrow x^a < y^a.$$

Asadar, prin principiul inductiei, $\forall n \in \mathbf{N}, n > 1 \; n! < n^n$.
□

**O demonstratie falsă**

**Teorema 2.4**: Toate iMac[1]-urile au aceeasi culoare.
**Demonstratie**: Mai întâi o reformulare: $\forall n \in \mathbf{N}$, dacă $n > 0$ atunci orice multime de $n$ iMac-uri este monocromă. Demonstratia de face prin inductie pe numărul de iMac-uri într-un set.

- Cazul de bază: demonstrarea propozitiei $P(1)$

$P(1)$ este propozitia care afirmă că orice multime de 1 iMac este monocromă. Este evident adevărată.

- Pasul inductiv: demonstrarea propozitieie $P(n) \Rightarrow P(n + 1)$ pentru toate numerele naturale $n > 0$.
    1. Ipoteza inductivă spune că fiecare multime de $n$ iMac-uri este monocromă
    2. De demonstrat: fiecare set de $n + 1$ iMac-uri este monocromă

[1] Denumire comercială a unor calculatoare.

3. Pentru fiecare set de $n + 1$ iMac-uri, $S = \{i_1, \ldots, i_{n+1}\}$, se consideră submultimile de $n$ elemente $S_1 = \{i_1, \ldots, i_n\}$ si $S_2 = \{i_2, \ldots, i_{n+1}\}$. Aceste multimi au $n - 1$ elemente comune si reuniunea lor este $S$.
4. Prin ipoteza inductivă, $S_1$ este monocromă. Prin urmare $i_1$ are aceeasi culoare cu $i_n$.
5. Prin ipoteza inductivă, $S_2$ este monocromă. Prin urmare $i_n$ are aceeasi culoare cu $i_{n+1}$.
6. Asadar, toate elementele din $S$ au aceeasi culoare, adică fiecare multime de $n + 1$ iMac-uri este monocromă.

Asadar, din principiul inductiei, toate iMac-urile sunt de aceeasi culoare.

□

Desi concluzia este evident absurdă, eroarea din demonstratie nu este usor de diagnosticat deoarece pasul inductiv *este* corect pentru un caz "tipic" cum este, să spunem, $n = 3$. Este realmente adevărat că dacă orice multime de 3 elemente este monocromă atunci orice multime de 4 elemente este monocromă; si la fel pentru întregi $n$ mai mari. Trebuie văzut ceva mai adânc. Este clar că $P(1)$ este adevărată dar $P(2)$ este falsă, asa că eroarea trebuie să provină din pasul inductiv când $n = 1$. În acel caz, $S$ are două elemente si submultimile de $n$ elemente sunt singletonuri $\{i_1\}$ si $\{i_2\}$. Pasul 4 este corect deoarece $i_1$ si $i_n$ sunt acelasi obiect. Eroarea apare în pasul 5: spunem corect că $S_2 = \{i_2, \ldots, i_{n+1}\}$, dar asta *nu* înseamnă că $i_n$ este un element din $S_2$! Mai intuitiv, pasul inductiv se bazează pe o intersectie nevidă a multimilor $S_1$ si $S_2$ astfel încât culoarea se poate "propaga" de la $S_1$ la $S_2$. Când $S$ are numai 2 elemente nu acesta este cazul.

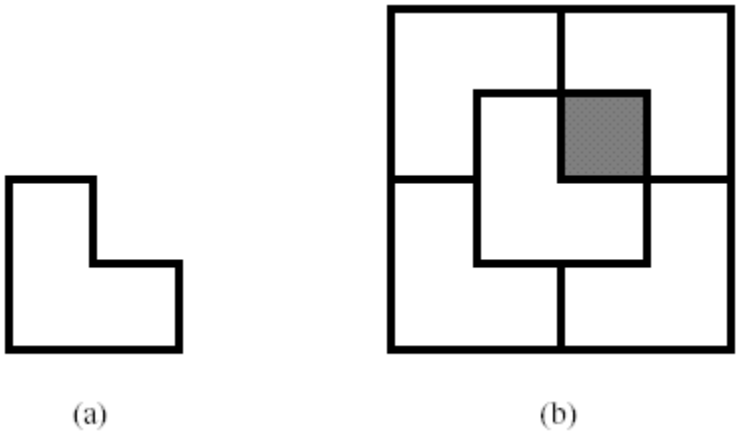
## Întărirea ipotezei inductive

În toate exemplele precedente s-a luat proprietatea $P(n)$ direct din enuntul teoremei de demonstrat. Aceasta nu operează totdeauna; uneori ipoteza inductivă nu este capabilă a da suficientă "sevă" pentru a lega o proprietate de următoarea. Iată imediat un exemplu.

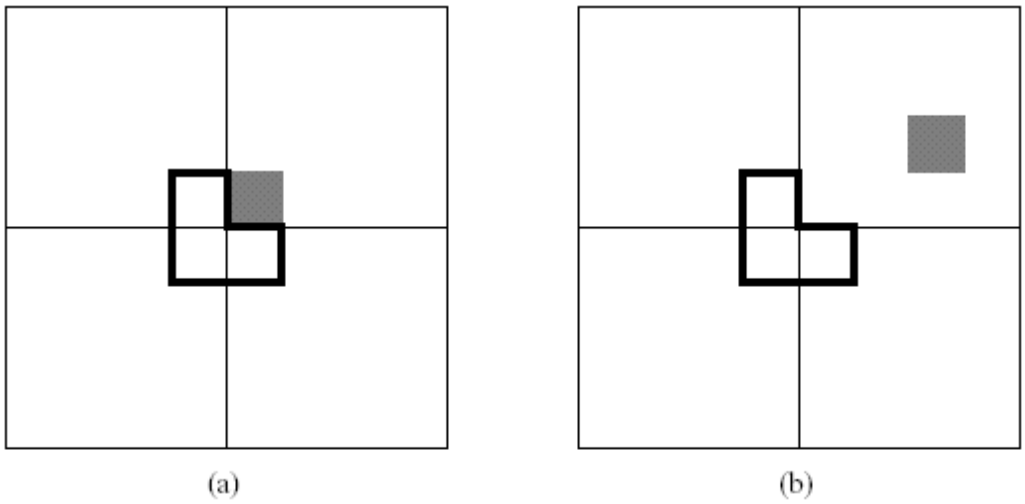
Studiul acoperirii (tiling) este un joc matematic cu implicatii serioase în proiectarea circuitelor, în fizica solidului si în alte domenii. O *acoperire* este o chiar o acoperire în sensul comun a unei regiuni (de regulă plană) cu exemplare multiple ale unei aceleiasi forme de dimensiuni mai mici, desigur. Unul din rezultatele cele mai cunoscute este acela că o regiune 8x8 cu două pătrate situate în vârfuri opuse extrase, detasate nu poate fi acoperită (tiled) cu pietre de domino 2x1.

În cazul adus aici în discutie, un departament de stiinta calculatoarelor obsedat de binar intentionează să-si facă o clădire nouă în jurul unei curti de dimensiunea $2^n x 2^n$ pentru un anumit $n$. Un constructor de stat cam inept a aprovizionat departamentul cu pietre în formă de L pentru a pava curtea (v.figura). Un student priceput care a urmat si a trecut disciplina Matematici discrete aplicate a demonstrat că $\forall\ n\ 3|2^{2n} - 1$ astfel că orice acoperire lasă un

gol, o portiune neacoperită. "N-are importantă" spune seful de catedră tot timpul priceput, "Lăsăm golul în centru si punem acolo o flatantă statuie pentru donatorul de fonduri pentru clădire, pe care încă îl căutăm", stiind oarecum că acesta va fi Pokemon Toys Inc.




(a) O piatră în formă de L.
(b) O acoperire a unei regiuni 4x4 cu astfel de pietre cu o gaură în mijloc




(a) Pasul inductiv pentru prima tentativă de demonstratie: o regiune de dimensiunile $2^{n+1}$x$2^{n+1}$ cu o gaură centrală poate fi divizată în patru regiuni de $2^n$x$2^n$ ale căror patru colturi centrale pot fi acoperite cu o gaură si o piatră în formă de L
(b) Pasul inductiv pentru a doua tentativă de demonstratie: o regiune de dimensiunile $2^{n+1}$x$2^{n+1}$ cu o gaură situată oriunde poate fi divizată în patru regiuni de $2^n$x$2^n$; una din cele patru regiuni contine gaura si cele trei colturi centrale ale celorlalate regiuni pot fi acoperite cu o piatră în formă de L

**Teorema 2.5**: $\forall\ n \in \mathbf{N}$, orice regiune de dimensiunile $2^n \times 2^n$ cu o gaură centrală are o acoperire L exactă.

**Demonstratie**: (Prima tentativă) Demonstratie prin inductie pe numerele naturale.

- Cazul de bază: demonstratia propozitiei $P(0)$.

P(0) este propozitia care spune că o regiune $1 \times 1$ cu o gaură centrală are o acoperire L exactă. Este adevărată, deoarece sunt necesare 0 pietre.

- Pasul inductiv: demonstratia propozitiei $P(n) \Rightarrow P(n + 1)$ pentru orice $n \in \mathbf{N}$.
  1. Ipoteza inductivă afirmă că fiecare regiune de dimensiunea $2^n \times 2^n$ cu o gaură centrală are o acoperire L exactă.
  2. De demonstrat: fiecare regiune de dimensiunea $2^{n+1} \times 2^{n+1}$ cu o gaură centrală are o acoperire L exactă.
  3. Regiunea $2^{n+1} \times 2^{n+1}$ poate fi divizată în patru regiuni de dimensiunea $2^n \times 2^n$ ale căror colturi pot fi acoperite cu o gaură si o piatră sub formă de L (v.figura).
  4. Fiecare dintre cele patru regiuni rezultate sunt de dimensiune $2^n \times 2^n$ cu o gaură. Asadar, prin ipoteza inductivă, are o acoperire L exactă.
  5. Dacă un set de regiuni disjuncte au o acoperire exactă, reuniunea lor are o acoperire exactă (De notat: aceasta necesită o inductie separată!). Asadar, fiecare regiune de dimensiunea $2^{n+1} \times 2^{n+1}$ cu o gaură centrală are o acoperire L exactă.

Stop, stop, stop. Utilizarea ipotezei inductive în pasul 4 este discutabilă! Ipoteza inductivă garantează o acoperire L pentru regiuni cu gaură *centrală*, nu cu o gaură situată periferic, *la colt*.

□

Când apare un astfel de impas, trebuie luată în considerare *întărirea* teoremei de demonstrat. Chestiunea aceasta nu este tocmai intuitivă – dacă n-a fost posibilă demonstrarea unei teoreme, are sens încercarea de a demonstra una mai dificilă?

Demonstrarea unei teoreme mai tari are avantajul întăririi ipotezei inductive $P(n)$, care poate abilita o demonstratie pentru $P(n + 1)$. Dar, atentie!, ea întăreste totodată si $P(n + 1)$, ceea ce poate fi un dezavantaj. Uneori procedeul lucrează, alteori nu. Deocamdată nu există o retetă de succes dincolo de "explorare!" sau "practică!"

Pentru exemplul curent, întărirea evidentă este renuntarea la restrictia ca gaura să fie centrală.

**Teorema 2.6**: Orice regiune de dimensiunile $2^n \times 2^n$ cu o gaură are o acoperire L exactă.

Aceasta este o teoremă mai tare deoarece gaura poate fi acum oriunde si nu neapărat în centru; teorema anterioară este un caz particular. Dacă este cunoscut faptul $P'(n)$ pentru un $n$ particular, atunci apare posibilitatea certă de a completa vechiul pas inductiv $P(n + 1)$ din încercarea precedentă de demonstratie,

deoarece acum pot fi acoperite si regiuni cu colturi lipsă. Aceasta ilustrează avantajul întăririi teoremei.

Din nefericire, nu mai interesează demonstratia propozitiei vechi $P(n + 1)$ – "Fiecare regiune de dimensiunea $2^{n+1}$x$2^{n+1}$ cu o gaură centrală are o acoperire L exactă". În locul ei trebuie demonstrată propozitia mai tare $P'(n + 1)$ – "Fiecare regiune de dimensiunea $2^{n+1}$x$2^{n+1}$ cu o gaură (situată oriunde) are o acoperire L exactă". Din fericire asta se poate demonstra fiind dată $P'(n)$. Demonstratia este sugerată în figura (b) de mai sus si detaliile sunt lăsate în seama cititorului.

Sunt în acest exemplu cel putin încă două aspecte interesante. Primul, demonstratia inductivă se traduce foarte natural într-un algoritm recursiv de *construire* a unei acoperiri L. Al doilea, spre deosebire de demonstratiile precedente, demonstratia este de fapt *informală*. Structura argumentului inductiv este perfect în regulă, dar revendicările detaliate au ca suport apelul la o reprezentare grafică si nu la axiome generale. Asta se întâmplă pentru că nu s-au formulat axiome asupra formelor bidimensionale, asupra pavajelor, asupra divizării regiunilor etc. Pentru cazuri simple baza poate fi intuitia umană asupra acestor subiecte, dar pentru "pavarea" unor regiuni 11-dimensionale care au găuri 9-dimensionale cu "pietre" 10-dimensionale (ca în teoria sirurilor sau în planificarea traiectoriilor de deplasare a robotilor), *trebuie* revenit la o bază axiomatică.

# Lecția 3

Această lectie acoperă alte variante ale inductiei, inclusiv cea a inductiei tari si suplimentul axiomelor de bună ordonare strâns legate de inductia tare. Aceste tehnici sunt apoi aplicate pentru a dovedi proprietătile programelor recursive simple.

## Inductia tare

**Axioma 3.1** (**Inductia tare**): Pentru orice proprietate $P$, dacă $P(0)$ si $\forall\ n \in \mathbf{N}$ $(P(0) \wedge P(1) \wedge \ldots \wedge P(n) \Rightarrow P(n+1))$, atunci $\forall\ n \in \mathbf{N}\ P(n)$
Aceasta spune că dacă toate propozitiile care urmează sunt adevărate:

$$P(0)$$
$$P(0) \Rightarrow P(1)$$
$$P(0) \wedge P(1) \Rightarrow P(2)$$
$$P(0) \wedge P(1) \wedge P(2) \Rightarrow P(3)$$
$$P(0) \wedge P(1) \wedge P(2) \wedge P(3) \Rightarrow P(4)$$

s.a.m.d., atunci $P(n)$ trebuie să fie adevărată pentru orice $n$. Intuitiv, acest fapt pare destul de usor de înteles. Dacă adevărul lui $P$, pe toată secventa de la 0 la $n$ implică totdeauna adevărul lui $P(n + 1)$, atunci se obtine nemijlocit adevărul lui $P$ pe toată secventa până la $n + 1$, care implică adevărul pentru $P(n + 2)$ si tot asa *ad infinitum*.
Dacă comparăm axioma inductiei tari cu axioma inductiei din lectia anterioară, se poate vedea că inductia tare pare *a facilita* demonstrarea lucrurilor. Cu inductia simplă, se poate demonstra $P(n + 1)$ fiind dată ipoteza inductivă $P(n)$; cu inductia tare trebuie presupusă adevărată ipoteza $P(0) \wedge P(1) \wedge \ldots \wedge P(n)$ care este mult mai tare.
Se consideră exemplul următor care este jumătate din teorema fundamentală a aritmeticii (celalată jumătate spune că produsul este unic).
**Teorema 3.1**: Orice număr natural $n > 1$ poate fi scris ca un produs de numere prime.
Pentru a demonstra aceast fapt trebuie mai întâi definite numerele prime:
**Definitia 3.1** (**Numere prime**): Un număr natural este prim dacă si numai dacă are exact doi factori (1 si $n$). 1 nu este el însusi un număr prim.
Să vedem ce se întâmplă când se încearcă inductia simplă.
**Demonstratie** (încercarea 1): Demonstratie prin inductie peste numerele naturale mai mari ca unitatea.

- Cazul de bază: demonstrarea propozitiei $P(2)$.

P(2) este propozitia care spune că 2 poate fi scris ca un produs de numere prime. Ea este adevărată pentru că 2 se poate scrie ca produs de un singur număr prim, el însusi. (Reamintim că 1 nu este număr prim!)

- Pasul inductiv: demonstrarea propozitiei $P(n) \Rightarrow P(n + 1)$, pentru toate numerele naturale $n > 1$.
    1. Ipoteza inductivă spune că $n$ poate fi scris ca un produs de numere prime
    2. De demonstrat: $n + 1$ poate fi scris ca un produs de numere prime
    3. Demonstratia este blocată: fiind dată $P(n)$ se poate stabili usor $P(2n)$ sau $P(7n)$, dar $P(n + 1)$ nu se leagă în vreun fel convenabil de $P(n)$.

□

Cu inductia tare se poate face o conexiune între $P(n + 1)$ si fapte anterioare din secventă, fapte care sunt de data aceasta relevante. De exemplu, dacă $n + 1 = 72$, atunci $P(36)$ si $P(24)$ sunt fapte utilizabile în demonstratie.

**Demonstratie**: Demonstratie prin inductie tare peste numerele naturale mai mari ca unitatea.

- Cazul de bază: se demonstrează $P(2)$ ca mai devreme.
- Pasul inductiv: se demonstrează că $P(2) \wedge \ldots \wedge P(n) \Rightarrow P(n + 1)$ pentru toate numerele naturale $n > 1$.
    1. Ipoteza inductivă: se demonstrează propozitia $P(2) \wedge \ldots \wedge P(n) \Rightarrow P(n + 1)$ pentru toate numerele naturale $n > 1$.
    2. De demonstrat: $n + 1$ poate fi scris ca un produs de numere prime.
    3. Demonstratie pe cazuri:
        - $n + 1$ este prim: atunci $n + 1$ poate fi scris ca produsul unui număr prim, el însusi
        - $n + 1$ nu este prim: atunci prin definitia numerelor prime, există întregi $a$, $b$ astfel încât $2 \leq a, b < n + 1$ si $n + 1 = a.b$. Prin ipoteza inductivă, atât $a$ cât si $b$ pot fi scrise ca un produs de numere prime. Asadar $n + 1$ poate fi scris ca un produs de numere prime.

□

**Teorema 3.2**: Orice cantitate întreagă de trimiteri postale cu taxa de la 8¢ în sus poate fi acoperită exact cu timbre de 3¢ si de 5¢.

Cu o inductie tare se poate face o conexiune între $P(n + 1)$ si faptele anterioare din secventă. În particular, $P(n - 2)$ este relevantă pentru că $n + 1$ poate fi compus din solutia pentru $n - 2$ plus o marcă de 3¢. Astfel pasul inductiv lucrează dacă adevărul propozitiei $P(n - 2)$ este deja cunoscut. Nu aceasta va fi situatia dacă $n + 1$ este 9 sau 10, astfel încât acestea vor trebui tratate separat.

**Demonstratie**: demonstratia se face prin inductie tare pe numerele naturale $n \geq 8$.

- Cazul de bază: se demonstrează $P(8)$.

$P(8)$ este propozitia care spune că taxa pentru o trimitere postală de 8¢ poate fi compusă din timbre de 3¢ si 5¢. Propozitia este adevărată si necesită câte un timbru din fiecare tip/valoare.

- Pasul inductiv: se demonstrează că $P(8) \wedge \ldots \wedge P(n) \Rightarrow P(n+1)$ pentru toate numerele naturale $n \geq 8$.
    1. Ipoteza inductivă spune că, pentru toate numerele naturale $m$ de la 8 la $n$, orice taxă de $m¢$ poate fi compusă din mărci de 3¢ si 5¢.
    2. De dovedit: taxa postală de $(n+1)¢$ poate fi compusă din mărci de 3¢ si 5¢.
    3. Cazurile când $n+1$ este 9 sau 10 trebuie verificate separat. Pentru o trimitere cu taxe de 9¢ poate fi timbrată cu mărci de 3¢ si una cu taxe de 10¢ poate fi timbrată cu mărci de 5¢.
    4. Pentru toate numerele naturale $n + 1 > 10$, ipoteza inductivă decurge din propozitia $P(n-2)$. Dacă taxa de $(n-2)¢$ poate fi compusă din timbre de 3¢ si 5¢ atunci, simplu, taxa de $(n+1)¢$ poate fi compusă din mărci de 3¢ si de 5¢ prin adăugarea unei mărci de 3¢.

□

De observat că, precum în problema acoperirii cu tigle, cu pietre de pavaj, demonstratia inductivă conduce la un algoritm recursiv simplu de a selecta combinatia de timbre.

De observat de asemenea că o demonstratie prin inductie tare poate reclama verificarea mai multor "cazuri speciale" pentru a stabili o fundamentare solidă a secventei din pasul inductiv. Este usor a examina unul sau mai multe astfel de cazuri.

**Inductia simplă si inductia tare**

S-a văzut că inductia tare facilitează anumite demonstratii chiar când inductia simplă esuează. O întrebare naturală este dacă axioma inductiei tari este de fapt *mai tare logic* decât axioma inductie simple; dacă este asa, atunci teoremele care pot fi demonstrate prin inductia tare fac parte dintr-o supramultime strictă a teoremelor care pot fi demonstrate prin inductia simplă.

Să investigăm problema. Mai întâi, axioma inductiei simple este o consecintă a axiomei inductiei tari? Intuitiv, acesta pare a fi adevărul. Să punem cele două axiome alături si să le examinăm structura (luăm ca implicită restrictia la numerele naturale):

Simplă: $P(0) \wedge [\forall n\ P(n) \Rightarrow P(n+1)] \Rightarrow \forall n\ P(n)$

Tare: $P(0) \wedge [\forall n\ P(0) \wedge \ldots \wedge P(n) \Rightarrow P(n+1)] \Rightarrow \forall n\ P(n)$

Aceste propozitii pot fi reduse la următoarele două forme de bază (cu propozitiile $A$, $B$, $B'$ si $C$ cu semnificatie evidentă):

Simplă: $A \wedge B \Rightarrow C$

Tare: $A \wedge B' \Rightarrow C$

Acum, dacă $P(n) \Rightarrow P(n+1)$, atunci $P(0) \wedge P(1) \wedge \ldots \wedge P(n) \Rightarrow P(n+1)$. Asadar, $B \Rightarrow B'$ (adică $B$ este mai tare decât $B'$). Asadar, dacă propozitia $A \wedge B'$ este suficientă pentru a demonstra $C$, atunci cu sigurantă faptul mai tare $A \wedge B$ este de asemenea suficient pentru a demonstra $C$. (Aceasta se verifică usor apelând la tabele de adevăr). Prin urmare, axioma inductiei tari decurge din axioma inductiei simple.
Al doilea, axioma inductiei simple decurge din axioma inductiei tari? S-ar părea că nu, dar de fapt ea decurge. Se poate vedea aceasta prin definirea, pentru o proprietate oarecare $P(n)$, a propozitiei

$$Q(n) \Leftrightarrow P(0) \wedge P(1) \wedge \ldots \wedge P(n)$$

Asta înseamnă că $Q(n)$ este propietatea "$P$ se mentine adevărată de la 0 la $n$". Ideea este că inductia simplă utilizând $Q$ este de fapt identică cu inductia tare utilizând $P$. Axioma inductiei simple prin $Q$ este

$$Q(0) \wedge [\forall\, n\, Q(n) \Rightarrow Q(n+1)] \Rightarrow \forall\, n\, Q(n)$$

Prin expandarea definitiei lui $Q$, se obtine

$$P(0) \wedge [\forall\, n\, P(0) \wedge \ldots \wedge P(n) \Rightarrow P(0) \wedge \ldots \wedge P(n) \wedge P(n+1)] \Rightarrow \\ \Rightarrow \forall\, n\, P(0) \wedge \ldots \wedge P(n)$$

O gândire de câteva clipe (se recomandă cititorului să facă singur acest rationament) scoate la iveală faptul că această propozitie este echivalentă logic cu propozitia

$$P(0) \wedge [\forall\, n\, P(0) \wedge \ldots \wedge P(n) \Rightarrow P(n+1)] \Rightarrow \forall\, n\, P(n)$$

care este axioma inductiei tari. Astfel am arătat (probabil mai curând informal) următoarea
**Teoremă 3.3**: Axioma inductiei tari si axioma inductiei simple sunt echivalente logic.
De ce există atunci două forme diferite de inductie? Răspunsul este că inductia tare reaminteste utilizatorului ei de oportunitatea utilizării în pasul inductiv a propozitiei $P(0) \wedge \ldots \wedge P(n)$ când $P(n)$ este definită în modul "natural" din enuntarea teoremei de demonstrat.

**Principiul bunei ordonări**

Dacă cineva cumpăneste putin asupra modului cum functionează inductia, poate formula întrebarea "Cum ar putea axioma inductiei să nu fie adevărată?" Pentru a fi în contradictie cu axioma inductiei ar trebui să fie satisfăcut antecedentul ei (astfel $P(0)$ si $P(n) \Rightarrow P(n+1)$ sunt adevărate pentru orice $n$) concomitent cu violarea consecintei ei (astfel $\exists\, n\, \neg P(n)$). Să considerăm primul $n$ pentru care

$P(n)$ este falsă. Prin definitie stim că $P(n-1)$ este adevărată; si prin ipoteză se stie că $P(n-1) \Rightarrow P(n)$; asadar apare o contradictie directă, evidentă.
Am *demonstrat* axioma inductiei? Nu! Am verificat doar că axioma inductiei decurge dintr-o altă axiomă, care a fost folosită implicit pentru definirea "primului $n$ pentru care $P(n)$ este falsă".
**Axioma 3.2** (**Buna ordonare**): Orice multime de numere naturale nevidă are un cel mai mic element.
Dar nu are orice submultime nevidă de elemente ordonabile un cel mai mic element? Nu! Orice multime *finită* are un cel mai mic element, dar nu *orice* multime pur si simplu, care poate fi *infinită*. De exemplu, nici întregii, nici măcar numerele rationale pozitive nu au un cel mai mic element.
Principiul bunei ordonări în afară că dă substantă axiomelor inductiei are si utilitate directă, de sine stătătoare. Un exemplu particular elegant priveste existenta ciclurilor în turnee (tournaments).
**Definitia 3.2** (**Round-Robin**): Un turneu Round-Robin este unul în care fiecare jucător $p$ joacă cu fiecare alt jucător $q$ exact o dată si fie câstigă ($p \succ q$), fie pierde ($q \succ p$).
**Definitia 3.3** (**Ciclu**): Un ciclu într-un turneu este un set de jucători $\{p_1, \ldots p_k\}$ astfel încât $p_1 \succ p_2 \succ \ldots \succ p_{k-1} \succ p_k \succ p_1$.
**Teorema 3.4**: În orice turneu Round-Robin, dacă există un ciclu, atunci există un ciclu de lungime 3.
**Demonstratie**: Demonstratia este prin contradictie.

1. Presupunem că teorema este falsă. Considerăm multimea de lungimi ale ciclurilor din turneu. Prin ipoteză, aceasta trebuie să nu fie vidă.
2. Din principiul bunei ordonări, multimea aceasta trebuie să aibă un cel mai mic element $k$. Prin ipoteză, $k > 3$.
3. Fie $p_1$, $p_2$, $p_3$ primele trei elemente din ciclul de lungime $k$ si luăm în consideratie rezultatul meciului dintre $p_1$ si $p_3$.
4. Cazul 1: $p_1 \succ p_3$. Atunci avem $p_1 \succ p_3 \succ \ldots \succ p_{k-1} \succ p_k \succ p_1$, adică un ciclu de lungime $k-1$, ceea ce contrazice presupunerea că cel mai scurt ciclu are lungimea $k > 3$.
5. Cazul 2: $p_3 \succ p_1$. Atunci avem $p_1 \succ p_2 \succ p_3 \succ p_1$, adică un ciclu de lungime 3, ceea ce contrazice din nou presupunerea că cel mai mic ciclu are lungimea $k > 3$.
6. Prin definitia turneelor Round-Robin, sau $p_1 \succ p_3$ sau $p_3 \succ p_1$. Asadar, există o contradictie.
7. Prin urmare, trebuie să existe cazul în care orice turneu cu un ciclu are un ciclu de lungime 3.

□
Această demonstratie ilustrează o cale uzuală de a utiliza buna ordonare combinată cu demonstratia prin contradicitie. Principiul bunei ordonări permite a focaliza discutia pe un contraexemplu concret, cu proprietetatea că orice exemplu mai restrâns satisface o proprietate anumită. Pentru unele demonstratii, acesta poate fi un proces de gândire mai usor decât inductia.

**Inductia si recursia**

Există o legătură strânsă între inductie si recursie. În esentă fiecare functie recursivă îsi bazează corectitudinea pe o demonstratie inductivă. Să ne reamintim că o functie recursivă se aplică ea însăsi pe un argument "mai mic". Demonstratia inductivă spune că dacă functia inductivă lucrează pe toate argumentele mai mici ea va lucra si pe argumentul curent.
Vom începe cu o veche cunostintă: functia factorial. Să dăm o definitie recursivă pentru o functie $f(n)$ si să arătăm că ea este identică cu $n!$. Pentru orice $n \in \mathbf{N}$,

$f(n) = 1$ dacă $n = 0$
$f(n) = n\, f(n-1)$ în toate celelalte cazuri

**Teorema 3.5**: Pentru orice număr natural $n$, $f(n) = n!$
**Demonstratie**: Demonstratia este prin inductie pe numerele naturale. Fie $P(n)$ propozitia conform căreia $f(n) = n!$

- Cazul de bază: demonstrarea lui $P(0)$.

$P(0)$ este propozitia conform căreia $f(0) = 0!$. Prin definitia de mai sus, $f(0) = 1 = 0!$, asadar $P(0)$ este adevărată.

- Pasul inductiv: demonstratia propozitiei $P(n) \Rightarrow P(n+1)$ pentru orice $n \in \mathbf{N}$.
  1. Ipoteza inductivă este $f(n) = n!$.
  2. De demonstrat: $f(n+1) = (n+1)!$.
  3. Prin definitia de mai sus

$$\begin{aligned} f(n+1) &= (n+1).f(n) \quad \text{deoarece } n \in \mathbf{N} \text{ astfel că } (n+1) \neq 0 \\ &= (n+1).n! \quad \text{prin ipoteza inductivă} \\ &= (n+1).n.(n-1)\ldots 1 = (n+1)! \end{aligned}$$

Asadar, pe baza principiului inductiei, $\forall\, n \in \mathbf{N}\ f(n) = n!$.
□

**Functii matematice si programe reale**

Discutia de mai sus se aplică la o definire pur matematică a lui $f(n)$. Dacă rationamentul se referă la un program real, atunci programul trebuie mai întâi să fie scris într-un limbaj real, cum ar fi Scheme:

```
(define (factorial n)
  (if (= n 0)
      1
    (* n (factorial (- n 1)))))
```

Declaratia de corectitudine a unui program nu este asa de directă ca pentru o functie definită matematic.

**Teorema 3.6**: Pentru orice număr natural $n$ (în reprezentarea lor în calculator), rezultatul evaluării expresiei (`factorial` $n$) este reprezentarea în calculator a lui $n!$.

Demonstratia că programul este corect este foarte asemănătoare demonstratiei că functia recursivă are valori corecte. Despre notatii: elementele sintactice reale ale limbajului de programare sunt în alte caractere, în timp ce variabilele care dau domeniul lor sunt înclinate.

**Demonstratie**: Demonstratia este prin inductie pe numerele naturale. Fie $P(n)$ propozitia (`factorial` $n$) = $n!$.

- Cazul de bază: verificarea propozitiei $P(0)$.

$P(0)$ este propozitia (`factorial` 0) = 0!. Prin definitia de mai sus,

```
(factorial 0)
    = (if (= 0 0) 1 (* 0 (factorial (- 0 1))))
    = 1   prin evaluarea lui if
```

- Pasul inductiv: demonstratia propozitiei $P(n) \Rightarrow P(n+1)$ pentru orice $n \in \mathbf{N}$.
  1. Ipoteza inductivă este (`factorial` $n$) = $n!$.
  2. De demonstrat: (`factorial` ($n$ + 1)) = ($n$ + 1)!.
  3. Prin definitia de mai sus,

(`factorial` ($n$ + 1))
= (`if (=` ($n$ + 1) `0) 1 (*` ($n$ + 1) `(factorial (-` ($n$ + 1) `1))))`
= (`*` ($n$ + 1) `(factorial (-` ($n$ + 1) `1)))` deoarece $n \in \mathbf{N}$ si $(n+1) \neq 0$
= (`*` ($n$ + 1) `(factorial` $n$))
= ($n$ + 1)!

Prin urmare, principiul inductiei conduce la $n \in \mathbf{N}$ (`factorial` $n$) = $n!$.

□

Note asupra demonstratiei:

- Se face aici apel implicit la mai multe aspecte ale evaluării de programe cum sunt legătura între parametri, definitia expresiilor `if` si corespondenta între functia matematică "–" si functia intrinsecă "–". Aceste leme sunt o parte esentială a definirii limbajului de programare si poate fi declarată si demonstrată o dată pentru toti si pentru totdeauna.
- Teorema asa cum este enuntată este aproape sigur falsă! O probă *reală* de corectitudine, pentru o teoremă potrivit redusă, trebuie să manevreze diferentele între entitătile matematice si entitătile corespunzătoare din calculator. De pildă, $n!$ este bine definită pentru orice număr natural, dar (`factorial` $n$) esuează dacă $n$ este suficient de mare pentru a produce depăsire la multiplicarea de întregi. Un alt mod de a spune aceasta este că operatorul * nu este acelasi cu functia de multiplicare matematică.
- Asa cum e definită, functia (`factorial` $n$) produce o eroare pentru intrări nenumerice, neîntregi sau negative. (Exercitiu: Ce eroare apare când

argumentul este negativ?). O specificare completă pentru un sistem realmente robust trebuie să producă răspunsuri corecte la orice intrări posibile.

În ce mai mare parte a expunerii, se vor utiliza definitiile matematice mai curând decât definitiile Scheme deoarece fac demonstratiile mai curate tipografic si teoremele adevărate.

# Lecția 4

Această lectie completează metodele generale de demonstratie prin inductie. La început sunt date exemple de principii inductive pentru domenii diferite de cel al numerelor naturale, domenii care includ siruri, arbori si perechi de numere. Apoi va fi introdus principiul general de inductie bine fundamentat, care generează de la caz la caz o gamă de principii inductive potrivite unor varii domenii.

## Inducția pentru obiecte diferite de numere

Persoane, altele decât matematicienii puri scriu uneori programe care manipulează alte obiecte decât numere naturale – de exemplu siruri, liste, masive, tabele hash[2], programe, orare aeriene si multe, multe altele. Până acum exemplele de inductie expuse s-au ocupat de inductia pe numere naturale. Cum ajută asta în alte domenii?
Un răspuns este că se pot face demonstratii inductive pe numere naturale care corespund *dimensiunii* obiectelor considerate. Se presupune că se urmăreste a demonstra că $\forall\ s$, $P(s)$ pe un domeniu de **siruri**. Apoi se defineste o propozitie pe numere întregi astfel:

$Q(n)$ este proprietatea că orice sir $s$ de lungime $n$ satisface $P(s)$.

Apoi o demonstratie prin inductie a faptului că $\forall\ n$, $Q(n)$ stabileste că $\forall\ s$, $P(s)$.
Similar, se pot dovedi lucruri despre arbori prin inductie după adâncimea arborelui sau despre programe după numărul de simboluri din program. Aceste genuri de inductie pot deveni foarte complicate si nenaturale. Să presupunem că n-am auzit vreodată de numere naturale; se poate totusi face ceva cu sirurile, arborii, programele? Se dovedeste că se pot defini principii inductive foarte naturale pentru aceste tipuri de obiecte fără a mentiona în vreun fel numere.

## Un principiu inductiv pentru siruri

Să scriem un algoritm recursiv pentru *inversarea* unui sir si să arătăm că acesta lucrează corect.
Mai întâi va trebui să spunem ce este acela un sir. Elementele unui sir sunt **simboluri** extrase dintr-o multime de simboluri numită **alfabet**, notată uzual cu

[2] Hash function – o functie care face două operatii: toacă si amestecă în scopul unic de a realiza o amprentă a unui articol-dată, amprentă denumită uzual hash value. Valoarea hash se prezintă uzual ca un sir de caractere alfanumerice aleator, mai curând straniu. O functie hash "bună" este o functie hash care asigură cât mai putine coliziuni, coliziuni în sensul reprezentării prin amprente identice a două (sau mai multe) articole-dată.

$\Sigma$. De exemplu, dacă $\Sigma = \{a, b\}$, atunci un sir poate fi o secventă de $a$-uri si de $b$-uri. $\Sigma^*$ notează multimea tuturor sirurilor posibile pe alfabetul $\Sigma$ si include totdeauna sirul vid, notat cu $\lambda$. Fiecare simbol din $\Sigma$ este totodată si un sir de lungime 1. (Notă: această proprietate distinge în particular sirurile de liste; dar în general, rationamentele asupra sirurilor sunt similare cu rationamentele asupra listelor).

Calea principală de a construi siruri este **concatenarea**. Dacă $s_1$ si $s_2$ sunt siruri, concatenatul lor este tot un sir si este scris ca $s_1s_2$ sau $s_1.s_2$ dacă o sriere mai clară reclamă eventual punctul. Concatenarea se defineste astefel:

**Axioma 4.1** (Concatenarea):

$\forall\ s \in \Sigma^*\ \ \lambda.s = s.\lambda = s$

$\forall\ a \in \Sigma\ \forall\ s_1, s_2 \in \Sigma^*\ \ (a.s_1).s_2 = a.(s_1.s_2)$

Ceea ce a făcut Peano pentru numerele naturale, se face aici pentru a furniza axiome referitoare la siruri, apoi se va formula un principiu inductiv care permite demonstratii pentru toate sirurile. Sirurile satisfac următoarele axiome:

**Axioma 4.2** (Siruri):

Sirul vid este un sir: $\lambda \in \Sigma^*$

Adăugând un simbol la un sir se obtine un sir: $\forall\ a \in \Sigma\ \forall\ s \in \Sigma^*\ \ a.s \in \Sigma^*$.

Deoarece aceste axiome nu *definesc* strict sirurile, apare necesar un principiu inductiv pentru a construi demonstratii pe toate sirurile:

**Axioma 4.3** (inductia pe siruri):

Pentru orice proprietate $P$,

dacă $P(\lambda)$ si $\forall\ a \in \Sigma\ \forall\ s \in \Sigma^*\ (P(s) \Rightarrow\ \ P(a.s))$,

atunci $\forall\ s \in \Sigma^*\ P(s)$

Aceasta este o instantă a **inductiei structurale**, unde un set de axiome defineste modul în care sunt construite obiectele dintr-o multime si cum un principiu inductiv utilizează repetat pasul de constructie pentru a acoperi întregul domeniu. Aici, "." este **constructorul** pentru domeniul de siruri, întocmai cum "+1" este constructorul pentru numerele naturale.

De notat că numerele nu apar nicăieri în aceste axiome. Se pot face demonstratii gândind numai la obiectele în discutie. Să definim acum o functie care inversează un sir si să dovedim că ea lucrează corect.

**Axioma 4.4** (a inversării):

$r(\lambda) = \lambda$

$\forall\ a \in \Sigma\ \forall\ s \in \Sigma^*\ r(a.s) = r(s).a$

Ne-ar plăcea să spunem ceva de genul "pentru fiecare sir $s$, $r(s)$ este inversul lui". Pentru a face aceasta o teoremă precisă, este necesar un mod oarecare, independent, nerecursiv de a spune ce se întelege prin inversare! Sunt moduri diverse de a face aceasta dintre care cel mai usor este a uza de avantajul scrierii cu "puncte puncte":

**Teorema 4.1**: $\forall\ s \in \Sigma^*$ fie $s = a_1a_2\ldots a_n$; atunci $r(s) = a_n\ldots a_2a_1$.

**Demonstratie**: Demonstratia este prin inductie pe sirurile construite pe alfabetul $\Sigma$. Fie $P(s)$ propozitia conform căreia dacă $s = a_1a_2\ldots a_n$ atunci $r(s) = a_n\ldots a_2a_1$.

- Cazul de bază: demonstratia pentru propozitia $P(\lambda)$, care este adevărată prin definitie.
- Pasul inductiv: demonstrarea propozitiei $P(s) \Rightarrow P(a.s)$ pentru $\forall\ a \in \Sigma\ \forall\ s \in \Sigma^*$.
    1. Ipoteza inductivă declară că pentru un sir arbitrar $s$, dacă $s = a_1a_2\ldots a_n$ atunci $r(s) = a_n\ldots a_2a_1$.
    2. De demonstrat: pentru orice simbol $a$, $r(a.s) = a_n\ldots a_2a_1a$.
    3. Prin axioma inversării,

$$r(a.s) = r(s).a = a_n\ldots a_2a_1a.$$

Asadar, prin principiul inductiei pentru siruri, pentru orice sir $s$, $r(s)$ îl inversează.

□

Alternativ, s-ar fi putut demonstra teorema prin inductie pe lungimea sirului. Este un bun exercitiu de a face detaliat această demonstratie si de a o compara cu metoda de mai sus.

**Inductia pe arbori binari**

Arborii sunt structuri de date fundamentale în stiinta calculatoarelor, care sunt substratul unor implementări eficiente în multe domenii care includ bazele de date, grafica, compilatoarele, editoarele de texte si de imagini, optimizarea, jocurile si multe altele. Arborii sunt utilizati si pentru a reprezenta expresii în limbajele formale. Aici se studiază forma lor primară, de bază: **arborii binari**. Arborii binari includ liste (ca în limbajele Lisp sau Scheme), care au `nil` ca frunza lor cea mai din dreapta.

În teoria arborilor binari se începe uzula cu **atomii**, care sunt arbori fără ramificatii. **A** este multimea de atomi finită sau infinită. Arborii (**T**) se construiesc folosind operatorul • (constructor) (în practică, orice obiect poate fi un atom atât timp cât el este distinct si unic). Se va trata numai cazul arborilor binari completi, arbori în care fiecare nod are zero sau doi descendenti.

**Axioma 4.5** (**Arbori binari completi**):

Fiecare atom este un arbore: $\forall\ a \in \mathbf{A}\ [a \in \mathbf{T}]$

Prin compunerea a doi arbori se obtine un arbore: $\forall\ t_1, t_2 \in \mathbf{T}\ [t_1 \bullet t_2 \in \mathbf{T}]$

Principiul inductiei pentru arbori spune că dacă $P$ este adevărată pentru toti atomii si dacă adevărul lui $P$ pentru orice doi arbori implică adevărul lui $P$ pentru compusul lor, atunci $P$ este adevărată pentru toti arborii.

**Axioma 4.6** (**Inductia pe arbori binari completi**):

Pentru orice proprietate $P$,

dacă $\forall\ a \in \mathbf{A}\ P(a)$

si $\forall\ t_1, t_2 \in \mathbf{T}\ [P(t_1) \wedge P(t_2) \Rightarrow P(t_1 \bullet t_2)]$

atunci $\forall\ t \in \mathbf{T}\ P(t)$

Pe arbori pot fi definite multe predicate si functii utile între care:

- *leaf*($a$, $t$) este adevărată dacă si numai dacă atomul $a$ este frunză (leaf) a arborelui $t$

- $t_1 \prec t_2$ este adevărată dacă si numai dacă arborele $t_1$ este un subarbore propriu al arborelui $t_2$
- *count*($t$) notează numărul de frunze ale arborelui $t$
- *depth*($t$) notează **adâncimea** arborelui $t$; adâncimea unui atom este zero
- *balanced*($t$) este adevărat dacă si numai dacă $t$ este un arbore binar balansat, echilibrat.

Se defineste imediat *leaf*, celelalte rămân ca exercitiu:

**Axioma 4.7** (**Leaf**):

$$\forall\ a \in \mathbf{A}\ \ \forall\ t \in \mathbf{T}\ \ leaf(t, a) \Leftrightarrow\ \ t = a$$

$$\forall\ a \in \mathbf{A}\ \ \forall\ t_1, t_2 \in \mathbf{T}\ \ leaf(a, t_1 \bullet t_2) \Leftrightarrow\ \ leaf(a, t_1) \vee leaf(a, t_2)$$

Nu este usor a *demonstra* că definitiile unor astfel de functii de bază sunt corecte, deoarece "specificarea" functiei este dificil a fi scrisă în vreo formă care să fie mai simplă decât definitia însăsi. Să vedem o functie ceva mai putin simplă: functia *maxleaf*($t$) returnează frunza cea mai mare a unui arbore $t$, unde atomii sunt constrânsi să fie numere.

**Axioma 4.8** (**Maxleaf**):

$$\forall\ a \in \mathbf{A}\ \ maxleaf(a) = a$$

$$\forall\ t_1, t_2 \in \mathbf{T}\ \ maxleaf(t_1 \bullet t_2) = max(maxleaf(t_1), maxleaf(t_2))$$

Functia *maxleaf* este "corectă" dacă satisface două proprietăti: prima, *maxleaf*($t$) trebuie să fie mai mare sau egală cu oricare frunză a lui $t$; a doua (*adesea uitată*), *maxleaf*($t$) trebuie să fie o frunză în $t$!

Să dovedim mai întâi proprietatea a doua:

**Teorema 4.2**: Pentru fiecare arbore, $t$, *maxleaf*($t$) este o frunză în $t$.

**Demonstratie**: Demonstratia este prin inductie pe arborii binari si pe atomii **A**. Fie $P(t)$ propozitia *leaf*(*maxleaf*($t$), $t$).

- Cazul de bază: se demonstrează că $\forall\ a \in \mathbf{A}\ \ P(a)$.

$P(a)$ este propozitia *leaf*(*maxleaf*($a$), $a$), care este echivalentă prin substitutie propozitiei *leaf*($a$, $a$) care este adevărată prin definitie.

- Pasul inductiv: de dovedit $P(t_1) \wedge P(t_2) \Rightarrow\ \ P(t_1 \bullet t_2)$ pentru orice $t_1, t_2 \in \mathbf{T}$.
  1. Ipoteza inductivă spune că $leaf(maxleaf(t_1), t_1) \wedge leaf(maxleaf(t_2), t_2)$
  2. De demonstrat: $leaf(maxleaf(t_1 \bullet t_2), t_1 \bullet t_2)$
  3. Prin definitia de mai devreme, $maxleaf(t_1 \bullet t_2) = max(maxleaf(t_1), maxleaf(t_2))$
  4. Deoarece $\forall\ x, y\ \ [(max(x, y) = x) \vee (max(x, y) = y)]$, avem $(maxleaf(t_1 \bullet t_2) = maxleaf(t_1)) \vee (maxleaf(t_1 \bullet t_2) = maxleaf(t_2))$.
  5. Substituind în ipoteza inductivă, se obtine
     $$leaf(maxleaf(t_1 \bullet t_2), t_1) \vee leaf(maxleaf(t_1 \bullet t_2), t_2)$$
  6. Asadar, prin definitia functiei *leaf*,
     $$leaf(maxleaf(t_1 \bullet t_2), t_1 \bullet t_2).$$

Prin urmare, pe baza principiului inductiei binare, pentru orice arbore $t$, *maxleaf*($t$) este o frunză (leaf) a lui $t$.

□

Cealaltă parte a verificării este următoarea (demonstratie lăsată ca exercitiu):

**Teorema 4.3**: Pentru orice arbore *t*, *maxleaf*(*t*) este mai mare decât sau egală cu orice frunză a arborelui *t*.

Inductia pe arbori apare foarte naturală. Se poate face o demonstratie similară utilizând inductia pe numere naturale? Cu certitudine se pot face demonstratii diverse prin inductie după *adâncimea* unui arbore. $P(n)$ ar putea afirma că toti arborii de profunzime $n$ satisfac o anume proprietate $Q$. Din nefericire, pasul inductiv pentru o inductie *simplă* ar arăta astfel:

Stiind că: toti arborii $t$ de adâncime $n$ satisfac $Q(t)$

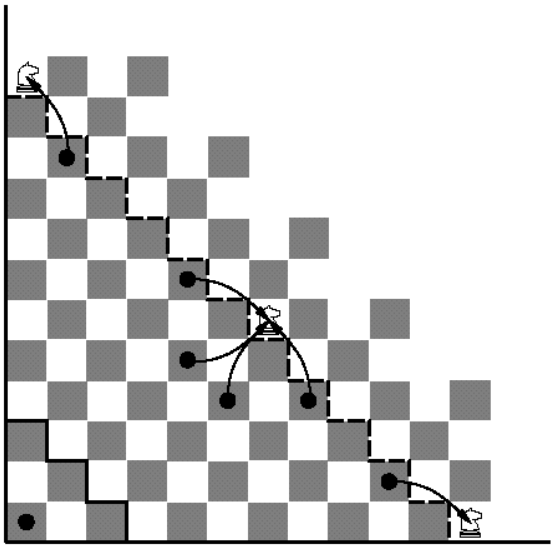
Să se demonstreze că: toti arborii $t$ de adâncime $n + 1$ satisfac $Q(t)$

Acest fapt este uzual imposibil: pentru un arbore de adâncime $n + 1$, un subarbore are adâncimea $n$, dar nu în mod necesar ceilalti subarbori au aceeasi adâncime $n$. Inductia tare pe adâncimea arborilor lucrează: ea poate fi utilizată oricând în locul inductiei pe arbori.

De observat că formalizarea de mai sus relativ la arbori descrie numai arbori întregi. Cu toate acestea, ea poate fi usor generalizată la descrierea arborilor binari care nu sunt necesarmente integrali (întregi), adică, acolo unde fiecare nod poate avea 0, 1 sau 2 descendenti. Este usor a pune detaliile la locul cuvenit, asa încât le lăsăm în seama cititorului.

**Inductia pe perechi de numere naturale**

Adesea este necesar a dovedi proprietăti pe **produsul cartezian** al unor multimi date. Produsul cartezian al multimilor **A** si **B** se scrie **A**x**B**. Este multimea tuturor perechilor ($a$, $b$) cu $a \in$ **A** si $b \in$ **B**. De pildă, multimea **N**x**N** este multimea tuturor perechilor de numere naturale. Astfel de multimi apar când se demonstrează proprietăti ale unor functii cu două argumente, când se demonstrează fapte despre toate punctele unei grile etc.



Turul calului cu indicarea pătratului "caz-de-bază", a miscărilor regulamentare ale calului si a "pasului inductiv"

Un exemplu: deplasarea calului pe o tablă de sah. Se va demonstra că un cal care pleacă din pătratul (0, 0) poate vizita oricare alt pătrat situat într-un cadran nenegativ nelimitat. Figura de mai sus arată (în parte) tabla de sah infinită si ilustrează miscările pe care calul le poate face.

Pentru a demonstra acest rezultat, trebuie precizate unele fapte relativ la deplasarea calului. În particular, este necesară

**Axioma 4.9** (**Deplasarea calului**): Dacă pătratul $(x \pm 1, y \pm 2)$ sau pătratul $(x \pm 2, y \pm 1)$ poate fi atins de cal, atunci pătratul $(x, y)$ poate fi atins de cal.

Mai este necesar un principiu inductiv pe perechi de numere naturale. Ideea pentru demonstratia deplasării acoperitoare a calului este de a stabili o regiune care poate fi atinsă si apoi a arăta că orice pătrat adiacent acelei regiuni poate fi atins; asadar regiunea creste pentru a umple cadranul nemărginit. Sunt multe căi de a defini forma acestei regiuni; aici se va utiliza o regiune triunghiulară ca în figura de mai sus.

Principiul inductiv constă, informal, în faptul că dacă adevărul propozitiei $P$ pentru orice pereche $(x', y')$ situată în regiunea "situată imediat sub" $(x, y)$ implică adevărul propozitiei $P$ pentru $(x, y)$, atunci $P$ este adevărată pentru toate perechile $(x, y)$. De observat că acesta este un principiu inductiv tare.

**Axioma 4.10** (**Inductia tare** (**pentru perechi**)):

Pentru o proprietate $P$,
dacă $\forall\ x, y \in \mathbf{N}$
$[\forall\ x', y' \in \mathbf{N}\ \ (x' + y') < (x + y) \Rightarrow\ \ P(x', y')] \Rightarrow\ \ P(x, y)$
atunci $\forall\ x, y \in \mathbf{N}\ \ P(x, y)$.

Dar unde este cazul de bază? Este prezent, dar este ascuns observatiei imediate. Când $(x, y) = (0, 0)$, conditia $[\forall\ x', y' \in \mathbf{N}\ \ (x' + y') < (x + y) \Rightarrow\ \ P(x', y')]$ este adevărată prin lipsa de sens dată fiind absenta oricăror astfel de perechi. Asadar, $P(0, 0)$ este o parte a premiselor care trebuie demonstrată. Mai general, "cazul de bază" este multimea de perechi $(x, y)$ pentru care ipoteza inductivă nu este suficientă pentru a da o demonstratie.

Acum sunt toate conditiile de a demonstra:

**Teorema 4.4**: $\forall\ x, y \in \mathbf{N}$, pătratul de coordonate $(x, y)$ poate fi atins de un cal pornind din (0, 0).

**Demonstratie**: Demonstratia este prin inductie tare pe perechi de numere naturale. Fie $P(x, y)$ propozitia conform căreia pătratul $(x, y)$ poate fi atins de un cal care porneste din (0, 0).

- Cazul de bază: propozitiile $P(0, 0)$, $P(0, 1)$, $P(0, 2)$, $P(1, 0)$, $P(1, 1)$, $P(2, 0)$, pentru care $x + y \leq 2$, trebuie dovedite separat. Fiecare dintre acestea pot fi stabilite prin aplicarea potrivită a axiomei miscării calului.
- Pasul inductiv: Se demonstrează că pentru orice $(x, y)$ astfel ca $x + y > 2$ $[\forall\ x', y' \in \mathbf{N}\ \ (x' + y') < (x + y) \Rightarrow\ \ P(x', y')] \Rightarrow\ \ P(x, y)$.

1. Ipoteza inductivă arată că pentru orice $x', y' \in \mathbf{N}$ astfel încât $(x' + y') < (x + y)$, pătratul $(x', y')$ poate fi atins din (0, 0)
2. Toate pătratele $(x', y') = (x - 2, y \pm 1)$ si $(x', y') = (x \pm 1, y - 2)$ satisfac conditia $(x' + y') < (x + y)$.
3. Pentru orice $x, y \in \mathbf{N}$ astfel încât $x + y > 2$, cel putin unul din aceste pătrate este pe tablă, adică satisface $x', y' \in \mathbf{N}$ (demonstratie caz cu caz).
4. Prin urmare, prin axioma deplasării calului, pătratul $(x, y)$ poate fi atins din (0, 0).

Asadar, prin principiul inductiei tari pentru perechi de numere, fiecare pătrat în cadranul pozitiv nemărginit poate fi atins de cal pornind din (0, 0).

□

Demonstratia poate fi făcută si prin inductia tare pe numere naturale utilizând $n = x + y$ ca variabilă inductivă. Care din cele două căi este mai elegantă este o chestiune de gust; dar rămâne important uzul notiunii potrivite de "mai mic" pentru perechi de numere naturale. Pentru unele demonstratii "mai mic" poate fi definit ca "cel putin un număr din pereche este mai mic si celălalt nu este mai mare", care dă regiuni rectangulare care, pas cu pas, umplu cadranul. În problema turului făcut de cal, totusi, unele din miscările necesare violează această ordonare.

**Inducția bine fundamentată**

Privind toate principiile inductive examinate până acum, o temă recurentă rămâne: din proprietătile elementelor "mai mici" derivă prin demonstratii proprietăti pentru elementele "mai mari". $n$ este mai mic decât $n + 1$; $s$ este mai mic decât $a.s$; $t_1$ si $t_2$ sunt mai mici decât $t_1 \bullet t_2$ si asa mai departe.

Principiul inductiei tari pentru perechi, discutată în sectiunea anterioară, dă o deschidere spre formalizarea acestei idei într-un principiu general inductiv. Se furnizează simplu o notiune generalizată de "mai mic decât" în loc de a utiliza semnul "<". S-a notat această relatie cu "$\prec$", care se presupune a fi definită pe o multime, care va fi aceea, **X** si care interesează (numere naturale, multimi, arbori, perechi, siruri, liste, orare pentru liniile aeriene etc.). Pentru ca inductia să lucreze se cere ca "$\prec$" să aibă proprietatea de bună fundamentare:

**Definitia 4.1** (**buna fundamentare**): O relatie $\prec$ pe **X** este bine fundamentată dacă nu poate exista o secventă descrescătoare si infinită de elemente din **X**, elemente în relatia $\prec$.

Fiind dat acest fapt, se poate formula principiul inductiei bine fundamentate, principiu fată de care toate principiile discutate până acum sunt cazuri particulare:

**Axioma 4.11** (**Inductia bine fundamentată**):

Pentru orice proprietate $P$ si pentru orice relatia pe **X**, $\prec$, bine fundamentată,
dacă $\forall\, x \in \mathbf{X}\ [[\forall\, y \in \mathbf{X}\ y \prec x \Rightarrow P(y)] \Rightarrow P(x)]$
atunci $\forall\, x \in \mathbf{X}\ P(x)$

Ca si în cazul inductiei pe perechi, principiul inductiei bine fundamentate include cerinta de a stabili cazul-de-bază – adică demonstrarea propozitiei $P(x)$ în mod independent pentru toti acei $x$-i unde ipoteza inductivă nu este suficientă.
Proprietatea de bună fundamentare este usor de observat în toate cazurile tratate mai devreme. Există de asemenea un echivalent generalizat al bunei ordonări:
**Definitia 4.2** (**buna ordonare**): O multime **X** este bine ordonată de relatia $\prec$ dacă si numai dacă fiecare submultime nevidă a lui $X$ are cel putin un element minimal în raport cu $\prec$.
Poate fi dovedită acum următoarea teoremă foarte generală:
**Teorema 4.5**: O relatie $\prec$ pe **X** este bine fundamentată dacă si numai dacă **X** este bine ordonată de $\prec$.
Desi aceasta apare ca foarte abstractă si, poate, inutilă, ea este de fapt utilizată tot timpul de programatorii care scriu functii recursive care execută lucruri complexe cu argumentele lor. Se consideră următorul schelet (skeleton) recursiv:

$f(x)$ = if $B(x)$ then $k$ else $f(g(x))$

Aceasta se va încheia dacă si numai dacă $g(x) \prec x$ pentru o bună ordonare a lui **X** cu elementul(ele) minimal(e) satifăcând $B(x)$. Astfel, programatorul trebuie să fie sigur că aplicarea repetată a lui $g$ nu poate genera o secventă infinită de valori care nu satisfac $B$.
Uneori, "mai mic" poate fi surprinzător de *non*-evident. Se consideră următoarea functie pe numerele naturale:

$f(0) = 1; f(1) = 1$

dacă $n > 1$ este par atunci $f(n) = f(n/2)$, altminteri $f(n) = f(3n + 1)$.

Conjectura lui Collatz afirmă că $\forall\ n \in \mathbf{N}\ \ f(n) = 1$. Cititorul ar putea dori să verifice acest fapt pentru valori diverse ale lui $n$. Nu este cunoscută însă vreo demonstratie.

# Lecția 5

## Divide-et-impera si mergesort

Una dintre operatiunile fundamentale executate de calculatoare este sortarea. Sortarea înseamnă punerea în ordine ascendentă a *n* obiecte care provin dintr-o multime total ordonată. Aceste obiecte pot fi cuvinte ale unei limbi care trebuie ordonate alfabetic: fiind date *n* cuvinte, sortarea necesită asezarea lor în ordine alfabetică. Pentru a evita detaliile obositoare se va presupune mai departe că cele *n* obiecte sunt distincte.

Cum se elaborează o procedură generală pentru a face această operatie? După nu prea multă gândire, vor apărea idei cam de genul următoarelor:

**Metoda 1**. Printr-o scanare a listei se poate găsi cel mai mare articol si el poate fi plasat la finalul listei de iesire, listei rezultante. Apoi, prin scanarea articolelor rămase se poate găsi al doilea ca mărime si asa în continuare. (Această metodă este cunoscută ca **sortarea prin selectie**).

**Metoda 2**. Se ia primul articol (item) si se depune în lista de iesire. Se ia al doilea articol si se inserează în ordinea corectă fată de primul articol. Se continuă în această manieră, de fiecare dată inserând următorul articol în pozitia corectă printre articolele inserate anterior – această pozitie poate fi găsită printr-o scanare liniară printre aceste articole. (Această metodă este cunoscută ca **sortare prin insertie**).

Cât de bune sunt aceste metode? Este destul de usor de văzut că ambele sunt *corecte*, adică ambele produc o versiune sortată a listei initiale. (Ca exercitiu, cititorul poate afirma acest fapt formal si-l poate demonstra pentru fiecare metodă prin inductie după *n*). Dar cât de eficiente sunt aceste metode? Mai întâi metoda prin selectie. Este usor de văzut că prima scanare necesită exact $n - 1$ comparatii de articole pentru a stabili cel mai mare element; similar, a doua scanare necesită $n - 2$ comparatii s.a.m.d. Numărul total de comparatii necesare este, prin urmare

$$(n-1) + (n-2) + \ldots + 2 + 1 = \sum_{i=1}^{n-1} i = \frac{1}{2} n(n-1)$$

S-a folosit aici formula pentru suma primelor $(n - 1)$ numere naturale discutată într-o sectiune anterioară.

Dar metoda prin insertie? Pentru inserarea celui de al doilea articol este necesară o comparatie; pentru a insera al treilea element ar fi necesare (în cel mai rău caz) două comparatii; în general pentru articolul *i* ar fi necesare (în cel mai rău caz) $i - 1$ comparatii. Astfel, numărul de comparatii ar fi în cel mai rău caz

$$\sum_{i=1}^{n-1} i = \frac{1}{2} n(n-1)$$

la fel ca în cazul anterior.

Astfel numărul de comparatii executate prin cele două metode este în cazul cel mai defavorabil $(1/2)n^2 - (1/2)n \approx (1/2)n^2$ pentru $n$ mare. Deoarece comparatiile sunt partea principală a efortului de calcul se poate gândi despre $n^2$ ca o măsură a eficientei algoritmilor (ca o functie de $n$, numărul articolelor de sortat). Numărând comparatiile, si nu toate operatiile-masină de nivel inferior, se obtine o măsură destul de exactă a eficientei care nu depinde de detaliile masinii, de limbajul de programare sau de implementarea particulară. Pentru acelasi motiv, este potrivit a renunta la factorii constanti (ca 1/2, de pildă) ca si la termenii de ordin inferior (cum este $(1/2)n$). Se spune că timpul de executie al acestor metode este "$O(n^2)$" (si se citeste de "ordinul $n^2$" sau "$O$-mare de $n^2$") pentru a indica faptul că $n^2$ este aici o măsură primară. (Despre notatia $O$ se va vorbi mai mult mai departe).

Cât de util este un algoritm de sortare de $O(n^2)$? De exemplu, dacă e necesar a sorta 2,1 miloane de conturi de brokeraj prin numărul de cont, uzând de un PC la 1 GHz? Este optimistă presupunerea că în medie se face o comparatie la fiecare 10 cicluri ale unitătii centrale (cu includerea încărcării, stocării si manipulării listelor). Asadar, se pot face 100 de milioane de comparatii pe secundă. Un algoritm care necesită $(1/2)n^2$ comparatii consumă peste 6 ore. Pare a fi cam mult.

Se poate utiliza un algoritm de sortare mai rapid decât $O(n^2)$? Se poate prezuma că nu se poate atinge $O(n)$ pentru că numai parcurgerea tuturor celor $n$ articole consumă $n$ pasi. Rezultă deductiv că nu se poate mai mult decât a fi aproape de această comportare ideală prin utilzarea inteligentă a recurentei. Mai jos este dat un algoritm de sortare numit "Mergesort", care necesită pentru a sorta $n$ articole numai $O(n\log_2 n)$ comparatii. Pentru problema discutată în paragraful precedent acest algoritm reduce timpul de calcul la 0,4 secunde.

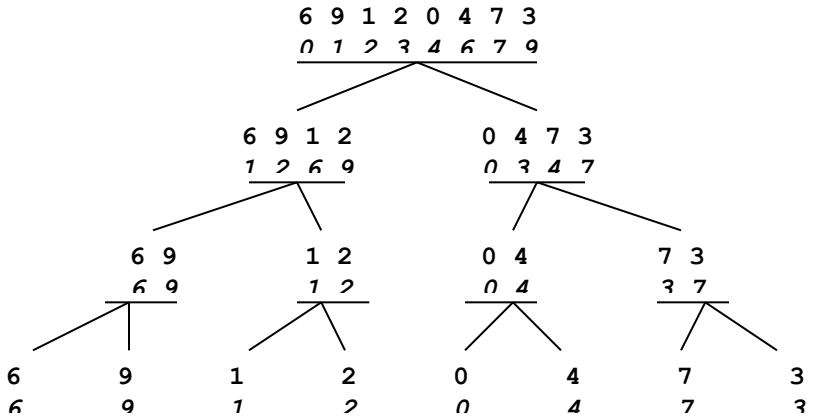
Utilizarea recurenței este unul din exemplele cele mai simple preluat din tehnicile puternice din categoria "*divide-et-impera*". În secțiunile următoare vor fi date si alte exemple. Ideea este a diviza intrarea în două sau mai multe bucăti (mai mici, desigur) sau în "subprobleme" care, separat, pot fi "cucerite" apoi printr-o aplicatie recursivă a aceluiasi algoritm. În cele din urmă, soluțiile pentru subprobleme trebuie alipite, puse laolaltă adecvat pentru a forma solutia problemei initiale. Algoritmii *divide-et-impera* variază mult în rafinamentul operatiilor "divide" si "lipire-impera".

Pentru problema sortării, un mod natural de divizare a datelor de intrare constă în a împărti lista celor $n$ articole în două liste fiecare de dimensiunea $n/2$; operatia se face usor prin luarea primelor $n/2$ articole si a celor $n/2$ articole rămase. (Pentru simplitate, pentru a evita numeroase rotunjiri în sus si în jos, se presupune că $n$ este o putere a lui 2 astfel încât fiecare divizare devine imediat posibilă. Această presupunere nu este necesară în practică). Apoi, recursiv, se sortează fiecare listă mai scurtă. În final, se pun laolaltă sublistele sortate pentru

a obtine o versiune sortată a listei initiale, originale. Algoritmul este scris mai detaliat imediat:

Algoritmul Mergesort(*S*)
{*S* este o listă de articole dintr-o multime total ordonată, *n* este o putere a lui 2}
if |*S*| = 1 then return *S*
else
divide *S* into *T* (primele *n*/2 articole) and *U* (*n*/2 articole rămase)
*T'* := Mergesort(*T*)
*U'* := Mergesort(*U*)
*S'* := Merge(*T'* ,*U'*)
return *S'*

Algoritmul nu este deplin specificat deoarece nu s-a definit procedura "Merge". Cu toate acestea, ideea este evidentă: pentru a pune împreună două liste sortate, *T* si *U*, într-o singură listă sortată se face o scanare pe ambele liste deodată, "în paralel", plasând la fiecare pas articolul cel mai mic din cele două în examinare în lista de iesire. Astfel, se începe cu primele articole, $t_1$ si $u_1$, si se plasează cel mai mic dintre ele (fie acesta $u_1$) ca prim articol în lista-rezultat. Apoi se compară $t_1$ cu $u_2$ si se asează din nou cel mai mic (fie acela $t_1$) ca al doilea articol în lista-rezultat. Se compară apoi $t_2$ cu $u_2$ si se plasează cel mai mic din ele ca al treilea articol în lista de iesire si tot asa, până când una din liste este epuizată. La acest moment operația se încheie prin adăugarea elementelor remanente în lista neterminată, la finalul listei de iesire. Procedura necesită cel mult o comparatie pentru fiecare element al listei asamblate (merged) (cu exceptia ultimului), astfel că numărul total de comparatii necesare pentru a asambla cele două liste de lungime combinată *n* este de cel mult *n* – 1.




Arborele apelurilor recursive la Mergesort cu listele de intrare si de iesire

Scrierea rutinei Merge rămâne ca exercitiu pentru cititor. Se dă totusi aici **specificarea intrare-iesire**, adică proprietătile cerute a fi deținute de iesiri fiind date intrări adecvate. Demonstratia este lăsată, din nou, ca exercitiu.

**Lema 5.1** (**Merge**): Dacă A si B sunt liste disjuncte sortate, atunci Merge(A, B) este o permutare sortată a reuniunii $A \cup B$.

Acum se va demonstra că algoritmul Mergesort este corect. De observat că se poate face asta *chiar dacă subrutina* Merge *nu a fost scrisă*, deoarece tot ce este necesar este *a specifica* Merge pentru a desăvârşi demonstratia pentru Mergesort. Similar, pentru a analiza timpul executiei lui Mergesort este necesară numai cunoasterea timpului de executie al lui Merge (care *nu* decurge din specificarea ei).

**Teorema 5.1**: Algoritmul Mergesort, când este alimentat ca o listă $S$ de $n$ articole distincte ($n$ o putere a lui doi) produce o versiune sortată a lui $S$.

**Demonstratie**: Mai întâi teorema trebuie enuntată ceva mai precis. Ce se întelege prin "versiunea sortată a lui $S$"? Prin aceasta se întelege că lista de iesire $S'$ are următoarele două proprietăti:

(i) Lista $S'$ este o permutarea a listei $S$, adică ea constă exact din aceleasi elemente, articole, posibil în altă ordine.

(ii) Lista $S'$ este sortată, adică dacă $S' = [s_1', s_2', \ldots, s_n']$ atunci $s_1' < s_2' < \ldots < s_n'$.

Se dovedeste acum prin inductie pe lungimea $n$ a listei de intrare $S$ că lista de iesire $S'$ este o versiune sortată a lui $S$. Cazul de bază ($n = 1$) este facil si imediat: aici $S' = S$, care este desigur propria sa versiune sortată deoarece contine un singur element.

Acum se ia în consideratie cazul $n > 1$ ($n$ putere a lui 2). În acest caz, din definitia algoritmului se vede că iesirea $S'$ este exact Merge($T'$, $U'$) cu $T'$ = Mergesort($T$) si $U'$ = Mergesort($U$) si $T$, $U$ respectiv prima si a doua jumătate a lui $S$. Deoarece $T$ si $U$ sunt liste de lungime $n/2$, se poate aplica ipoteza inductivă pentru a deduce că $T'$ si $U'$ sunt versiuni sortate ale lui $T$ si $U$, respectiv, adică atât $T'$ cât si $U'$ satisfac proprietătile (i) si (ii) de mai sus.

Pentru a încheia demonstratia, este necesar a se apela la proprietătile rutinei Merge. Deoarece $T'$ si $U'$ sunt liste disjuncte sortate, lema Merge spune că $S'$ este o permutarea sortată a reuniunii $T' \cup U'$. Acum se apelează la o proprietate generală a permutărilor: pentru orice două liste disjuncte $T$, $U$, dacă $T'$ este o permutare a lui $T$ si $U'$ este o permutarea a lui $U$, atunci $T' \cup U'$ este o permutarea a lui $T \cup U$. Acum $T' \cup U'$ este prin definitie o permutare a lui $S$. asadar $S'$ este o permutare sortată a lui $S$.

□

Să revenim la eficienta algoritmului Mergesort. Ca si mai devreme, se numără numai comparatiile cerute pentru a sorta cele $n$ articole. Se poate folosi ca argument informal arborele recursiv din figura de mai sus. Primul nivel (rădăcina) este o problemă de dimensiune $n$, al doilea nivel constă în două probleme de dimensiune $n/2$, al treilea nivel în patru probleme de dimensiune $n/4$ s.a.m.d. Numărul total de niveluri este $\log n + 1$, cu partea cea mai de jos (frunzele) $n$ probleme de dimensiune 1. Câte comparatii sunt executate la

fiecare nivel? La nivelul rădăcinii sunt executate în cel mai nefericit caz $n - 1$ comparatii. La nivelul secund se execută în cel mai rău caz $(n/2) - 1$ comparatii pentru fiecare din cele două subprobleme, pentru un total de $n - 2$. La nivelul 3 se execută $4((n/4) - 1) = n - 4$ comparatii s.a.m.d. Astfel la fiecare nivel se execută cel mult $n$ comparatii (cu exceptia nivelului frunzelor unde nu se face nici o comparatie). Deoarece sunt $\log_2 n$ niveluri (nivelurile care nu sunt frunze), numărul total de comparatii este de cel mult $n\log n$.

Cu o numărare mai îngrijită se poate stabili următoarea margine superioară:

**Teoreme 5.2**: Numărul maxim de comparatii, $C(n)$, efectuat de algoritmul Mergesort pentru o listă de intrare de lungime $n$ (o putere a lui 2) satisface relatia $C(n) \leq n\log n - n + 1$ (logaritmul este în baza 2 ori de câte ori nu se specifică altfel).

**Demonstratie**: Teorema se demonstrează prin inductie tare după $n$. Cazul de bază ($n = 1$) este usor de dovedit: aici Mergesort nu execută nici o comparatie (simplu, el serveşte la ieşire lista de intrare) si valoarea expresiei $n\log n - n + 1$ când $n = 1$ este zero. Asadar, cazul de bază se verifică.

Urmează cazul $n > 1$ ($n$ o putere a lui 2). Mergesort execută două apeluri recursive, fiecare pe o listă de lungime $n/2$ si apelează Merge pe două liste cu lungimile însumate $n$. Asadar

$$C(n) \leq 2C(n/2) + (n - 1)$$

Acesta est pasul crucial: atentie la întelegerea lui! Primul termen din partea dreaptă vine din cele două apeluri recursive (fiecare, prin definitia lui $C(.)$, necesită cel mult $C(n/2)$ comparatii); al doilea termen vine din pasul Merge (care reclamă cel mult $n - 1$ comparatii).

Acum se continuă demonstratia cu putină algebră si cu aplicarea ipotezei inductive:

$$\begin{aligned} C(n) &\leq 2C(n/2) + (n - 1) \leq \\ &\leq 2[(n/2)\log(n/2) - (n/2) + 1] + n - 1 = \\ &= n(\log n - 1) + 1 = n\log n - n + 1 \end{aligned}$$

în linia a doua s-a utilizat ipoteza inductivă (tare). Teorema este astfel demonstrată prin inductie.

□

# Lecția 6

## Expresii booleene si functii booleene

Această lectie revine la subiectul logicii propozitionale. Dacă în lectia 1 s-a studiat acest subiect ca o cale de întelegere potrivită a demonstratiilor si rationamentelor, acum acelasi subiect se reia din perspectiva calculatoarelor si a dispozitivelor numerice. Ulterior se vor găsi căi de a manipula expresiile logice în mod algoritmic pentru a rezolva automat probleme dificile. În cursul acestei întreprinderi, vor fi trecute în revistă unele notiuni fundamentale despre complexitate. Se va prezenta si un program-joc destul de bine cunoscut sub numele **Minesweeper**.

Exact cum aritmetica se ocupă cu toată matematica izvorîtă din operatiile cu numere, studiul functiilor booleene se ocupă cu toată matematica rezultată din operatii cu **valorile booleene** *adevărat* si *fals*, care se vor nota mai departe cu $T$ si $F$ (1 si 0 sunt de asemenea larg utilizate). În ciuda faptului că sunt numai două valori, multă matematică interesantă se dezvoltă plecând tocmai de la aceste două valori.
Începem printr-o definitie constructivă formală a multimii de **expresii booleene** (sau formule boolene, sau expresii ale logicii propozitiilor, sau propozitii propozitionale). De notat similaritatea cu definitia arborilor binari etc. Treaba este ceva mai complexă aici deoarece multimea este mult mai complexă.
Fie **X** o multime de **simboluri propozitionale** $\{X_1, \ldots, X_n\}$ (denumite si variabile booleene) si **B** multimea de expresii booleene pe **X** (de observat că mai jos expresiile însesi sunt subliniate pentru a evita confuzia cu notatia logică de care sunt înconjurate. Nu se va continua prea departe cu această scriere dar cititorul o poate utiliza mental dacă se consideră în pericol de confuzie).
**Definitia 6.1** (**Expresii booleene**):

$\underline{T} \in \mathbf{B}$ si $\underline{F} \in \mathbf{B}$
$\forall\ X \in \mathbf{X}\ [\underline{X} \in \mathbf{B}]$
$\forall\ B \in \mathbf{B}\ [\underline{\neg B} \in \mathbf{B}]$
$\forall\ B_1, B_2 \in \mathbf{B}\ [\underline{B_1} \wedge \underline{B_2} \in \mathbf{B}]$
$\forall\ B_1, B_2 \in \mathbf{B}\ [\underline{B_1} \vee \underline{B_2} \in \mathbf{B}]$
$\forall\ B_1, B_2 \in \mathbf{B}\ [\underline{B_1} \Rightarrow \underline{B_2} \in \mathbf{B}]$
$\forall\ B_1, B_2 \in \mathbf{B}\ [\underline{B_1} \Leftrightarrow \underline{B_2} \in \mathbf{B}]$

Pentru a demonstra unele lucruri relativ la expresiile booleene va fi necesar următorul principiu inductiv:
**Axioma 6.1** (**Inductia pe expresii booleene**):

Pentru orice proprietate $P$, dacă
$P(T)$ si
$P(F)$ si
$\forall\ X\in\ \mathbf{X}\ P(X)$ si
$\forall\ B\in\ \mathbf{B}\ [P(B)\Rightarrow\ P(\neg B)]$ si
$\forall\ B_1, B_2\in\ \mathbf{B}\ [P(B_1)\wedge\ P(B_2)\Rightarrow\ P(B_1\wedge\ B_2)]$ si
$\forall\ B_1, B_2\in\ \mathbf{B}\ [P(B_1)\wedge\ P(B_2)\Rightarrow\ P(B_1\vee\ B_2)]$ si
$\forall\ B_1, B_2\in\ \mathbf{B}\ [P(B_1)\wedge\ P(B_2)\Rightarrow\ P(B_1\Rightarrow\ B_2)]$ si
$\forall\ B_1, B_2\in\ \mathbf{B}\ [P(B_1)\wedge\ P(B_2)\Rightarrow\ P(B_1\Leftrightarrow\ B_2)]$
atunci
$\forall\ B\in\ \mathbf{B}\ P(B)$.

| $P$ | $Q$ | $\neg P$ | $P\wedge\ Q$ | $P\vee\ Q$ | $P\Rightarrow\ Q$ | $P\Leftrightarrow\ Q$ |
|---|---|---|---|---|---|---|
| $F$ | $F$ | $T$ | $F$ | $F$ | $T$ | $T$ |
| $F$ | $T$ | $T$ | $F$ | $T$ | $T$ | $F$ |
| $T$ | $F$ | $F$ | $F$ | $T$ | $F$ | $F$ |
| $T$ | $T$ | $F$ | $T$ | $T$ | $T$ | $T$ |

Tabelul de adevăr pentru operatorii booleni

Terminologie utilă: o expresie de forma $B_1\wedge\ B_2$ este denumită **conjuncție**; $B_1$ si $B_2$ se numesc **conjuncți**. O expresie de forma $B_1\vee\ B_2$ se numeste **disjuncție**; $B_1$ si $B_2$ se numesc **disjuncți**.
Mai mult despre "sintaxa" expresiilor booleene. Ce înseamnă asta? Se poate considera o expresie booleană ca fiind o reprezentare a unei **funcții booleene**, în acelasi mod în care o expresie aritmetică ca $x + y$ reprezintă functia de adunare (de observat că sunt multe expresii aritmetice diferite care reprezintă aceeasi functie aritmetică. De exemplu, $(x + y + x – x)/1$ este aceeasi functie de $x$ si $y$ ca si $x + y$. De observat totodată că termenul "functie" este utilizat aici în sensul matematic cunoscut.).
O expresie booleană pe $\{X_1, \ldots, X_n\}$ are o **valoare de adevăr** pentru orice **atribuire** completă de valori $T/F$ pentru $\{X_1, \ldots, X_n\}$. (Atribuirile complete se mai numesc si **modele**, cum se va vedea mai târziu; o asignare pentru care o expresie are valoarea $T$ este numită un *model al acelei expresii*). Orice expresie booleană $B$ reprezintă prin urmare o functie care aplică $n$-upluri de valori booleene într-o valoare booleană:

$$B(X_1, \ldots, X_n):\{T, F\}^n \rightarrow \{T, F\}$$

Notatia $\{T, F\}^n$ înseamnă produsul cartezian cu $n$ factori identici cu multimea $\{T, F\}$. De exemplu, expresia booleană $X_1\wedge\ X_2$ pe multimea de simboluri $\{X_1, X_2\}$ aplică perechi de valori booleene într-o valoare booleană care este un "şi" al celor două intrări.
Regulile de evaluare a unei expresii în cazul unei asignări (atribuiri) sunt date de tabelele de adevăr pentru operatorii booleeni (vezi tabelul de mai sus). Astfel, fiecare simbol poate fi înlocuit cu valoarea sa potrivit asignării, apoi

expresia poate fi evaluată "de-jos-în-sus" (*bottom-up* – de la bază la vârf) ca si o expresie aritmetică. De exemplu, cu asignarea $\{A = T, B = F\}$, expresia $(A \wedge (A \Rightarrow B)) \Rightarrow B$ devine

$$[(T \wedge (T \Rightarrow F)) \Rightarrow F] = [(T \wedge F) \Rightarrow F] = [F \Rightarrow F] = T$$

Se poate da de asemenea o definitie recursivă de-sus-în-jos a valorii de adevăr a unei expresii. Fie $M$ o atribuire si fie $X_M$ valoarea lui $X$ corespunzătoare lui $M$. Atunci

$\forall X \in \mathbf{X}$ [eval$(X, M) = X_M$]

$\forall B \in \mathbf{B}$ [eval$(\neg B, M) = \neg$ (eval$(B, M)$)]

$\forall B_1, B_2 \in \mathbf{B}$ [eval$(B_1 \wedge B_2, M)$ = eval$(B_1, M) \wedge$ eval$(B_2, M)$]

etc.

De observat că în evaluarea de mai sus regulile utilizate în cazul operatorilor booleeni $\neg$, $\wedge$ si ceilalti, apar ca functii care operează pe valori booleene si nu ca operatori logici în propozitiile definitorii.

Fiind dată o definitie precisă a ceea ce expresiile semnifică, se poate defini acum următoarea notatie utilă:

**Definitia 6.2** (**Echivalența logică**):

Două expresii booleene pe aceeasi multime de variabile sunt **echivalente logic** dacă si numai dacă ele returnează aceleasi valori de adevăr pentru orice asignare de valori pentru variabile posibilă; adică ele reprezintă aceeasi functie booleană.

Pentru "echivalent logic cu" se va folosi simbolul $\equiv$. Câteva echivalențe evidente, toate verificabile recurgând la tabelele de adevăr:

$(A \wedge B) \equiv (B \wedge A)$ (comutativitate)

$(A \vee B) \equiv (B \vee A)$ (comutativitate)

$((A \wedge B) \wedge C) \equiv (A \wedge (B \wedge C))$ (asociativitate)

$((A \vee B) \vee C) \equiv (A \vee (B \vee C))$ (asociativitate)

$(A \Rightarrow B) \equiv (\neg A \vee B)$

$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \wedge (B \Rightarrow A))$

$\neg(A \wedge B) \equiv (\neg A \vee \neg B)$ (de Morgan)

$\neg(A \vee B) \equiv (\neg A \wedge \neg B)$ (de Morgan)

$(A \wedge B) \equiv \neg (\neg A \vee \neg B)$

$(A \vee B) \equiv \neg (\neg A \wedge \neg B)$

$(A \vee (B \wedge C)) \equiv ((A \vee B) \wedge (A \vee C))$ (distributivitate)

$(A \wedge (B \vee C)) \equiv ((A \wedge B) \vee (A \wedge C))$ (distributivitate)

Deoarece operatiile $\wedge$ si $\vee$ sunt asociative, se pot scrie expresii ca $A \wedge B \wedge C$ si $A \vee B \vee C$ – adică prin omiterea parantezelor care normal ar putea fi necesare – fără teamă de ambiguităti. Tot asa, fiind dată comutativitatea, aceste expresii pot fi gândite ca niste conjunctii sau disjunctii aplicate unor multimi de expresii.

Din setul de echivalente de mai sus, se poate vedea (cel putin informal) că orice expresie booleană poate fi scrisă utilizând numai operatorii $\wedge$ si $\neg$. Se poate înlocui $\Leftrightarrow$ cu $\Rightarrow$ si $\wedge$. Apoi se poate înlocui $\Rightarrow$ cu $\vee$ si $\neg$. Apoi se poate înlocui $\vee$ cu $\wedge$ si $\neg$. (O argumentatie similară arată că $\vee$ si $\neg$ sunt de asemenea

suficiente). Această argumentatie informală poate fi făcută riguroasă aplicând principiul inductiei pe expresii booleene. Mai departe în această sectiune se va arăta cum se utilizează principiul inductiei pentrua dovedi un rezultat mai tare: fiecare expresie booleană poate fi rescrisă utilizând un singur operator logic.

**O reprezentare minimală**

În afara găsirii unei reprezentări compacte pentru o functie booleană, proiectantii de circuite preferă adesea expresii care uzează de un singur tip de operator boolean – preferabil unul care corespunde unui circuit simplu cu tranzistor pe un cip. Operatorul boolean "nand" (nici), scris ca $A|B$ si echivalent cu $\neg(A \wedge B)$, este usor de implementat pe un cip. Se constată si următorul fapt interesant:

**Teorema 6.1**: Pentru orice expresie booleană, există o expresie logică echivalentă care utilizează numai operatorul | (nici).

Se dă o demonstratie inductivă completă pentru a vedea cum arată o inductie peste expresii booleene.

**Demonstratie**: Demonstratia este prin inductie pe expresii booleene pe variabilele **X**. Fie $P(B)$ propozitia conform căreia $B$ poate fi exprimată utilizând numai operatorul | (nici).

- Cazul de bază: a se dovedi că $P(T)$, $P(F)$ si $\forall X \in \mathbf{X}\ P(X)$ – este adevărat pentru că expresiile nu necesită nici un operator.
- Pasul inductiv ($\neg$): se dovedeste că $\forall B \in \mathbf{B}\ [P(B) \Rightarrow P(\neg B)]$.
    1. Ipoteza inductivă spune că $B$ poate fi exprimată utilizând numai |. Fie $NF(B)$ (forma NAND a lui $B$) o astfel de expresie
    2. De demonstrat: $\neg B$ poate fi exprimată utilizând numai |.
    3. Din definitia lui | avem
       $\neg B \equiv (B|B)$
       $\equiv (NF(B)|NF(B))$ prin ipoteza inductivă
    4. Asadar, există o expresie echivalentă pentru $\neg B$ care contine numai operatorul |.
- Pasul inductiv ($\wedge$): se demonstrează
  $\forall B_1, B_2 \in \mathbf{B}\ [P(B_1) \wedge P(B_2) \Rightarrow P(B_1 \wedge B_2)]$
    1. Ipoteza inductivă afirmă că $B_1$ si $B_2$ pot fi exprimate utilizând exclusiv operatorul |. Fie $NF(B_1)$ si $NF(B_2)$ acele expresii
    2. De demonstrat: $B_1 \wedge B_2$ poate fi exprimată folosind numai |
    3. Acum $\wedge$ este negarea lui |, astfel încât
       $(B_1 \wedge B_2) \equiv \neg(B_1|B_2) \equiv ((B_1|B_2)|(B_1|B_2))$
       $\equiv ((NF(B_1)|NF(B_2))|((NF(B_1)|NF(B_2))$ prin ipoteza inductivă
    4. Asadar, există o expresie echivalentă cu $(B_1 \wedge B_2)$ care contine numai operatorul |.
- Pasii rămasi (pentru $\vee$ , $\Rightarrow$ , $\Leftrightarrow$ ) sunt lăsati ca exercitiu.

Asadar, prin principiul inductiei pentru expresii booleene, pentru orice expresie booleană există o expresie logic echivalentă care utilizează numai operatorul |.
□
De observat utilizarea crucială a ipotezei inductive în această demonstratie! De pildă, în demonstratia pentru $\neg B$, expresia care contine numai | este expresia $NF(B)|NF(B)$. Expresia $B|B$ poate contine absolut orice deoarece $B$ este numai o expresie booleană arbitrară.
De observat că demonstratia dă direct un algoritm de conversie recursiv, cum este adesea cazul cu demonstratiile inductive. Totusi, conversia la forma NAND poate da o dezvoltare foarte extinsă a expresiei.
Pasii omisi în demonstratia de mai sus pot fi parcursi prin încă alte echivalente care includ pe |. O demonstratie similară stabileste că fiecare expresie booleană poate fi scrisă utilizând $\wedge$ si $\neg$ (sau $\vee$ si $\neg$), utilizând numai echivalențele standard date mai devreme. În esență, se utilizează echivalența care înlocuieste $\Leftrightarrow$ prin $\Rightarrow$ si $\wedge$ ; echivalenta care înlocuieste $\Rightarrow$ prin $\vee$ si $\neg$; echivalenta care înlocuieste $\vee$ cu $\wedge$ si $\neg$.

**Forme normale**

O **formă normală** a unei expresii este uzual o submultime de expresii de o sintaxă standard astfel încât pe de o parte orice expresie poate fi rescrisă în forma normală, pe de altă parte acea expresie în forma normală are anumite proprietăti interesante. Prin restrângerea formei, se pot adesea găsi algoritmi simpli si/sau eficienți pentru manipularea expresiilor.
Prima formă normală studiată aici este denumită forma normală disjunctivă sau DNF (*Disjunctive Normal Form*). În DNF, fiecare expresie este o *disjunctie* de *conjunctii* de *literale*. O **literală** este o variabilă booleană sau negatia sa. De exemplu, expresia următoare este în DNF:

$$(A \wedge \neg B) \vee (B \wedge \neg C) \vee (A \wedge \neg C \wedge \neg D)$$

De notat că DNF este generoasă cu operatorii dar foarte strictă în ceea ce priveste scrierea cu paranteze: un singur nivel de disjunctie si un singur nivel de conjunctie în interiorul fiecărui disjunct. DNF este o formă normală **completă**, adică se poate stabili următoarea teoremă:
**Teorema 6.2**: Pentru orice expresie booleană există o expresie DNF logic echivalentă.
**Demonstratie**: Fiind dată o expresie booleană $B$, se consideră descrierea ei prin tabelul de adevăr. În particular, se consideră acele linii ale tabelului de adevăr unde valoarea expresiei este $T$. Fiecare astfel de linie este specificată printr-o conjunctie de literale, o literală pentru fiecare variabilă. Disjuncția acestor conjuncții este echivalentă logic cu $B$.
□
DNF este foarte utilizată în proiectarea de circuite. De retinut că expresia DNF obtinută direct din tabelul de adevăr are atâția disjuncți câte valori $T$ se regăsesc în coloana-rezultat a tabelului de adevăr. **Minimizarea logică** se ocupă cu

metodele de a reduce dimensiunea acestor expresii prin eliminarea si combinarea disjuncților.

Dar în domeniul sistemelor de raționamente logice, este mult mai utilizată **forma normală conjunctivă** (CNF – *Conjunctive Normal Form*). În CNF, fiecare expresie este o *conjunctie* de *disjunctii* de *literale*. O disjunctie de literale este numită **clauză**. De pildă, expresia următoare este o CNF:

$$(\neg A \vee B) \wedge (\neg B \vee \neg C) \wedge (A \vee C \vee V)$$

Se poate dovedi cu usurință următorul rezultat:

**Teorema 6.3**: Pentru orice expresie booleană, există o expresie CNF logic echivalentă.

**Demonstratie**: orice expresie booleană $B$ este logic echivalentă cu conjuncția *negațiilor* fiecărei linii din tabelul ei de adevăr cu valoarea $F$. Negatia fiecărei linii este negatia unei conjunctii de literale care (conform legii lui de Morgan) este echivalentă unei disjuncții de negatii de literale care este echivalentă unei disjuncții de literale

□

Un alt mod de a găsi expresii CNF logic echivalente unei expresii date se bazează pe un proces de transformare recursiv. Aceasta nu cere construirea tabelului de adevăr pentru expresia în cauză si poate produce expresii CNF mult mai compacte.

Pasii sunt după cum urmează:

1. Se elimină $\Leftrightarrow$ înlocuind $A \Leftrightarrow B$ cu $(A \Rightarrow B) \wedge (B \Rightarrow A)$
2. Se elimină $\Rightarrow$ înlocuind $A \Rightarrow B$ cu $\neg A \vee B$
3. Se obtine o expresie care contine numai $\wedge$ , $\vee$ si $\neg$. Conversia lui $\neg CNF(A)$ în CNF, unde $CNF(A)$ este CNF echivalentă lui $A$ este extrem de dificilă. Asadar, e de preferat "a deplasa $\neg$ în interior" utilizând următoarele operatii:
$$\neg(\neg A) \equiv A$$
$$\neg (A \wedge B) \equiv (\neg A \vee \neg B) \text{ (de Morgan)}$$
$$\neg (A \vee B) \equiv (\neg A \wedge \neg B) \text{ (de Morgan)}$$
Prin aplicarea repetată a acestor operatii rezultă o expresie care contine operatorii $\wedge$ si $\vee$ ascunsi în paranteze, aplicati unor literale (Acest fapt este usor de probat prin inductie, foarte asemănător cu demonstrația de la NAND).
4. Acum se aplică legea de distributivitate, distribuind $\wedge$ peste $\vee$ ori de câte ori este posibil, cu rezultat o expresie CNF.

Acum urmează demonstratia formală pentru pasul ultim: el rezultă, cum s-a afirmat, ca o expresie CNF.

**Teorema 6.4**: Fie $B$ o expresie booleană construită din operatorii $\wedge$ , $\vee$ si $\neg$, cu $\neg$ aplicat numai variabilelor. Există o expresie CNF echivalentă logic cu $B$.

Evident, această teoremă se poate demonstra simplu apelând la schema propusă la demonstrarea teoremei 6.3. Dar asta ar însemna un algoritm bazat pe construirea unui tabel de adevăr, ceea ce poate fi evitat. Să vedem cum se face aceasta recursiv.

**Demonstrație**: Demonstrația este prin inductie pe expresii booleene de variabilele **X**. Fie $P(B)$ propozitia conform căreia $B$ poate fi exprimată în CNF;

presupunem că $B$ contine numai operatorii $\wedge$ , $\vee$ si $\neg$, cu $\neg$ aplicat numai variabilelor.

- Cazul de bază: $P(T)$, $P(F)$ si $\forall X \in \mathbf{X}\ P(X)$ si $\forall X \in \mathbf{X}\ P(\neg X)$.
  Acestea sunt adevărate deoarece o conjunctie a unei disjunctii pentru o literală este echivalentă cu acea literală.
- Pasul inductiv ($\wedge$ ): trebuie dovedit că $\forall B_1, B_2 \in \mathbf{B}\ [P(B_1) \wedge P(B_2) \Rightarrow P(B_1 \wedge B_2)]$
  1. Ipoteza inductivă afirmă că $B_1$ si $B_2$ pot fi exprimate în CNF. Fie $CNF(B_1)$ si $CNF(B_2)$ cele două expresii de acest gen.
  2. De demonstrat: $B_1 \wedge B_2$ poate fi exprimată în CNF.
  3. Prin ipoteza inductivă, avem
     $$B_1 \wedge B_2 \equiv CNF(B_1) \wedge CNF(B_2)$$
     $$\equiv \left(C_1^1 \wedge \ldots \wedge C_1^m\right) \wedge \left(C_2^1 \wedge \ldots \wedge C_2^n\right) \ (C_j^i \text{ sunt clauze})$$
     $$\equiv C_1^1 \wedge \ldots \wedge C_1^m \wedge C_2^1 \wedge \ldots \wedge C_2^n$$
  4. Asdar, $B_1 \wedge B_2$ este echivalentă unei expresii CNF.
- Pasul inductiv ($\vee$ ): trebuie dovedit că $\forall B_1, B_2 \in \mathbf{B}\ [P(B_1) \vee P(B_2) \Rightarrow P(B_1 \vee B_2)]$
  1. Ipoteza inductivă afirmă că $B_1$ si $B_2$ pot fi exprimate în CNF. Fie $CNF(B_1)$ si $CNF(B_2)$ cele două expresii de acest gen.
  2. De demonstrat: $B_1 \vee B_2$ poate fi exprimată ca o CNF.
  3. Prin ipoteza inductivă, avem
     $$B_1 \vee B_2 \equiv CNF(B_1) \vee CNF(B_2)$$
     $$\equiv \left(C_1^1 \wedge \ldots \wedge C_1^m\right) \vee \left(C_2^1 \wedge \ldots \wedge C_2^n\right) \ (C_j^i \text{ sunt clauze})$$
     $$\equiv \left(C_1^1 \vee C_2^1\right) \wedge \left(C_1^1 \vee C_2^2\right) \wedge \ldots \wedge \left(C_1^m \vee C_2^{n-1}\right) \wedge \left(C_1^m \vee C_2^n\right)$$
  4. Prin asociativitatea operatiei $\vee$ , fiecare expresie de forma $\left(C_1^i \vee C_2^j\right)$ este echivalentă unei singure clauze care contine toate literalele din cele două clauze.
  5. Asdar, $B_1 \vee B_2$ este echivalentă unei expresii CNF.

Prin urmare, orice expresie booleană construită cu opeartorii $\wedge$ , $\vee$ si $\neg$, cu $\neg$ aplicată numai variabilelor, este echivalentă logic unei expresii în CNF

□

Acest proces "aplatizează" prin urmare expresia logică, care poate avea multe niveluri de paranteze, în două niveluri. În proces, expresia se poate extinde enorm; pasul distributiv care converteste DNF în CNF poate da o explozie exponentială dacă este aplicată disjunctiilor una-în-alta (a se vedea mai jos). Cât despre conversia la forma NAND, demonstrația dă direct un algoritm de conversie recursiv.

**Conversia directă între CNF si DNF**

Să examinăm pe scurt conversia directă a DNF-urilor la CNF si invers. Se utilizează regulile distributivitătii; deoarece acestea sunt simetrice în raport cu

$\wedge$ si $\vee$ , ceea ce se constată pentru un sens al conversiei se aplică si celuilalt. Asa că să ne ocupăm de transformarea DNF la CNF.
La început un caz foarte simplu:

$$(A \wedge B) \vee (C \wedge D) \equiv (A \vee (C \wedge D)) \wedge (B \vee (C \wedge D))$$
$$\equiv (A \vee C) \wedge (A \vee D) \wedge (B \vee C) \wedge (B \vee D)$$

Acum se adaugă încă un termen:

$$(A \wedge B) \vee (C \wedge D) \vee (E \wedge F)$$
$$\equiv [(A \vee C) \wedge (A \vee D) \wedge (B \vee C) \wedge (B \vee D)] \vee (E \wedge F)$$
$$\equiv \{[(A \vee C) \wedge (A \vee D) \wedge (B \vee C) \wedge (B \vee D)] \vee E\}$$
$$\wedge \{[(A \vee C) \wedge (A \vee D) \wedge (B \vee C) \wedge (B \vee D)] \vee F\}$$
$$\equiv (A \vee C \vee E) \wedge (A \vee D \vee E) \wedge (B \vee C \vee E) \wedge (B \vee D \vee E)$$
$$\wedge (A \vee C \vee F) \wedge (A \vee D \vee F) \wedge (B \vee C \vee F) \wedge (B \vee D \vee F)$$

Tiparul devine clar: caluzele CNF constau din fiecare $k$-tuplu posibil de literale luat din cei $k$ termeni din DNF, câte unul din fiecare. Astfel, se poate formula conjectura că dacă o expresie DNF are $k$ termeni, fiecare continând $l$ literale, CNF echivalentă obținută prin distributivitate va avea un număr de $l^k$ clauze, fiecare continând $k$ literale (ceea ce se poate demonstra prin inductie). Asadar, poate exista o explozie exponențială în convertirea de la DNF la CNF. Lectia următoare aduce în discutie faptul că este aproape inevitabil ca unele expresii CNF mici să aibă o cea mai mică DNF echivalentă care este exponențial mai mare.

## Lecția 7

Sectiunea 6 a acestor Note de curs a introdus formele normală disjunctivă (DNF) si normală conjunctivă (CNF) si a afirmat că CNF este mult mai naturală pentru aplicatiile care implică rationamente. De ce? Sunt două motive evidente.
Primul, rationamentul utilizează colectii de propozitii (sentences[3]) (denumite uneori baze de date sau baze de cunostințe) care sunt exprimate natural ca o conjunctie de propozitii (sentences) – bazele de cunostințe se consideră cu toate propozițiile (sentences) adevărate. Traducerea aceastor conjuncții în DNF ar putea implica o muncă inutilă si o extindere a dimensiunii reprezentărilor.
Al doilea, multe propoziții (sentences) utilizate în raționamente sunt implicații cu antecedente multe si concluzie unică. Acestea au forma $P_1 \wedge \ldots \wedge P_k \Rightarrow Q$ (Programele logice constau pe de-a-ntregul în astfel de propoziții (sentences)). Aceasta se transformă astfel:

$$\begin{aligned}[P_1 \wedge \ldots \wedge P_k \Rightarrow Q] &\equiv [\neg(P_1 \wedge \ldots \wedge P_k) \vee Q] \\ &\equiv [(\neg P_1 \vee \ldots \vee \neg P_k) \vee Q] \quad \text{(de Morgan)} \\ &\equiv [\neg P_1 \vee \ldots \vee \neg P_k \vee Q] \quad \text{(asociativitate)}\end{aligned}$$

Asadar, orice propoziție gen implicație se converteste usor într-o clauză cu acelasi număr de literale.
Multe probleme interesante din stiința computerelor pot fi convertite la reprezentarea CNF si apoi solutia este readusă la limba originară a problemei. De ce se procedează asa?

- Deoarece se poate lucra la găsirea de algoritmi eficienți pentru CNF în loc a găsi algoritmi eficienți pentru sute de probleme diferite
- Deoarece se poate câstiga prin a profita de munca făcută de alți oameni pentru a găsi algoritmi eficienți pentru CNF
- Deoarece uneori găsim, odată stabilită CNF, că avem un caz special sau altul de CNF pentru care sunt cunoscuți algoritmi foarte eficienți (cum sunt cei liniari în timp).

Există alte obiective legate de "formele canonice" dincolo de CNF, inclusiv inversarea matricilor si calculul determinanților, **programarea liniară** si stabilirea rădăcinilor polinoamelor. Pe măsură ce cineva devine un specialist bun în calculatoare, acela îsi dezvoltă un "*web*" mental de probleme de calcul standard înrudite si învață să potrivească (*to map*) orice problemă nouă pe acest *web*. Jocul Minesweeper este un exemplu.

[3] Am mentinut (si) termenul din textul original, *sentence*, care poate fi diferit ca sens de celălalt utilizat până acum, *proposition*, care va fi utilizat si în continuare fără dublura lui din limba originalului.

**Jocul Minesweeper**

Regulile jocului Minesweeper sunt următoarele:

- Jocul este jucat de un singur jucător pe o tablă *X*x*Y* (se folosesc coordonatele carteziene astfel încât (1, 1) este în stânga jos si (*X*, 1) este în dreapta jos). Afisajul este la început vid. Jucătorului i se spune numărul total de mine (rămase) nedescoperite; acestea sunt distribuite uniform aleator pe tablă (vezi figura 1(*a*)).
- La fiecare tur jucătorul are trei optiuni:

1. *Să marcheze* un pătrat ca o mină; afisajul este actualizat si numărul total de mine este scăzut cu 1 (indiferent dacă mina există în realitate)
2. *Să de-marcheze* un pătrat; marcajul de mină este sters din pătrat, se revine la blank
3. Să verifice un pătrat; dacă pătratul este minat jucătorul pierde. Altminteri, afisajul este actualizat pentru a arăta *numărul* de mine în pătratele adiacente (adiacente orizontal, vertical sau diagonal). Dacă acest număr este zero, pătratele ediacente sunt verificate automat recursiv până când se atinge un rezultat al numărătorii nenul.

- Jocul este câstigat când toate minele au fost descoperite, localizate si toate pătratele fără mine au fost verificate (vezi figura, sectiunea (*b*)).

| Mine rămase: 6 | | | | | | Mine rămase: 0 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | | | | | | 4 | m | 2 | 2 | m |
| 3 | | | | | | 3 | 3 | 4 | m | 2 |
| 2 | | | | | | 2 | m | m | 2 | 1 |
| 1 | | | | | | 1 | m | 3 | 1 | 0 |
| | 1 | 2 | 3 | 4 | | | 1 | 2 | 3 | 4 |
| (*a*) | | | | | | (*b*) | | | | |

| Mine rămase: 1 | | | | | | Mine rămase: 2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |
| 2 | | | | | 2 | | | | | |
| 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| | 1 | 2 | 3 | | | 1 | 2 | 3 | 4 | 5 |
| (*c*) | | | | | | (*d*) | | | | |

**Exemple de Minesweeper**. (*a*) afisajul initial al jocului 4x4. (*b*) afisajul final după descoperirea tuturor minelor. (*c*) cazul simplu: numai o solutie. (*d*) două solutii posibile, dar ambele au (3, 1) blank.

Să definim un pătrat sigur ca unul care, fiind dată informatia disponibilă, nu poate contine o mină. Evident, orice jucător ar vrea să testeze numai pătrate

sigure si să marcheze ca min(at)e numai acele pătrate care sunt cert în acea stare. Asadar, noțiunea de verificare logică este centrală pentru jocul Minesweeper.

Mulți pasi în jocul Minesweeper implică simplu "completarea" in jurul unui pătrat – pătratul este cunoscut a avea *k* mine în jur si aceste *k* mine sunt deja descoperite, astfel că toate pătratele adiacente rămase sunt sigure (unele implementări oferă posibilitatea de a face asta cu un singur click). Cazul dual este acela în care un pătrat este cunoscut a avea *k* mine adiacente si *k* pătrate adiacente blank astfel încât ele pot fi toate mine. Marea majoritate a rezultatelor implică unul din aceste două tipuri de pasi.

Câteva exemple simple de rationamente netriviale în Minesweeper: Primul are în vedere figura 1(*c*). Se porneste cu 1 în (1, 1); asta implică faptul că există o mină în (1, 2) sau (2, 2). Această mină "satisface" unitatea din (2, 1); astfel (3, 2) este sigur (nu are mină). Similar, pornind cu 1 în (3, 1) se poate arăta că (1, 2) este sigur. Asfel, (2,2) are mina.

În figura 1 (*d*) se poate repeta rationamentul de mai sus de la orice cap de linie pentru a stabili că (3, 2) este sigur. Dar există două lumi posibile consistente cu toată informatia (adică baza de cunostinte are *două modele*): mine în (1, 2) si (4, 2) sau mine în (2, 2) si (5, 2). Nu se pot adăuga informatii suplimentare: verificarea lui (3, 2) nu este de nici un ajutor.

Jucând Minesweeper ca persoană umană, se învață treptat recunoasterea unui set de pattern-uri cu demonstrațiile logice asociate de dificultate variabilă. Fiecare pare a fi *ad hoc* si ele nu sunt cu siguranță nici sistematice si nici complete, în sensul că tot cu siguranță jucătorul omite unele instanțe în care poate fi făcută o miscare logică.

La această etapă se poate formula jocul Minesweeper ca o problemă de rationament logic. Aici se va parcurge numai un exemplu simplu cu concentarea pe procesele logice din raționamente. Lecția următoare va conține o formulare mai completă.

Pornire: exemplul simplu din figura 1 (*c*). Mai întâi variabilele. Se pune $X_{x,y}$ a fi adevărată dacă si numai dacă (*x*, *y*) contine o mină. De exemplu, $X_{1,2}$ este adevărată dacă (1, 2) (stânga sus) contine o mină. Sunt cunoscute următoarele fapte:

- (1, 1) are o mină adiacentă, astfel exact una din variabilele $X_{1,2}$ si $X_{2,2}$ este adevărată. Să numim această propozitie $N_{1,1}$; ea este echivalentă cu două disjunctii. Prima spune că cel putin una este adevărată, a doua spune că cel putin una este falsă
$$(X_{1,2} \vee X_{2,2}) \wedge (\neg X_{1,2} \vee \neg X_{2,2})$$
- (2, 1) are o mină adiacentă, asa că exact una din $X_{1,2}$, $X_{2,2}$ si $X_{3,2}$ este adevărată. Aceasta este propozitia $N_{2,1}$:
$$(X_{1,2} \vee X_{2,2} \vee X_{3,2}) \wedge (\neg X_{1,2} \vee \neg X_{2,2}) \wedge (\neg X_{2,2} \vee \neg X_{3,2}) \wedge (\neg X_{3,2} \vee \neg X_{1,2})$$
- (3, 1) are o mină adiacentă, asa că exact una din $X_{2,2}$ si $X_{3,2}$ este adevărată. Aceasta este propozitia $N_{3,1}$:
$$(X_{2,2} \vee X_{3,2}) \wedge (\neg X_{2,2} \vee \neg X_{3,2})$$

- Există exact o mină rămasă. Aceasta este "restrictia globală" $G$:
  $(X_{1,2} \vee X_{2,2} \vee X_{3,2}) \wedge (\neg X_{1,2} \vee \neg X_{2,2}) \wedge (\neg X_{2,2} \vee \neg X_{3,2}) \wedge (\neg X_{3,2} \vee \neg X_{1,2})$

De observat că în acest caz $G$ este exact acelasi lucru cu $N_{2,1}$.
Conjuncția propozițiilor $N_{1,1} \wedge N_{2,1} \wedge N_{3,1} \wedge G$ este o reprezentare CNF a tot ceea ce se cunoaste, dat fiind afisajul. Fie $d$ afisajul si $CNF(d)$ reprezentarea lui ca CNF. Suntem interesați acum în a decide care pătrate sunt sigure si care sunt minate. De pildă, întrebarea dacă pătratul (1, 2) este sigur corespunde cu a decide dacă

$$CNF(d) \models \neg X_{1,2}$$

O dovadă că $CNF(d)$ are drept consecintă $\neg X_{1,2}$ oferă o garantie completă că pătratul (1, 2) este sigur deoarece asta înseamnă că nu este nici o mină în (1, 2) în orice lume posibilă (configuratie de mine) consistentă cu ceea ce spune afisajul.

**Consecința si dovada**

Această secțiune oferă o metodă de verificare simplă si completă care derivă direct din definiția consecinței necesare (entailment). În lectia 1 s-a dat o definiție în termeni de "lumi posibile". Este posibilă o concizie întrucâtva mai pronunțată. Spunem că o atribuire (assignment) completă $M$ este **un model** al unei propozitii $P$ dacă $P$ este adevărată în $M$. Atunci avem definitia următoare:
**Definitia 7.1** (**consecința = entailment**): $P \models Q$ dacă si numai dacă $Q$ este adevărată în orice model al lui $P$.
Să ilustrăm ideea pentru jocul Minesweeper. $P$ este propozitia care corespunde tuturor informatiilor cunoscute, $CNF(d)$. $Q$ este propozitia conform căreia pătratul (1, 2) este sigur, adică $\neg X_{1,2}$. Variabilele sunt $X_{1,2}$, $X_{2,2}$ si $X_{3,2}$ astfel că există 8 modele. Le putem verifica pe fiecare (adică fiecare configuratie de mine), să vedem dacă este un model pentru $CNF(d)$ (adică este consistent cu afisajul) si dacă este asa să verificăm dacă este si model pentru $\neg X_{1,2}$ (adică nu există mină în pătratul (1, 2)).
Similar, dacă (2, 2) contine o mină în orice model al $CNF(d)$, atunci am dovedit că (2, 2) contine o mină. În figura 1 ($d$), unele modele pentru $CNF(d)$ au o mină în (2, 3), unele nu au, prin urmare nu se poate dovedi nimic.
Pentru logica propozițională în general, expresiile finite pot contine numai un număr finit de variabile astfel că numărul de modele posibile este finit. Asadar, se poate totdeauna utiliza această demonstratie-prin-tabel-de-adevăr, care mai este numită si **model-checking**:

pentru a determina dacă $P \models Q$, în care $P$ si $Q$ sunt expresii booleene de $X_1, \ldots, X_n$:
  pentru fiecare model posibil $M = \{X_1 = t_1, \ldots, X_n = t_n\}$
    dacă $P$ este adevărată în $M$
      atunci dacă $Q$ este falsă în $M$ returnează "nu"
  returnează "da"

Acest algoritm va fi făcut mai concret în secțiunea următoare. Pentru moment, a se nota că în cel mai rău caz, timpul de executie este $O(2^n)$, deoarece sunt $2^n$

modele de verificat. De observat totodată că un model este reprezentat ca un *set* de asignări ale varibilelor individuale.

**Validitatea si posibilitatea-de-a-satisface o propoziție**

**Definitia 7.2 (validitate)**: O propozitie **validă** (cunoscută si ca o **tautologie**) este o propoziție care este adevărată pentru *orice* model posibil.
Deoarece *T* este adevărată pentru orice model posibil, o propozitie (sentence) validă este echivalentă logic cu *T*. La ce sunt bune propozitiile (sentence) valide? Din definitia consecinței necesare (entailment) se poate extrage faptul următor:
**Teorema 7.1**: $P \models Q$ dacă si numai dacă propozitia ($P \Rightarrow Q$) este validă.
Aceasta este numită uneori *teorema deducției*. Demonstrația decurge direct din definiția implicației. Se poate imagina o demonstrație prin model-checking ca un test de validitate, care cere o verificare pe toate modelele.
Pentru unele probleme, suntem satisfăcuți a găsi *orice* (any) model în care informația cunoscută este adevărată. De pildă, să presupunem că ni se dă o tablă de Minesweeper cu unele mine marcate si unele pătrate cunoscute sau necunoscute si suntem îndemnați a determina dacă informația dată este consistentă cu unele configuratii posibile (dacă nu, atunci cineva a comis o greseală!). Apoi ne întrebăm dacă propoziția care descrie tabla dată poate fi vreodată satisfăcută:
**Definitia 7.3** (**propoziție posibil-a-fi-satisfăcută**): O propozitie *posibil-a-fi-satisfăcută* este o propozitie care este adevărată pentru (cel putin) *un* model.
Spunem că dacă o propoziție *P* este adevărată pentru un model *M*, atunci *M* satisface pe *P*. Posibilitatea ca o propoziție să fie satisfăcută poate fi verificată printr-o variantă aproape evidentă a algoritmului de mai sus, adaptată pentru a fi utilă în demonstratie:

pentru a determina dacă *P* este posibil-a-fi-satisfăcută, unde *P* este o expresie booleană în $X_1, \ldots, X_n$:
  pentru fiecare model posibil $M = \{X_1 = t_1, \ldots, X_n = t_n\}$
    Dacă *P* este adevărată în *M* atunci returnează "da"
  returnează "nu"

Multe probleme din stiința calculatoarelor sunt realmente probleme care vizează posibilitatea-de-a-fi-satisfăcute anumite propozitii. Ca exemplu s-ar putea da o problemă de orar (timetabling) formulată în una din temele de casă; aceasta este o instanță a unei clase imense de probleme denumite probleme de satisfacere cu restricții, în care trebuie găsită mulțimea de valori ale unor variabile astfel încât o colectie de restricții este satisfăcută în totalitate.
Validitatea si posibilitatea-de-a-fi-satisfăcută sunt desigur în conexiune: *P* este validă dacă si numai dacă $\neg P$ este imposibil-a-fi-satisfăcută; prin contrapozitie, *P* este posibil-a-fi-satisfăcută dacă si numai dacă $\neg P$ nu este validă. Avem, de asemenea, rezultatul util următor:
**Teorema 7.2**: $P \models Q$ dacă si numai dacă propozitia ($P \wedge \neg Q$) este imposibil-a-fi-satisfăcută.

Ca exercițiu, a se încerca demonstrarea acestei teoreme. Ea corespunde exact unei demonstratii prin *reducere la absurd* (*reductio ad absurdum*) – imposibilitatea-de-a-fi-satisfăcută înseamnă că o contradictie trebuie să fie continută în ($P \wedge \neg Q$). Ce înseamnă aceasta în practică este că putem testa consecința (entailment) la fel ca si posibilitatea-de-a-fi-satisfăcută utilizând un algoritm legat de posibilitatea-de-a-fi-satisfăcută o propoziție. Astfel, de acum încolo se va vorbi de modul cum se implementează testarea posibilitătii-de-a-fi-satisfăcută o propozitie în loc de o consecintă necesară (entailment).

**Testarea recursivă a posibilitătii-de-satisfacere a unei propozitii**

Lectia 6 a dat o definitie simplă (*eval*) pentru evaluarea unei expresii booleene într-un model. Dacă toate expresiile sunt în CNF, se poate utiliza o metodă încă mai simplă: o expresie CNF este adevărată dacă si numai dacă fiecare clauză în parte este adevărată; o clauză este adevărată dacă si numai dacă o literală este adevărată.
Acum, cum se generează modelele? De preferat a nu se construi mai întâi întregul tabel de adevăr si apoi a fi parcurs de-a lungul si de-a latul! Asta ar necesita un spatiu exponential si spatiul este uneori mult mai costisitor decât timpul. În locul unei astfel de proceduri, se pot enumera modelele recursiv după cum urmează. Fie $M_{1...i}$ un model partial care specifică valori pentru variabilele $X_1, \ldots, X_i$ si fie o *completare* a lui $M_{1...i}$ orice model $X_1, \ldots, X_n$ care coincide cu $M_{1...i}$ pe $X_1, \ldots, X_i$. Acum se defineste funcția *satisface*($P$, $M_{1...i}$) a fi adevărată dacă si numai dacă $P$ este adevărată într-un model care este o completare a lui $M_{1...i}$. Evident, $P$ este posibil-a-fi-satisfăcută dacă si numai dacă *satisface*($P$, {}) este adevărată.

*satisface*($P$, $M_{1...n}$) este adevărată dacă si numai dacă $P$ este adevărată în $M_{1...n}$.
dacă $i < n$, *satisface*($P$, $M_{1...i}$) este adevărată dacă si numai dacă
*satisface*($P$, $M_{1...i} \cup \{X_{i+1} = T\}$) este adevărată
sau
*satisface*($P$, $M_{1...i} \cup \{X_{i+1} = F\}$) este adevărată

În timp ce tabelul de adevăr ocupă un spațiu exponențial, acest algoritm consumă numai un spațiu liniar – profunzimea recursiei este cel mult $n$. Durata algoritmului este încă $O(2^n)$ în cel mai rău caz, care se produce când $P$ este imposibil-a-fi-satisfăcută. Dacă $P$ este posibil-a-fi-satisfăcută, durata de calcul poate fi mult mai mică.
Pentru discuția legată de jocul Minesweeper, e de asteptat ca multe dintre pătrate să fie nici garantat minate nici garantat sigure (în special când cazurile "evidente" au fost deja evidențiate), asa încât multe încercări de verificare se vor termina fără a fi necesară enumerarea tuturor modelelor.

# Lecția 8

Notele lecției a 7-a au descris metode de raționament logic bazat pe testarea posbilității-de-satisfacere a unei propoziții si le-au introdus în jocul cunoscut sub numele de Minesweeper. Lecția aceasta arată cum se construieste un program Minesweeper complet. Se va vedea că unele aspecte ale jocului sunt intractabile sub aspectul calculelor dacă sunt manipulate la modul naiv. Metodele de descompunere a problemei pot fi de ajutor.

## Minesweeper în CNF

Lecția anterioară a dat un exemplu simplu de cum se formulează o descriere logică a unui afisaj Minesweeper printr-un set de clauze. Dacă *d* este un afisaj, *CNF*(*d*) reprezintă expresia CNF corespunzătoare. Vom arăta acum cum se construieste sistematic *CNF*(*d*) pentru un afisaj oarecare.
*CNF*(*d*) constă în propoziții generate de fiecare dintre pătratele cunoscute, la care se adaugă restricția globală relativ la numărul total de mine rămase. Se începe cu pătratele cunoscute. Se consideră un pătrat cunoscut, cum ar fi (2, 1) în exemplul care urmează (reluat din lecția a 7-a):

| | | | |
|---|---|---|---|
| 2 | | | |
| 1 | 1 | **1** | 1 |
| | 1 | 2 | 3 |

(2, 1) are o mină adiacentă. Sunt cinci pătrate adiacente; două din ele au fost verificate si sunt cunoscute ca sigure; 0 din ele sunt marcate deja ca minate. Sunt $n = 5 - 2 - 0 = 3$ pătrate adiacente necunoscute dintre care $k = 1 - 0$ sunt minate. Astfel, este necesar a exprima în CNF propoziția conform căreia *k* din cele *n* pătrate adiacente necunoscute sunt cu mine. Denumim această propoziție *KN*(*k*, *n*).
Să vedem cu ce instrumente lucrăm. CNF cere o conjuncție de disjuncții de literale. O disjuncție de literale înseamnă "cel puțin unul din termeni (literale) este adevărat", adică o inegalitate. Literalele pot fi fie "(*i*, *j*) conține o mină" fie "(*i*, *j*) nu conține o mină". Deoarece *KN*(*k*, *n*) este simetrică pe de-a-ntregul în raport cu pătratele necunoscute, se poate prezuma generarea de clauze care sunt simetrice – de pildă, toate literalele pozitive sau toate literalele negative.
Începem prin a scrie *KN*(*k*, *n*) ca două inegalități:

$$KN(k, n) \equiv (U(k, n) \wedge L(k, n))$$

în care

$U(k, n)$ înseamnă că cel mult $k$ din cele $n$ pătrate conțin o mină
$L(k, n)$ înseamnă că cel puțin $k$ din cele $n$ pătrate conțin o mină
Dar cum se exprimă "cel mult $k$" utilizând clauze care spun "cel puțin unul"? Se consideră orice submulțime de $k + 1$ pătrate din cele $n$ necunoscute. Dacă cel puțin $k$ sunt minate, atunci cel puțin unul nu este minat; situația inversă este de asemenea adevărată. Astfel, avem:

$U(k, n) \equiv$ pentru orice $k + 1$ pătrate din cele $n$, cel puțin unul nu este minat

Similar, considerând orice submulțime de $n - k + 1$ pătrate: dacă cel puțin $k$ din toate cele $n$ pătrate sunt minate, atunci cel puțin unul din oricare $n - k + 1$ pătrate trebuie să fie cu mină; reciproca este adevărată. Asadar

$L(k, n) \equiv$ pentru orice $k + 1$ pătrate din cele $n$, cel puțin unul este minat

Aplicând aceste formulări pătratului (2, 1) din exemplul de mai sus, unde $k = 1$ si $n = 3$, se obține:

$U(1, 3) \equiv$ pentru orice 2 pătrate din cele 3, cel puțin unul nu este minat

$L(1, 3) \equiv$ pentru orice 3 pătrate din cele 3, cel puțin unul este minat

Traducând într-o expresie booleană, se obțin relațiile

$$U(1, 3) \equiv (\neg X_{1,2} \vee \neg X_{2,2}) \wedge (\neg X_{2,2} \vee \neg X_{3,2}) \wedge (\neg X_{3,2} \vee \neg X_{1,2})$$
$$L(1, 3) \equiv (X_{1,2} \vee X_{2,2} \vee X_{3,2})$$

exact ca în lecția 7.
De observat că aceste espresii sunt valide dacă avem $k > 0$ si $k + 1 \leq n$. Cazul $k = 0$ înseamnă simplu că expresia $KN(0, n)$ este conjuncția clauzelor $\neg X_i$ pentru toți $i$. Cazul $k + 1 > n$ poate să apară numai dacă avem $k = n$, adică toate $n$ variabilele sunt cu mine; atunci avem simplu clauzele $X_i$ pentru toți $i$.
Se poate de asemenea genera o expresie CNF recursiv după cum urmează:

$$KN(k, n) \equiv ((X_n \Rightarrow KN(k - 1, n - 1)) \wedge (\neg X_n \Rightarrow KN(k, n - 1)))$$

cu cazurile de bază la $k = n$ si la $k = 0$ ca mai sus. Presupunând că atât $KN(k - 1, n - 1)$ cât si $KN(k, n - 1)$ pot fi exprimate în CNF, este un pas simplu care uzează de distributivitate pentru a exprima $KN(k, n)$ în CNF. Expresia care rezultă din această recursivitate arată usor diferit de aceea obținută mai devreme, dar cele două variante sunt echivalente logic.
În plus, față de restricțiile "locale" rezultate din pătratele deja verificate, avem si restricția globală rezultată din numărul total de mine rămase, $M$:

$G$: Exact $M$ din pătratele necunoscute pe afisaj conțin mine.

Dacă sunt $B$ pătrate rămase necunoscute, acestea sunt o mulțime de clauze de forma $KN(M, B)$.

**Preliminarii: numărătoarea de obiecte**

Vom face acum o usoară digresiune pentru a intra provizoriu în chestiunea numărului de clauze generate pentru Minesweeper.
Fie $|KN(n,k)|$ numărul de clauze din $KN(n,k)$, utilizând prima noastră construcție. Cât de mare este acesta? Avem ecuația următoare:

$$|KN(n,k)| = |L(n,k)| + |U(n,k)| = C(n, n - k + 1) + C(n, k + 1)$$

unde notația $C(n, k)$ este definită imediat:

**Definiția 8.1** (**combinări**): $C(n, k)$ este numărul de submulțimi distincte de dimensiune (cardinal) $k$ extrase dintr-o mulțime de dimensiune (cardinal) $n$.
De exemplu, $C(4, 2) = 6$ deoarece sunt 6 submulțimi de dimensiune 2 în orice mulțime de dimensiune 4. $C(n, k)$ se pronunță uneori ca "alege $k$ din $n$". Se defineste imediat o cantitate înrudită, $P(n, k)$:
**Definitia 8.2** (**permutări**): $P(n, k)$ este numărul distinct de $k$-tuple ordonate distinct estrase fără înlocuire dintr-o multime de dimensiune $n$.
Deosebirea principală între $P(n, k)$ si $C(n, k)$ este aceea că pentru $P(n, k)$ contează ordinea, pentru $C(n, k)$ nu contează ordinea. Este mai usor a obtine mai întâi o formulă pentru $P(n, k)$:
**Teorema 8.1**: Pentru orice numere naturale $n$, $k$ astfel încât $k \leq n$,

$$P(n,k) = \frac{n!}{(n-k)!}$$

**Demonstratie**: Primul element al tuplului poate fi extras în $n$ moduri, al doilea în $(n – 1)$ moduri si asa mai departe până la ultiul element care poate fi extras în $n – k + 1$ moduri. Asadar, $P(n, k) = n(n – 1)\ldots(n – k + 1) = n!/(n – k)!$
□
Orice submultime de $k$ elemente din $n$ va apărea repetat în multimea de permutări cu $k!$ ordonări diferite. Asadar, reazultă pentru $C(n, k)$ formula următoare:
**Teorema 8.2**: Pentru orice numere naturale $n$, $k$ astfel încât $k \leq n$,

$$C(n,k) = \frac{n!}{k!(n-k)!}$$

Se poate vedea, prin simetrie, că are lor identitatea următoare:

$$C(n, k) = C(n, n – k)$$

Prin urmare, revenind la formula de mai sus pentru numărul de clauze generate, avem

$$|KN(n,k)| = C(n, n – k + 1) + C(n, k + 1) = C(n, k – 1) + C(n, k + 1)$$

În cazul cel mai nefericit, pentru un pătrat anumit, $n = 8$ si $k = 4$, ceea ce produce $|KN(8, 4)| = C(8, 3) + C(8, 5) = 112$ clauze. Acest număr nu este prea mare. Dar pentru restrictia globală, pe o tablă de 8x8 cu 10 mine (un caz usor), se obitne $|KN(64, 10)| = C(64, 9) + C(64, 11) = 771.136.366.336$. Astfel, trebuie gândit un alt mod de a manipula restrictia globală!
Întâmplător, nu este prea greu a demonstra existenta unei mărginiri inferioare pentru numerele cele mai mari $C(n, k)$, pentru orice $n$ dat.
**Teorema 8.3**: Pentru un $k$ oarecare, $C(n, k) \geq 2^n/(n + 1)$.
**Demonstratie**: Se consideră suma după $k$ a numerelor $C(n, k)$. Aceasta este suma numărului de submultimi de dimensiunea $k$, pentru toti $k$. Este tocmai totalul numerelor de submultimi ale unei multimi de dimensiune $n$, care este $2^n$, adică

$$\sum_{k=0}^{n} C(n,k) = 2^n$$

Acum, suma contine $n + 1$ termeni, asa încât cel putin unul dintre acestia trebuie să fie mai mare sau egal cu $2^n/(n + 1)$.
□
Ultimul pas este o aplicatie a principiului generalizat "*pigeonhole*": dacă $N$ obiecte sunt plasate în $k$ cutii, atunci există cel putin o cutie cu cel putin $\lceil N/k \rceil$ obiecte.
Finalul digresiunii! Vom reveni la numărare mai târziu când se va vorbi de probabilități.

**Un algoritm "brain-dead" pentru Minesweeper**

Fiind dată o reprezentare CNF a unui afisaj de Minesweeper *CNF*(*d*), există unele inferențe "evidente" care se pot face. De pildă, în exemplul din partea stângă a figurii alăturate, reprezentarea CNF este alcătuită din trei clauze simple, unitare, cu literale pozitive:

$$(X_{1,2}) \wedge (X_{2,2}) \wedge (X_{2,1})$$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | | | | | 2 | | |
| 1 | 3 | | | | 1 | 1 | m |
| | 1 | 2 | | | | 1 | 2 |

Simplu, prin eliminarea-de-și, se poate vedea că cele trei pătrate contin mine (cum era de asteptat). Similar, în exemplul din partea dreaptă, reprezentarea CNF este

$$(\neg X_{1,2}) \wedge (\neg X_{2,2})$$

Din nou, avem clauze unitare, simple, de data aceasta cu literale negate/ negative si putem conchide imediat că pătratele (1, 2) si (2, 2) sunt sigure. Astfel, concluziile care sunt "evidente" unui jucător uman sunt "evidente" si în reprezentarea CNF. Asa putem defini primul algoritm simplu:

**Definitia 8.3** (**Brain-Dead Minesweeper**):

Fiind dat un afisaj *d*, se generează *CNF*(*d*).
if *CNF*(*d*) contine o clauză unitară pozitivă ($X_{i,j}$), se marchează $(i, j)$ ca mină
else if *CNF*(*d*) contine o clauză unitară negativă ($\neg X_{i,j}$), explorează pătratul sigur $(i, j)$
else explorează la întâmplare un pătrat necunoscut.

Se poate aprecia cât de bine lucrează această schemă: nu prea bine în mod special! Există multe cazuri în care schema nu operează bine; cele două exemple din lectia 7 ("trei de 1" si "cinci de 1") nu au miscări "evidente" dar au miscări logic solide. Apelând numai la inferențele evidente nu se obține o strategie completă:

[4] Scrierea $\lceil a \rceil$ se referă la întregul egal sau imediat superior numărului *a*.

**Definiția 8.4** (**Completitudinea** (**unei proceduri de inferență**)): O procedură demonstrativă este **completă** dacă si numai dacă ea poate dovedi fiecare propoziție care este consecința (entailed by) unei propoziții date.

**Algoritmi logici pentru Minesweeper**

Pentru a obtine un algoritm logic complet pentru Minesweeper se utilizează noțiunea de testare a posibilității-de-satisfacere din lectia 7. Reamintim că dacă $CNF(d) \wedge (\neg X_{i,j})$ este imposibil-de-satisfăcut, atunci $X_{i,j}$ este consecintă a lui $CNF(d)$.

**Definiția 8.5** (**Minesweeper logic**, **mark I**):

Fiind dat un afisaj $d$, se generează $CNF(d)$.
if $CNF(d)$ contine o clauză unitară pozitivă ($X_{i,j}$), se marchează $(i, j)$ ca mină
else if $CNF(d)$ contine o clauză unitară negativă ($\neg X_{i,j}$), se testează fără reținere pătratul $(i, j)$
else if $CNF(d) \wedge (\neg X_{i,j})$ este imposibil-de-satisfăcut pentru orice $X_{i,j}$ din $CNF(d)$, se marchează $(i, j)$ ca având mină
else if $CNF(d) \wedge (X_{i,j})$ este imposibil-de-satifăcut pentru orice $X_{i,j}$ din $CNF(d)$, se testează $(i, j)$ fără grijă pătratul $(i, j)$
else se testează un pătrat necunoscut la întâmplare.

De observat că algoritmul nu specifică mina care trebuie marcată mai întâi dacă se pot identifica mai multe mine, nici care pătrat să se testeze mai întâi dacă sunt mai multe pătrate detectate ca sigure. Este un exercitiu interesant a demonstra următoarea:

**Teorema 8.4**: Între două miscări aleatoare, algoritmul **mark I** marchează exact aceeasi multime de mine si descoperă aceeasi multime de pătrate sigure, indiferent de ordinea de selectie.

Astfel, **mark I** face în principiu fiecare miscare garantată logic. În teorie asta-i foarte frumos; în practică nu lucrează bine deloc. Am văzut deja că restricția globală poate avea exponențial de multe clauze. Mai mult, restricția globală este definită pe *toate* variabilele de pe tablă, asa încât pentru o tablă $X \times Y$ vor trebui enumerate $2^{XY}$ modele. Este o problemă fără speranță de mare!

Trebuie adoptată o strategie mai subtilă. Să împărtim $CNF(d)$ în restricțiile locale $C(d)$ si restricția globală $G(d)$. Mai întâi, am putea pretinde pur si simplu că restricția globală nu există:

**Definitia 8.6** (**Minesweeper logic, mark II**): Identică cu mark I cu deosebirea că $CNF(d)$ este înlocuită cu $C(d)$, forma CNF a restricțiilor locale.

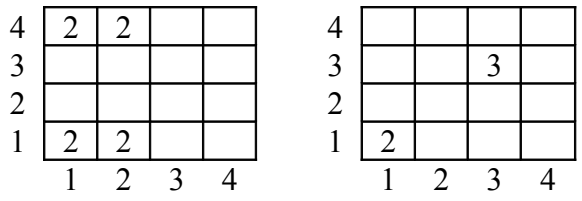
**Teorema 8.5**: Orice pătrat care este "sigur garantat" sau "minat garantat" în raport cu $C(d)$ este "sigur garantat" sau "minat garantat" si în raport cu $C(d) \wedge G(d)$.

Adică, miscările garantate **mark II** sunt corecte chiar dacă ele ignoră restricția globală! Este asta vreo proprietate specială, ciudată a jocului si, prin natura ei, a restrictiei globale? În realitate este numai un caz special al unei teoreme mult mai simple si mult mai puternice privitoare la **monotonia** logicii:

**Teorema 8.6**: Pentru orice propozitii $A$, $B$ si $C$, dacă $A \models C$ atunci $A \wedge B \models C$.
Demonstrația acestei teoreme este lăsată ca exercițiu. Este denumită *a monotoniei* deoarece pe măsură ce mulțimea faptelor cunoscute creste, mulțimea de concluzii consecutive (entailed) creste monoton; adăugând mai multe fapte cunoscute nu se poate *niciodată* invalida o concluzie stabilită anterior.
**Mark II** este mult mai eficientă decât **mark I**, deoarece variabilele din $C(d)$ sunt tocmai acele variabile necunoscute care sunt *adiacente* pătratelor cunoscute. Vom numi acestea **fringe** (tiv). Timpul de executie pentru **mark II** este $O(2^F)$ cu $F$ dimensiunea acestui *fringe*. Pătratele din background rămase vor fi numite **background**; sunt $B$ pătrate *background*.
**Mark II** joacă un rol destul de bun în jocul Minesweeper, dar uneori el trebuie să ghicească în cazurile când restricția globală are în realitate unele miscări garantate. Sunt aici două cazuri. Primul, restricția globală poate exclude unele modele ale restricțiilor locale, astfel încât miscările garantate pe *fringe* pot fi făcute pe baza modelelor rămase. Al doilea, restricțiile locale pot determina unele numere de mine pe *fringe* fixate, astfel încât *background*-ul trebuie să fie minat în totalitate (sau vid); aceasta permite miscări garantate în pătratele din *background* (cititorul este îndemnat să găsească exemple pentru aceste cazuri). Cum se poate încorpora restricția globală în algoritm? În esență prin adăugarea de verificări suplimentare în testul pentru posibilitatea-de-satisfacere, unde aceste verificări sunt implementate "extralogic" (adică *în afara* logicii formale) prin numărarea minelor din modele. Detaliile lui **mark III** sunt lăsate ca temă de casă.

**Importanța structurii problemelor**

Să considerăm problema din figura alăturată marcată cu (*a*). Orice examinare vizuală umană ar recunoaste instantaneu că realmente există aici două probleme separate, dacă se ingnoră restricția globală (este de asemenea clar că nu există nici o miscare logică pentru nici una din ele).




(*a*) (*b*)

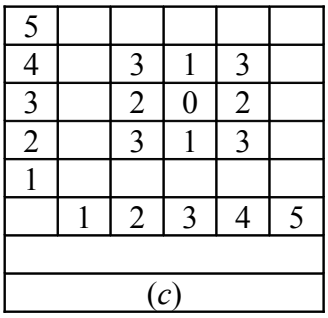



Figura 1: Descompunerea problemei. (*a*) Două probleme complet disjuncte. (*b*) Descompunere partială: dacă se cunoaste $X_{2,2}$, am putea avea două subprobleme disjuncte. (*c*) Aici, o posibilă multime de tăiere este $\{X_{2,1}, X_{3,1}, X_{4,1}, X_{2,5}, X_{3,5}, X_{4,5}\}$.

Cu toate acestea, algoritmul **mark II** generează o reprezentare CNF care include toate variabilele *fringe*, 12 în total, care dau $2^{12} = 4096$ de modele. Ar fi de dorit ca cele două probleme să poată fi rezolvate separat. Fiecare are şase variabile, astfel că s-ar putea ca pentru aceasta costul total să fie de cel mult $2^6 + 2^6 = 128$ de modele. Această reducere semnificativă este la îndemână tocmai pentru că funcția exponențială creste rapid cu cresterea numărului de componente. În fapt, dacă o problemă se poate diviza de la dimensiunea $n$ la $n/k$ bucăti de dimensiune egală $k$, costul total se modifică la $O(2^k n/k)$, ceea ce este liniar în $n$.

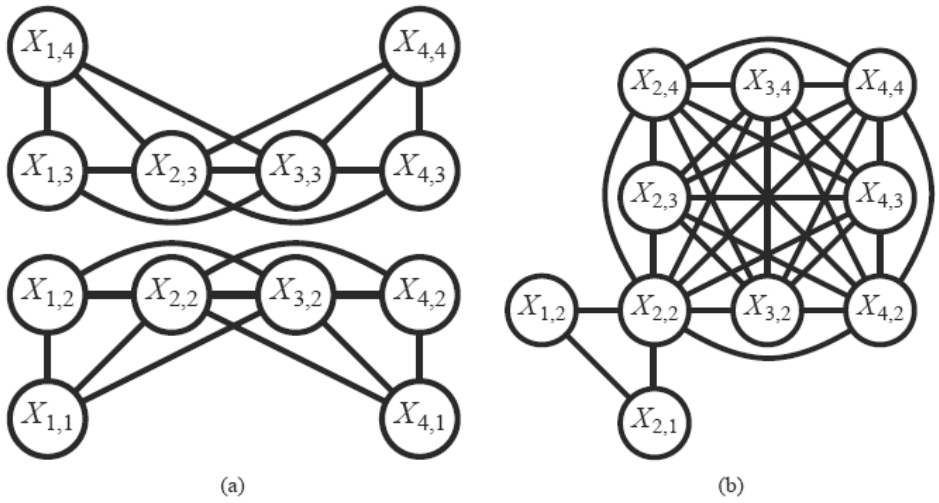
Acum trebuie defintă mai precis ideea de **descompunere a problemei**. Decompozabilitatea nu este ceva specific jocului Minesweeper; ea depinde de o anume structură specială din reprezentarea CNF pe care unele probleme de genul Minesweeper o generează. Dacă acea structură specială se poate identifica/recunoaste, atunci se poate efectua descompunerea *oricărei* probleme care poate fi reprezentată sub forma CNF.

Intuitiv, ideea este că variabilele se împart în două multimi $\{X_{1,4}, X_{1,3}, X_{2,3}, X_{3,3}, X_{4,3}, X_{4,4}\}$ şi $\{X_{1,1}, X_{1,2}, X_{2,2}, X_{3,2}, X_{4,2}, X_{4,1}\}$ astfel încât nici o variabilă dintr-o multime să nu aibă vreo "conexiune" cu vreo variabilă din cealaltă multime. În CNF, cele două submulțimi de clauze sunt "deconexate" dacă nu au variabile în comun.

**Definitia 8.7** (*Expresii deconexate*): Două expresii booleene $A$ si $B$ sunt deconexate dacă ele nu au variabile comune.

Graful din figura alăturată marcat cu (*a*) este o retea de noduri si arce care ilustrează conexitatea variabilelor pentru problema Minesweeper din figura de mai devreme marcată tot cu (*a*). Nodurile sunt legate în graf printr-un arc dacă ele apar împreună în vreo clauză. Deconexarea celor două multimi de noduri se vede cu usurintă în această reprezentare.

Acum e necesară o înțelegere a consecințelor deconexării pentru raționamentul logic. Deoarece posibilitatea-de-satisfacere este instrumentul canonic utilizat în raționament, să vedem ce se întâmplă.




(*a*) Graf care arată conectivitatea variabilelor din figura (*a*).
(*b*) Graful pentru figura (*b*)

**Lema 8.1**: Dacă două expresii booleene *A* si *B* sunt deconexate, atunci $A \wedge B$ este posibil-a-fi-satisfăcută dacă si numai dacă *A* este posibil-a-fi-satisfăcută si *B* este posibil-a-fi-satisfăcută.
**Demonstratie**: Fie $\mathbf{X}_A$ variabilele lui *A* si $\mathbf{X}_B$ variabilele lui *B*.
($\Rightarrow$ ): Se arată că dacă $A \wedge B$ este posibil-a-fi-satisfăcută, atunci *A* este posibil-a-fi-satisfăcută si *B* este posibil-a-fi-satisfăcută. Fie $M_{AB}$ un model pentru $A \wedge B$. Atunci *A* este adevărată în $M_{AB}$ si B este adevărată în $M_{AB}$. Fie $M_A$ submultimea din $M_{AB}$ care specifică variabilele din *A* si $M_B$ submultimea similară pentru *B*. Deoarece *A* si *B* nu au în comun vreo variabilă, *A* trebuie să fie adevărată în $M_A$ si *B* în $M_B$; asadar *A* si *B* sunt posibil-a-fi-satisfăcute.
($\Leftarrow$ ): Se arată că dacă *A* este posibil-a-fi-satisfăcută si *B* este posibil-a-fi-satisfăcută, atunci $A \wedge B$ este posibil-a-fi-satisfăcută. Fie $M_A$ un model pentru *A* si $M_B$ un model pentru *B*. Se defineste $M_{AB} = M_A \cup M_B$. Dacă *A* este adevărată în $M_A$ atunci este adevărată în $M_{AB}$; similar pentru *B*. Asadar $A \wedge B$ este adevărată în $M_{AB}$, astfel încât $A \wedge B$ este posibil-a-fi-satisfăcută.
□
Din lemă, urmează direct teorema care interesează:
**Teorema 8.7**: Fie *C* o expresie CNF; fie $C_1$ si $C_2$ expresii CNF astfel încât (1) $C_1$ si $C_2$ nu au variabile comune si (2) reuniunea clauzelor din $C_1$ si $C_2$ produce exact clauzele din *C*. Atunci, pentru orice propozitie *A*, $C \models A$ dacă si numai dacă $C_1 \models A$ sau $C_2 \models A$.
Acel "dacă si numai dacă" este aici foarte important. Implicatia $\Leftarrow$ este evidentă: rezultă simplu ca o aplicație a monotoniei. Ea se menține chiar dacă submulțimile *au* variabile comune! Astfel, orice pătrat care este garantat în ceea ce priveste orice submultime de restrictii este de asemenea garantat în raport cu toate restricțiile; astfel, o cale de a face eficientă investigarea completitudinii

constă chiar în împărţirea variabilelor în submulţimi mici si dovedirea oricăror garanţii posibile din fiecare submulţime (a se explora această idee ca temă de casă). Implicaţia ⇒ spune că dacă submulţimile sunt deconexate, se poate spune cu siguranţă că efectuarea verificărilor cu submulţimi *nu* duce la pierderea vreunei garanţii.

Exemplul care urmează, cel din figura (*b*), arată un caz în care variabilele nu pot fi separate în două mulţimi disjuncte. Graful din figura (*b*) se referă la acest caz. Aici tratarea prin simpla descompunere nu este de nici un ajutor si pare a fi cazul unei probleme cu un *fringe* de 10 variabile sau cu 1024 de modele.

Dar se poate utiliza următoarea strategie: dacă se cunoaste valoarea de adevăr a lui $X_{2,2}$, atunci acea variabilă va dispărea din toate clauzele care o conţin (înlocuită cu *T* sau *F*). Într-un astfel de caz, variabilele rămase se vor împărti în două mulţimi disjuncte. Asta este usor de văzut în figura (*b*): eliminarea nodului $X_{2,2}$ deconexează cele două jumătăţi. Planul constă asadar în a rezolva cele două jumătăti cu $X_{2,2} = T$ si cu $X_{2,2} = F$. Costul total va fi de cel mult $2(2^2 + 2^7) = 264$, substanţial mai mic decât 1024.

O mulţime de noduri a cărei eliminare împarte un graf în două sau mai multe componente neconectate este numită o **mulţime de tăiere**. Pentru problema în examinare, mulţimea de tăiere este $\{X_{2,2}\}$. Uneori nu este tocmai usor (trivial) a găsi o mulţime de tăiere bună; de exemplu, în figura de Minesweeper marcată cu (*c*), cea mai mică mulţime de tăiere are sase variabile – $\{X_{2,1}, X_{3,1}, X_{4,1}, X_{2,5}, X_{3,5}, X_{4,5}\}$. Acest fapt este cel mai bine confirmat prin examinarea vizuală a grafului problemei. Fără mulţimea de tăiere, problema are $2^{16} = 65.536$ de modele de verificat. Pentru variabilele din multimea de tăiere sunt 64 de asignări posibile si subproblemele rămase au fiecare câte 5 variabile asa încât costul total este de cel mult $64(2^5 + 2^5) = 4096$ adică de 16 ori mai putin si mai rapid decât în formularea iniţială, globală.

Această secţiune a dat unele idei privind importanţa structurii problemei si unele metode de a profita de structură. Clar, grafurile si algoritmii pentru grafuri au mult de-a face cu profitul posibil si de aceea, mai în profunzime vor face obiectul cursurilor următoare.

## Lecția 9

Secvența de lecții care urmează se referă la subiectul important al algoritmilor aritmetici. Se va dezvolta acest subiect până la întelegerea sistemului de criptare RSA cu cheie publică.

### Primalitate si factorizare

Să admitem că se dă un număr – fie acela 307131961967 – si se cere răspunsul la întrebarea: este prim sau nu? Nu este prima dată când (vi) se pune o astfel de întrebare. Cum se poate hotărî dacă un număr dat este prim?
Există, desigur, cel putin un algoritm pentru asta:

```
algorithm prime(x)
      y:=2
      repeat
            if x mod y = 0 then return(false)
            y:=y+1
      until y=x
      return(true)
```

Prin `x mod y` se întelege restul împărtirii întregi a lui $x$ prin $y$, cu $y$ nenul. Acest algoritm determină corect dacă numărul natural $x > 2$ este prim. El este implementarea definitiei unui număr prim prin explorarea exhaustivă a tuturor divizorilor posibili, de la 2 la $x - 1$. Dar acesta nu este un algoritm de prea mare utilitate: el ar fi nepractic până si pentru exemplul relativ modest propus, numărul cu 12 cifre de mai sus si, cum vom vedea, criptografia modernă cere ca numere cu câteva sute de cifre să fie testate pentru primalitate. Algoritmul necesită un număr de pasi proportional cu $x$ ceea ce nu este deloc bine.
Un alt algoritm mai rapid:

```
algorithm fasterprime(x)
      y:=2
      repeat
            if x mod y = 0 then return(false)
            y:=y+1
      until y*y≥x
      return(true)
```

Acum e ceva mai bine. Algoritmul acesta verifică toti divizorii posibili până la rădăcina pătrată a lui $x$. Si asta este suficient, deoarece dacă $x$ are un alt divizor decât 1 si el însusi, se ia în considerare cel mai mic dintre ei, fie acela $y$. Atunci, $x = y.z$, cu $z$ un întreg care este tot un divizor al lui $x$ diferit de 1 si de el însusi. Si întrucât $y$ este divizorul cel mai mic, $z \geq y$. Urmează că $y.y \leq y.z = x$ si $y$ nu

este mai mare decât rădăcina pătrată a lui $x$. Si al doilea algoritm caută, desigur, un astfel de $y$.
Acest algoritm este tot nesatisfăcător: într-un anume sens el este crescător "ca o exponentială" precum un algoritm exhaustiv pentru problema posibilitătii-de-satisfacere. Pentru a vedea de ce, trebuie să întelegem cum se evaluează timpul de calcul pentru algoritmi cu argumente care sunt numere naturale.
Sunt cunoscuti astfel de algoritmi, de pildă metodele de a aduna, de a multiplica, de a împărti numere întregi, învătate în scoala elementară. Pentru a aduna două numere, trebuie executate mai multe operatii elementare (adunarea a două cifre, retinerea transferului etc.) si numărul acestor operatii este proportional cu *numărul de cifre n* care alcătuiesc numerele de adunat: acest fapt se exprimă prin a spune că numărul de operatii este $O(n)$, pronuntat în englezeste "big-Oh of $n$", de ordinul lui $n$. Pentru a multiplica două numere este necesar un număr de operatii (consultarea tablei înmultirii, retinerea transferurilor etc.) care este proportional cu *pătratul* numărului de digiti, adică $O(n^2)$ (Cititorul să se asigure la acest punct că a înteles de ce este vorba aici de $n^2$; o mentiune: aici ne referim la multiplicarea "lungă", aceea care s-a învătat în clasele elementare; de fapt, un algoritm recursiv execută calculul într-un timp de cca. $O(n^{1,58})$; la zi, un algoritm complex atinge $O(n\log n\log\log n)$ care este putin mai lent decât dacă ar fi liniar în $n$). În contrast, primul algoritm pentru primalitate dat mai devreme ia un timp cel putin proportional cu $x$, care este de cca. $10^n$, unde $n$ este numărul de cifre din $x$; al doilea algoritm ia un timp cel putin proportional cu $10^{n/2}$, care este tot exponential în $n$.
Astfel, *în analizarea algoritmilor cu intrări numere întregi, este mult mai informativ a exprima durata calculelor ca functie de numărul de digiti ai intrării, nu de intrarea însăsi*. Nu importă dacă $n$ este numărul de biti ai intrării sau numărul de digiti zecimali. După cum se stie, aceste două numere sunt în relatia determinată de un factor mic si constant, în particular $\log_2 10 = 3{,}30\ldots$
Am ajuns la intrebarea cea mai importantă: există un algoritm de stabilire a primalitătii al cărui timp de executie să crească ca un polinom în numărul de digiti ai numărului de intrare? (cum sunt $n$, $n^2$, $n^3$ etc.?) Cum se va vedea, răspunsul este "da", un asemenea algoritm există. Acest algoritm are următoarea proprietate remarcabilă: determină dacă $x$ este prim sau nu fără a descoperi un factor al lui $x$ atunci când $x$ este compus (nu este prim). Cu alte cuvinte acest algoritm nu-l vom găsi continuând calea începută cu cei doi algortimi de mai devreme: nu este rezultatul unui mod inteligent de a examina din ce în ce mai putini divizori posibili ai lui $x$. Si este un motiv bun pentru care algoritmul rapid de stablire a primalitătii trebuie să fie ceva de genul: *nu există un algoritm polinomial cunoscut pentru descoperirea factorilor unui număr întreg*. Desigur, această problemă, cunoscută ca *problema factorizării*, este puternic suspectată de a fi dificilă, adică insolvabilă prin *vreun* algoritm al cărui timp de executie este polinomial (evident, în $n$). Această combinatie de fapte matematice sună a fi practic imposibilă, dar reprezintă un adevăr: factorizarea este dificilă, detectarea primalitătii este facilă! De fapt, se va vedea, criptografia modernă este bazată pe această distinctie subtilă dar puternică.

Pentru a întelege algoritmul pentru primalitate, factorizare si criptografie, trebuie mai întâi să aducem în discutie încă vreo câtiva algoritmi pentru numere naturale.

**Calculul celui mai mare divizor comun (cmmdc - gcd)**

Cel mai mare divizor comun al numerelor naturale $x$ si $y$, notat gcd($x$, $y$) este cel mai mare număr natural care le divide (exact) pe ambele (de retinut: zero nu divide nici un număr, dar este divizat de orice număr natural). Cum se calculează un gcd? Prin *algoritmul lui Euclid*, probabil primul algoritm inventat vreodată.

```
algoritm gcd(x,y)
      if y = 0 then return(x)
      else return(gcd(y,x mod y))
```

Putem exprima acelasi algoritm în ceva mai elegantul limbaj Scheme:

```
(define (gcd x y)
      (if (= y 0)
            x
            (gcd y (remainder x y)) ) )
```

Acest algoritm presupune că $x \geq y \geq 0$ si $x > 0$.

**Teorema 9.1**: Algoritmul de mai sus calculează corect gcd pentru $x$ si $y$ într-un timp $O(n)$, unde $n$ este numărul de biti în intrarea ($x$, $y$).

**Demonstratie**: Corectitudinea este demonstrată prin inductie (tare) după $y$, cel mai mic dintre cele două numere. Pentru fiecare $y \geq 0$, fie $P(y)$ propozitia conform căreia algoritmul calculează corect gcd($x$, $y$) pentru toate valorile lui $x \geq y$ (si $x > 0$). Certamente, $P(0)$ se verifică deoarece gcd($x$, 0) = $x$ si algoritmul calculează corect prin clauza "if". Pentru pasul inductiv, se poate presupune că $P(z)$ se verifică pentru orice $z < y$ (ipoteza inductivă); trebuie demonstrat că are loc $P(y)$. Observatia cheie aici este că gcd($x$, $y$) = gcd($y$, $x$ mod $y$) – adică, înlocuirea $x$ cu $x$ mod $y$ nu schimbă cel mai mare divizor comun (gcd – *greatest common divisor*). Asta se întâmplă deoarece un divizor $d$ al lui $y$ divide si pe $x$ dacă si numai dacă el divide pe $x$ mod $y$ (divizibilitatea prin $d$ nu este afectată prin adunarea sau scăderea unui multiplu de $d$ si $y$ este multiplu de $d$). Asadar, clauza "else" a algoritmului va returna corect valoarea furnizată de apelul recursiv gcd($y$, $x$ mod $y$) care calculează valori corecte. Dar deoarece $x$ mod $y < y$, stim că aceasta este un adevăr, din ipoteza inductivă. Verificarea lui $P(y)$ este acum completă si completă este si demonstratia inductivă.

Acum trecem la investigarea limitei $O(n)$ relativ la timpul de calcul. Este evident că argumentele apelurilor recursive devin din ce în ce mai mici ca număr de digiti (deoarece $y \leq x$ si $x$ mod $y < y$). Problema este cât de repede scade lungimea acestor argumente. Vom arăta că în calculul lui gcd($x$, $y$), după două apeluri recursive primul (cel mai mare) argument este mai scurt decât $x$ prin cel puntin un factor de 2 (presupunând $x > 0$). Sunt două cazuri de parcurs:

- $y \leq x/2$. Atunci primul argument y în apelul recursiv următor este deja mai mic decât x cu un factor de 2 si astfel în apelul recursiv următor va fi chiar mai mic;
- $x \geq y > x/2$. Atunci în două apeluri recursive primul argument va fi $x$ mod $y$, care este mai mic decât $x/2$.

Asadar, în ambele cazuri primul argument descreste printr-un factor de cel putin 2 la fiecare apel recursiv. Astfel, după cel mult $2n$ apeluri recursive, unde $n$ este numărul de biti în $x$, recursia se va opri (de notat că de fiecare dată primul argument este un număr natural)

□

De observat că partea a doua a demonstratiei de mai sus arată numai că numărul de apeluri recursive în timpul calculului este $O(n)$. Putem face aceeasi afirmatie pentru durata calculului dacă presupunem că fiecare apel necesită o durată constantă. Deoarece fiecare apel include o comparatie de întregi si o operatie mod(ulo) pare rational a considera durata aceasta constantă. Într-un model de calcul mai realist trebuie totusi să facem acest timp dependent de dimensiunile numerelor implicate: astfel comparatia ar necesita $O(n)$ operatii elementare (pe biti) si operatia mod, care este în fapt o împărtire, va necesita $O(n^2)$ operatii pentru un total de $O(n^2)$ operatii în fiecare apel recursiv (Aici $n$ este numărul maxim de biti în $x$ si $y$, care este exact numărul de biti în $x$). Astfel, într-un asemenea model, durata calculelor cu algoritmul lui Euclid este în realitate $O(n^3)$.

**Aritmetica modulară**

Un exemplu: calculul zilei din săptămână. Se presupune că secventa zilelor săptămânii (luni, marti, miercuri, joi, vineri, sâmbătă, duminică) este aplicată pe secventa de numere (0, 1, 2, 3, 4, 5, 6) astfel încât luni este 0, marti este 1 etc. În cazul că azi este joi = 3, dacă e de calculat ce zi a săptămânii va fi peste 10 zile, intuitiv am răspunde cu restul împărtirii la 7 a lui 3 + 10 = 13, care este 6, adică duminică. De fapt, pare a fi cam fără sens în acest context a aduna un număr ca 10 si mai la îndemână ar putea fi să găsim restul *lui* modulo 7, si anume 3, apoi să adunăm acest 3 la 3 pentru a găsi 6.

Dar dacă se continuă cu alte salturi de 10 zile? După 5 asemenea salturi am găsi 3 + 3*5 = 18, care dă 4 mod 7, adică vineri.

Acest exemplu arată că în unele împrejurări are sens să lucrăm într-o artimetică limitată, restrânsă de un număr anumit (aici 7), adică să facem aritmetică prin stabilirea pentru fiecare număr, numărul modulo 7 să spunem, si să repetăm aceasta pentru rezultate s.a.m.d. În afară de eficienta dată de mentinerea rezultatelor si valorilor intermediare la un nivel valoric scăzut, această tratare are aplicatii importante în criptografie, cum se va vedea mai departe.

Se pot scrie ecuatii în acest gen de aritmetică. Ecuatiile sunt scrise, de exemplu, sub forma

$$4 + 10*5 = 40 \bmod 7$$

(În literatură se scrie uneori ≡ în loc de = si partea cu mod $m$ se pune între paranteze). Aceste ecuatii se pot citi în două moduri. Unul constă în luarea în considerarea a operatiilor aritmetice obisnuite care se încheie de fiecare dată cu luarea valorii "mod $m$" atât pentru partea stângă cât si pentru partea dreaptă, înainte de verificare.
Modul celălalt este ceva mai sofisticat. Dacă $m$ este un întreg (de pildă 7), se defineste o relatie între întregi: numerele $x$ si $y$ se zice că sunt *congruente modulo m*, scris $x = y \bmod m$, dacă si numai dacă ele diferă printr-un multiplu al lui $m$, adică $x - y = k.m$ pentru un $k$ întreg (posibil negativ). Într-o formulare diferită, $x$ si $y$ sunt congruente modulo $m$ dacă au acelasi rest modulo $m$. Relatia "congruent modulo $m$" este o relatie de echivalentă: ea partitionează multimea întregilor în $m$ clase de echivalentă 0, 1, 2, …, $m - 1$.
**Teorema 9.2**. Dacă $a = c \bmod m$ si $b = d \bmod m$ atunci $a + b = c + d \bmod m$ si $a.b = c.d \bmod m$.
**Demonstratie**. Se stie că $c = a + k.m$ si $d = b + l.m$, asa că $c + d = a + b + (k + l).m$, ceea ce însemană $a + b = c + d \bmod m$. Demonstratia pentru produs este similară (exercitiu).
□
Ce spune această teoremă este că orice expresie aritmetică modulo $m$ poate fi totdeauna redusă la un număr natural mai mic decât $m$. Ca exemplu, fie expresia $(13 + 11).18 \bmod 7$. Uzând de teorema 9.2 de mai multe ori se poate scrie succesiv:

$$\begin{aligned}(13 + 11).18 &= (6 + 4).4 \bmod 7\\ &= 10.4 \bmod 7\\ &= 3.4 \bmod 7\\ &= 12 \bmod 7\\ &= 5 \bmod 7\end{aligned}$$

Recapitulând, se pot face totdeauna calcule modulo $m$ prin reducerea modulo $m$ a rezultatelor intermediare.

**Exponentierea**

O operatie standard în algoritmii aritmetici (inclusiv verificarea primalitătii si criptarea RSA) este ridicarea unui număr la o putere modulo un alt număr. Cu alte cuvinte, cum se calculează $x^y \bmod m$ când $x$, $y$, $m$ sunt numere naturale si $m > 0$? O tratare naivă ar consta în a calcula secventa $x \bmod m$, $x^2 \bmod m$, $x^3 \bmod m$, …, până la termenul al $y$-lea, dar asta ar consuma un timp exponential în numărul de biti în $y$. Aceeasi treabă se poate face mai bine dacă se utilizează "trucul" *ridicării la pătrat repetate*.

```
algorithm mod-exp(x, y, m)
      if y = 0 then return(1)
      else
            z = mod-exp(x, y div 2, m)
            if y mod 2 = 0 then return(z*z mod m)
            else return(x*z*z mod m)
```

Acest algoritm utilizează faptul că orice $y > 0$ poate fi scris fie ca $2a$, fie ca $2a + 1$, cu $a = \left\lfloor \frac{y}{2} \right\rfloor$ scris în pseudocodul de mai sus ca `y div 2`, plus relatiile cunoscute

$$x^{2a} = (x^a)^2 \quad \text{si} \quad x^{2a+1} = x.(x^a)^2.$$

Ca exercitiu, se pot utiliza aceste adevăruri pentru a construi o argumentare formală inductivă a faptului că algoritmul returnează totdeauna valoarea corectă.

Care este timpul de calcul? Ca de obicei pentru algoritmii recursivi si aici task-ul principal este a realiza câte apeluri recursive se comit. Se poate vedea că argumentul al doilea, $y$, este împărtit (întreg) prin 2 la fiecare apel astfel că numărul de apeluri recursive este exact cât numărul de biti, $n$, în $y$. Faptul este valabil (până la un factor constant minor) si în cazul când $n$ este numărul de digiti zecimali în $y$. Asadar, dacă luăm un consum de timp constant pentru fiecare operatie aritmetică (div, mod etc.) timpul de calcul pentru `mod-exp` este $O(n)$.

Într-un model mai realist, se poate evalua mai cu grijă durata fiecărui apel recursiv. Mai întâi, testul asupra lui $y$ în declaratia "if" implică numai o privire la bitul cel mai putin semnificativ din $y$ si calculul lui $\lfloor y/2 \rfloor$ este numai o deplasare (shift) în reprezentarea binară. De aceea aceste operatii iau un timp constant. Costul fiecărui apel recursiv este asadar dominat de operatia mod în rezultatul final care ia un timp $O(n^2)$ cu $n$ numărul maxim de biti în $x$ sau în $y$. Astfel, cu modelul mai realist durata executiei lui mod-exp pe numere de $n$ biti este $O(n^3)$.

**Inverse**

Un alt ingredient al algoritmilor aitmetici îl constituie inversele multiplicative modulo un număr $m$. *Inversul multiplicativ* al unui întreg pozitiv $x$ modulo $m$ este un număr $a$ astfel încât $ax = 1 \bmod m$. Dacă se lucrează cu numere rationale, atunci $a = 1/x$ este unicul invers al lui $x$. În aritmetica modulară, putem avea siguranta că $x$ are un invers mod $m$ si dacă are, este acesta unic (modulo $m$) si îl putem calcula?

Ca un prim exemplu, să luăm $x = 8$ si $m = 15$. Se găseste $2x = 16 = 1 \bmod 15$, asadar 2 este inversul multiplicativ al lui 8 mod 15. Ca un al doilea exemplu, să luăm $x = 12$ si $m = 15$. Secventa $\{ax \bmod m: a = 0, 1, 2, \ldots\}$ este periodică si ia valorile $\{0, 12, 9, 6, 3\}$ (de verificat!). Asadar, 12 *nu are* un invers multiplicativ mod 15.

Se pune întrebarea naturală: în ce conditii $x$ are un invers multiplicativ modulo $m$? Răspunsul: dacă si numai dacă $\gcd(m, x) = 1$. Asta înseamnă că $m$ si $x$ nu au nici un factor comun în afară de 1, ceea ce se exprimă spunând că $m$ si $x$ sunt *mutual prime*. Mai mult, când inversul există el este unic.

**Teorema 9.3**: Fie $m$, $x$ întregi pozitivi astfel încât gcd($m$, $x$) = 1. Numărul $x$ are în aceste conditii un invers multiplicativ modulo $m$, care este si unic (modulo $m$).

**Demonstratie**: Se consideră secventa de $m$ numere 0, $x$, $2x$, …, $(m-1)x$. Vom afirma că toate acestea sunt distincte modulo $m$. Deoarece sunt numai $m$ valori distincte modulo $m$, trebuie să existe un caz pentru care $ax = 1 \bmod m$ pentru exact un $a$ (modulo $m$). Acest $a$ este inversul multiplicativ unic.

Pentru a verifica afirmatia de mai sus, se presupune că $ax = bx \bmod m$ pentru două valori distincte $a$, $b$ în gama $0 \le a, b \le m - 1$. Dacă-i asa, atunci $(a - b)x = 0 \bmod m$ sau, echivalent, $(a - b)x = km$ pentru un $k$ (posibil nul sau negativ). Dar pentru că $m$ si $x$ sunt mutual (relativ) prime, urmează că $a - b$ trebuie să fie un multiplu întreg de $m$. Acest fapt nu-i posibil deoarece $a$, $b$ sunt numere distincte, nenegative, mai mici decât $m$.

□

În realitate, se dovedeste că gcd($m$, $x$) = 1 este totodată si conditia necesară de existentă a unui invers: dacă gcd($m$, $x$) > 1 atunci $x$ nu are un invers multiplicativ modulo $m$. Pentru demonstarea acestui aspect s-ar putea utiliza o idee similară celei din demonstratia de mai sus.

Deoarece unicitatea inversului multiplicativ când gcd($m$, $x$) = 1 este dovedită, vom scrie inversul lui $x$ ca $x^{-1} \bmod m$. Dar cum se calculează $x^{-1}$ când se dau $x$ si $m$? Să ajungem acolo pe o rută întrucâtva ocolitoare.

Pentru orice pereche de numere $x$, $y$, se admite că se poate calcula gcd($x$, $y$) dar se pot găsi si doi întregi $a$, $b$ astfel ca

$$d = \gcd(x, y) = ax + by$$

(ecuatia aceasta nu este una modulară; întregii $a$, $b$ pot fi 0 sau negativi). De exemplu, se poate scrie

$$1 = \gcd(35, 12) = -1 \cdot 35 + 3 \cdot 12$$

cu $a = -1$ si $b = 3$ valori posibile pentru $a$ si $b$.

Dacă putem face aceasta, atunci vom putea calcula inversul după cum urmează. Se aplică procedura de mai sus pentru numerele $m$, $x$ ceea ce produce întregii $a$, $b$ astfel încât

$$1 = \gcd(m, x) = am + bx$$

Dar asta înseamnă că $bx = 1 \bmod m$, astfel $b$ este un invers multiplicativ al lui $x$ modulo $m$. Reducerea $b$ modulo $m$ dă inversul unic pe care îl căutăm. În exemplul de mai devreme, se observă că 3 este inversul multiplicativ al lui 12 mod 35.

Astfel am redus problema calculului inverselor la aceea de a stabili întregii $a$ si $b$ care satisfac ecuatia de mai sus. Acum, deoarece această problemă este o generalizare a celui mai mare divizor comun de bază, nu este deloc surprinzător că o putem rezolva ca o extensie destul de simplă a algoritmului lui Euclid. Algoritmul care urmează *extended-gcd* are ca intrări o pereche de numere naturale $x \ge y$ ca în algoritmul lui Euclid si returnează trei întregi ($d$, $a$, $b$) astfel încât $d = \gcd(x, y)$ si $d = ax + by$.

```
algorithm extended-gcd
      if y = 0 then return (x, 1, 0)
```

```
else
      (d, a, b) := extended-gcd(y, x mod y)
      return((d, b, a - (x div y)*b))
```

De observat că acest algoritm are aceeasi formă ca algoritmul `gcd` de bază discutat mai devreme; singura diferentă este că acum transferăm de la un pas la altul si numerele $a$, $b$. Se poate parcurge de mână algoritmul pentru intrarea ($x$, $y$) = (35, 12) din exemplul numeric de mai sus si se poate verifica corectitudinea rezultatului dat de algoritm pentru $a$, $b$.

Să vedem acum de ce algoritmul lucrează. În cazul de bază ($y = 0$) este returnat valoarea c.m.m.d.c. $d = x$ ca si înainte, împreună cu valorile $a = 1$ si $b = 0$ care satisfac ecuatia $ax + by = d$. Dacă $y > 0$ se calculează mai întâi recursiv valorile ($d$, $a$, $b$) astfel ca

$$d = \gcd(y, x \bmod y) \text{ si } d = ay + b(x \bmod y)$$

Ca si în analiza algoritmului vanilla, cunoastem că acest $d$ va fi egal cu gcd($x$, $y$). Astfel prima componentă a tripletului returnat de algoritm este corectă.

Dar celelalte două componente? Să le spunem $A$ si $B$. Care vor fi valorile lor? Ei bine, din specificarea algoritmului ele trebuie să satisfacă relatia

$$d = Ax + By$$

Pentru a realiza cât trebuie să fie $A$ si $B$, trebuie rearanjată o ecuatie de mai sus după cum urmează:

$$d = ay + b(x \bmod y) = ay + b(x - \lfloor x/y \rfloor y) = bx + (a - \lfloor x/y \rfloor b)y$$

În faza a doua, s-a utilizat faptul că $x \bmod y = x - \lfloor x/y \rfloor y$ (de verificat!). Prin compararea ultimei variante cu ecuatia de imediat mai sus se vede că trebuie luat $A = b$ si $B = a - \lfloor x/y \rfloor b$. Este exact ceea ce face algoritmul si aceasta conchide verificarea de corectitudine pe care am întreprins-o.

Deoarece algoritmul extins al celui mai mare divizor comun, `extended-gcd`, are exact aceasi structură recursivă ca si versiunea vanilla, durata lui de lucru va fi aceeasi până la un factor constant (care reflectă timpul crescut pentru un apel recursiv). Astfel, încă o dată durata de calcul pe numere de $n$ biti va fi $O(n)$ operatii aritmetice si $O(n^3)$ operatii pe biti. Prin combinarea acestui adevăr cu discutia de mai devreme asupra inverselor, se vede că pentru orice $x$, $m$ cu $\gcd(m, x) = 1$ se poate calcula $x^{-1} \bmod m$ în aceleasi limite de timp.

# Lecția 10

## Primalitate

Studiem acum complexitatea a două probleme de calcul foarte importante si foarte legate una de alta.
PRIMALITATEA: Fiind dat un întreg, este el prim?
FACTORIZAREA: Fiind dat un întreg, care sunt factorii primi ai acestuia?
Evident, calitatea de a fi prim a unui număr întreg (primalitatea) nu poate fi mai dificil de stabilit decât factorizarea lui, deoarece dacă se cunoaste cum se stabileste un factor, ar trebui hotărît să se cunoască si cum se face testarea primalității. Ceea ce este surprinzător si fundamental – si baza criptografiei moderne – este că primalitatea este *usoară* în timp ce factorizarea este *dificilă*!
Cum se stie, primalitatea poate fi rezolvată trivial într-un timp $O(x)$ – de fapt sunt de testat numai factorii până la $\sqrt{x}$. Dar, e limpede, ambii acesti algoritmi sunt exponentiali – exponentiali în numărul $n$ de biti ai lui $x$, o măsură mai precisă si mai plină de sens a dimensiunii problemei (văzut asa, timpul de rulare a algoritmului devine $O(2^n)$ si $O(2^{n/2})$, respectiv). În fapt, urmarea acestei linii (de testare a din ce în ce mai putini factori) nu duce nicăieri: deoarece factorizarea este dificilă, singura sperantă de a găsi un algoritm rapid pentru primalitate este de a căuta unul care să decidă dacă un număr $n$ este prim fără a descoperi un factor al lui $n$ în cazul în care răspunsul este negativ.
Imediat, este descris un astfel de algoritm. Acest algoritm se bazează pe faptul următor relativ la exponentierea modulo un-număr-prim.
**Teorema 10.1** (**Mica teoremă a lui Fermat**): Dacă $p$ este număr prim, atunci pentru orice $a \neq 0 \bmod p$ are loc $a^{p-1} = 1 \bmod p$.
**Demonstratie**: Se consideră toate numerele nenule modulo $p$, $\Phi = \{1, 2, \ldots, p-1\}$. Acum, dacă se ia un $a$ din această multime si dacă se multiplică toate aceste numere prin $a$ modulo $p$, se obtine o altă multime $\Phi_a = \{a.1, a.2, \ldots, a.(p-1)\}$, toate mod $p$. Afirmăm că toate cele $p-1$ numere din $\Phi_a$ sunt distincte si în consecintă $\Phi_a = \Phi$. Ca demonstratie, dacă $a.i = a.j \bmod p$, atunci, prin multiplicarea ambilor membri cu $a^{-1} \bmod p$ (deoarece $p$ este prim si $a \neq 0 \bmod p$ se stie că $a$ are un invers) se obtine $i = j$. Prin urmare, produsele $\prod_{x \in \Phi} x$ si $\prod_{x \in \Phi_a} x$ sunt egale, adică

$$(a.1).(a.2)\ldots.(a.(p-1)) = 1.2\ldots.(p-1) \bmod p$$

Acum, prin multiplicarea egalității cu $1^{-1} \bmod p$, apoi cu $2^{-1} \bmod p$ s.a.m.d. până la $(p-1)^{-1} \bmod p$ se obtine teorema.
□

Teorema 10.1 sugerează un test de primalitate pentru $p$: se ia un număr $a \neq 0 \bmod p$ si se ridică la puterea a $(p - 1)$ modulo $p$. Dacă rezultatul nu este 1, atunci stim că $p$ nu este prim. Dar dacă $a^{p-1} = 1 \bmod p$? Putem fi siguri că $p$ este prim? Nu! Vor exista totdeauna numere $a$ care satisfac ecuatia (1 si $p - 1$ sunt două exemple foarte la îndemână). Reciproca teoremei 10.1 nu este adevărată. Tot ce se poate dovedi este că:

**Teorema 10.2**: Dacă $x$ nu este prim si dacă $x$ nu este un număr Carmichael atunci pentru cele mai multe numere $a \neq 0 \bmod x$, $a^{x-1} \neq 1 \bmod x$.

*Paranteză*: un număr $c$ este un număr Carmichael dacă nu este prim si încă pentru toti divizorii primi ai lui $c$ cu $d > 1$ se întâmplă astfel că $d - 1$ să dividă pe $c - 1$. Cel mai mic număr Carmichael este 561 = 3.11.17 (se observă, desigur, că 3 – 1, 11 – 1 si 17 – 1 divid fiecare pe 561 – 1). Dacă $c$ este un număr Carmichael si $a$ este relativ prim cu $c$, atunci $a^{c-1} = 1 \bmod c$. Puteti demonstra asta? *Final de paranteză*.

Teorema 10.2 (care ar fi un pic de deturnare a fi demonstrată acum) este o reciprocă slabă a teoremei 10.1: ea spune că dacă $x$ este compus si dacă $x$ se întâmplă a fi în multimea exceptiilor extrem de rare numite *numere Carmichael*, atunci testul de primalitate sugerat de mica teoremă a lui Fermat va expune desigur faptul că $x$ nu este prim cu probabilitatea de 50%.

Discutia de mai sus sugerează următorul *algoritm randomizat* pentru primalitate:

```
algorithm prime(x)
repeat K times:
      pick an integer a between 1 and x at random
      if a^(x-1) ≠ 1 mod x then return ("x is not a prime")
return ("with probability at least 1 – 2^(–K),
      x is either a prime or a Carmichael number")
```

Probabilitatea de corectitudine reclamată este usor de probat: dacă $x$ nu este nici prim si nici număr Carmichael, atunci teorema 10.2 spune că fiecare exponentiere va expune aceasta cu probabilitate de cel putin 0,5. Deoarece toate aceste $K$ încercări sunt independente (adică se alege de fiecare dată $a$ fără a interesa valorile $a$ anterioare), probabilitatea ca toti $K$ să esueze în a expune $x$ este $2^{-K}$. Luând $K = 100$ (si reamintind că numerele Carmichael sunt extrem de rare), se stabileste primalitatea la un grad de încredere (0,999… până la treizeci si nouă) care depăseste toate celelalte aspecte ale vietii si calculelor. Pentru a face un test se consumă $O(K.n^3)$ pasi, unde $n$ este numărul de biti ai lui $x$, deoarece el constă în $K$ exponentieri.

Incidental, există un algoritm aleator de detectare de numere Carmichael, întrucâtva mai elaborat, astfel există un algoritm aleator polinomial pentru primalitate. Acel algoritm mai elaborat nu este discutat în acest curs.

Numerele prime nu sunt numai usor de detectat dar sunt si relativ abundente:

**Teorema 10.3** (*Teorema numerelor prime*): Numărul de numere prime între 1 si $x$ este de circa $x/\ln x$ (Cu alte cuvinte, dacă se ia un număr aleator cu $D$ digiti

zecimali, sansa ca el să fie prim este putin mai mică de 1 la 2*D* (Una din circa 26 de persoane are codul numeric personal număr prim).
Teorema numerelor prime este un fapt dintre cele fundamentale în matematici si este foarte greu de demonstrat. Dar împreună cu algoritmul de mai devreme, ea ne abilitează să găsim usor numere prime mari (încercati câteva si retineti unul care verifică conditia de primalitate) si acesta este ingredientul cheie al algoritmului RSA. Un alt ingredient este varianta următoare a teoremei 10.1:
**Teorema 10.4**: Dacă $p$ si $q$ sunt prime, atunci pentru orice $a \neq 0 \bmod p.q$ avem $a^{(p-1)(q-1)} = 1 \bmod p.q$.
Demonstratia teoremei 10.4 este aceeasi ca a teoremei 10.1 cu exceptia faptului că se consideră multimea tuturor numerelor 1, 2, …, $p.q - 1$ care sunt relativ prime cu $p.q$. Se observă că există $(p - 1)(q - 1)$ astfel de numere – ca verificare, unul din fiecare $p$ numere între 0 si $p.q$ este divizibil cu $p$ si unul din fiecare $q$ prin $q$ si $pq(1 - 1/p)(1 - 1/q) = (p - 1)(q - 1)$.
Exemplu. Fie $p = 3$ si $q = 5$. Dintre toate numerele modulo $p.q = 15$, si anume {0, 1, 2, …, 14}, o treime sunt divizibile cu 3 ({0, 3, 6, 9, 12}) si din cele 10 rămase o cincime sunt divizibile cu 5 ({5, 10}). Toate numerele rămase în număr de $(p - 1)(q - 1) = 8$ ($\Phi$ = {1, 2, 4, 7, 8, 11, 13, 14}) sunt prime cu 15 si prin urmare ele au toate un invers modulo 15. Aceasta face posibilă demonstrarea teoremei 10.1 prin luarea lui $a$ oricare dintre aceste 8 numere.

# Lecția 11

## Criptografie si RSA

Criptografia se ocupă de scenarii de genul următor: două persoane (Alice si Bob) doresc să comunice în prezenta unei alte persoane deosebit de curioase (Eve). Se presupune că Alice vrea să trimită lui Bob mesajul $x$. Criptografia oferă solutii în forma următoare: Alice calculează o functie de $x$, $e(x)$, utilizând o cheie secretă, si trimite $e(x)$ pe canalul ascultat de Eve. Bob primeste $e(x)$ si utilizând cheia sa (care în criptografia traditională este aceeasi cu cheia Alicei, dar în criptografia modernă nu este aceeasi) calculează o functie $d(e(x)) = x$, recuperând astfel mesajul. Se prezumă că Eve este incapabilă să recupereze $x$ din $e(x)$ deoarece ea nu are cheia.

O metodă criptografică clasică este *substituirea de litere*. Alice si Bob au căzut de acord asupra unei permutări $\pi$ de litere $A$, …, $Z$ si un mesaj $m = a_1a_2…a_n$ alcătuit din $n$ litere devine $e(m) = \pi(a_1)\pi(a_2)…\pi(a_n)$. În pofida faptului că există 26! chei posibile, acesta este un sistem de criptare foarte slab, expus la *atacul* evident prin *frecventa literelor*.

Cu concursul calculatoarelor, criptografii au posibilitatea de a crea sisteme de criptare în care sunt substituite *blocuri* întregi *de litere* (sau biti), rezistente astfel la atacul prin frecvente. Cel mai de succes dintre ele este Data Encryption Standard (DES), o metodă de criptare sponsorizată de guvernul american, propusă în 1976. DES utilizează o cheie cu 64 de biti (pentru codarea fisierelor mai lungi, acestea se sparg în blocuri de 8 biti). Cei care au propus DES pretind că este un cod sigur; sunt si detractori care-l suspectează că la modul subtil nu este atât de sigur. Dimensiunea cheii de 56 este în general considerată inadecvată pentru criptografia serioasă, deoarece $2^{56}$ nu mai este un număr mare.

Cam la acelasi moment când DES a fost inventat, a fost propusă o idee nouă foarte atractivă: criptografie cu cheie publică (*public-key cryptography*). Bob ar avea două chei, cheia sa privată $k_d$, cunoscută numai de el, si cheia lui publică, $k_c$. (Poate că este mai potrivit a considera $k_c$ ca un lacăt si $k_d$ ca o cheie a lacătului.). Cheia publică $k_c$ este cunoscută tuturor – este pe pagina gazdă a lui Bob, de pildă. (Continuând cu metafora dubioasă cu care am început, este ca si cum ar exista mai multe cópii ale lacătului, cu care oamenii pot închide cutii care contin mesaje si sunt trimise lui Bob.) Dacă cineva, de pilă Alice, doreste să trimită un mesaj $x$ lui Bob, atunci îl criptează sub forma $e(x)$ care utilizează cheia publică. Bob îl decriptează folosindu-se de cheia lui privată. Partea inteligentă este că: (1) decriptarea este corectă, adică $d(e(x)) = x$; (2) nu se poate

calcula fezabil cheia privată a lui Bob din cheia lui publică sau $x$ din $e(x)$, astfel că protocolul este sigur (exact cum nu există o cale de a imagina cheia pentru un lacăt bun).

RSA – de la intialele a trei cercetători (Ron Rivest, Adi Shamir si Len Adleman) care l-au inventat – este un mod inteligent de a realiza ideea de cheie publică. Iată cum lucrează aceasta:

- *Generarea cheii*. Bob stabileste două numere prime mari, $p$ si $q$. ("Mare" înseamnă în zilele noastre câteva sute de digiti zecimali, curând va însemna poate peste o mie). Pentru a găsi asemenea numere, Bob generează repetat intregi din această gamă si îi supune testului Fermat până când două din ele trec testul. Apoi Bob calculează $n = p.q$. De asemenea, Bob calculează la întâmplare un întreg $e < n$, cu singura restrictie de a fi prim cu $(p - 1)$ si cu $(q - 1)$. Perechea $(n, e)$ este acum *cheia publică* a lui Bob si Bob o face cunoscută. (Practic, pentru a face codarea mai usoară (a se vdea mai jos), $e$ este adesea luat 3. Desigur, trebuie evitate numerele prime care sunt 1 mod 3.)

  Acum Bob poate genera cheia lui secretă. Tot ce are de făcut este a calcula (prin algoritmul lui Euclid) $d = e^{-1} \ mod \ (p - 1)(q - 1)$. Perechea $(n, d)$ este *cheia privată* a lui Bob.
- *Operarea*. Ori de câte ori Alice sau oricine altcineva doreste să-i trimită un mesaj lui Bob procedează astfel:
  - Fragmentează mesajul în siruri de biti de lungime $\lfloor \log n \rfloor$. (reamintim, este vorba de mai multe sute de biti). Se codează fiecare sir de biti prin algoritmul care urmează.
  - Fie $x$ un astfel de sir de biti si se consideră ca fiind un întreg mod $n$. Alice calculează $x^e$ mod $n$. Acesta aste mesajul codat $e(x)$ pe care Alice îl trimite lui Bob.
  - Bob, la primirea lui $e(x)$, calculează $e(x)^d$ mod $n$. Putină algebră (si reutilizând mica teoremă a lui Fermat pentru produsul a două numere prime prezentată în sectiunea precedentă) produce:

$$e(x)^d = x^{d.e} = x^{1 + m(p-1)(q-1)} = x \bmod n$$

De observat că această secventă de ecuatii stabileste faptul că Bob decodează corect. Prima ecuatie reaminteste de definitia lui $e(n)$. A doua decurge din faptul că $d$ este inversul lui $e$ mod $(p - 1)(q - 1)$ si astfel dacă se multiplică $d$ cu $e$ se obtine 1 plus un multiplu de $(p - 1)(q - 1)$. Ultima egalitate aminteste de mica teoremă a lui Fermat: $x^{(p-1)(q-1)} = 1$ mod $n$ (desigur, cu exceptia cazului în care $x$ este un multiplu de $p$ sau de $q$, o posibilitate foarte improbabilă); astfel, multiplul de $(p - 1)(q - 1)$ din exponent se poate ignora.

Astfel, Alice poate coda folosind cheia publică a lui Bob. Bob poate decoda utilizând cheia privată pe care numai el o stie. Dar ce se poate spune despre "curiosi"? Dacă Eve preia $e(x) = x^e$ mod $n$, ea poate face mai multe lucruri. Poate încerca toti $x$-ii posibili, să-i codeze cu cheia publică a lui Bob si să

găsească $x$-ul corect – dar asta ia mult prea mult timp. Sau, poate să calculeze $d = e^{-1} \bmod (p-1)(q-1)$ si din asta $e(x)^d \bmod n$ care este $x$. Dar pentru a face asta ea trebuie să stie $(p-1)(q-1)$; si dacă stie atât $p.q$ cât si $(p-1)(q-1)$ ea stie $p$ si $q$ (puteti spune de ce?). Cu alte cuvinte, ea poate factoriza pe $n$, un produs arbitrar de două numere prime mari – nimeni nu stie cum să facă asta rapid. Sau, în sfârsit, ea si-ar putea utiliza propria ei metodă ingenioasă pentru a decoda $e(x)$ fără factorizarea lui $n$ – este o convingere largă că nu există o asemenea metodă.
Astfel, după toate dovezile accesibile, RSA este sigur pentru un $n$ potrivit de mare.

**Semnături**

Criptografia cu cheie privată nu este numai o idee isteată si radical contraintuitivă, ea este o idee *fundamentală* de acest gen. Deoarece, odată în posesia cuiva, mai multe treburi care par imposibile pot fi efectuate. De exemplu, semnătura digitală.
Să presupunem că Alice vrea să-i trimită lui Bob un mesaj *semnat*. Adică vrea să expedieze un mesaj $m.s(m)$, cu $m$ un mesaj obisnuit si $s(m)$ o *semnătură*, o bucată de text care depinde de $m$ si care identifică în mod unic pe Alice (adică Bob este sigur că mesajul vine de la Alice si nu de la un impostor; de asemenea, Bob poate convinge o curte, un tribunal, că sigur Alice i-a trimis acel mesaj). Sună imposibil, nu-i asa?
Iată cum se face aceasta: $s(m) = d'(m)$ unde $d'(.)$ este functia de *decodare* a lui Alice. Adică Alice îsi imaginează pentru o secundă că mesajul $m$ pe care e pe cale să-l trimită lui Bob a fost primit de ea de la altcineva si i-a aplicat algoritmul ei de decodare (ridică la cheia ei secretă modulo cheia ei publică).
De îndată ce Bob primeste $m.s(m)$, el este plauzibil sigur că Alice a trimis mesajul deoarece numai Alice are cheia privată care poate transforma pe $m$ în $s(m)$.
Si există alte multe fapte contraintuitive care pot fi îndeplinite prin aplicarea criptografiei cu cheie publică: jocul de poker pe Net si demonstratia cuiva că o teoremă este adevărată fără a spune ceva despre adevăr (astfel de demonstratii sunt cunoscute ca *zero-knowledge proofs*).

**RSA si teorema restului chinezesc**

**Teorema restului chinezesc**

Să presupunem că avem un sistem de ecuatii simultane, poate ca acesta care urmează:

$$x \equiv 2 \pmod 5$$
$$x \equiv 5 \pmod 7$$

Ce putem spune despre $x$? Ei bine, putem spune că o solutie este $x = 12$; $x = 12$ satisface ambele ecuatii. Aceasta nu este singura solutie: de pildă $x = 12 + 35$ se

potriveste si ea ecuatiilor, dar si $x = 12 + 70$ si $x = 12 + 105$ s.a.m.d. Evident, adunând orice multiplu de 35 la o solutie se obtine o altă solutie validă, astfel că putem scrie rezumativ că $x \equiv 12 \pmod{35}$ este o solutie a sistemului de ecuatii dat.

Dar mai sunt si alte solutii? Pentru acest exemplu nu există o altă solutie; fiecare solutie este de forma $x \equiv 12 \pmod{35}$. Dar de ce nu? Dacă admitem existenta unei alte solutii $x'$, atunci din prima ecuatie se stie că $x \equiv 2 \pmod 5$ si $x' \equiv 2 \pmod 5$ si de aici $x \equiv x' \pmod 5$. Similar $x \equiv x' \pmod 7$. Dar prima congruentă spune că 5 este un divizor al diferentei $x - x'$, din a doua rezultă că si 7 este un divizor al diferentei $x - x'$, astfel că $x - x'$ trebuie să fie un multiplu de 35 (s-a utilizat aici faptul că $\gcd(5, 7) = 1$), care înseamnă $x \equiv x' \pmod{35}$. Cu alte cuvinte toate solutiile sunt aceleasi modulo 35: sau, echivalent, dacă tot ce ne preocupă este $x$ mod 35, solutia este unică.

Puteti verifica: aceleasi lucruri sunt adevărate dacă se înlocuiesc numerele 5, 7, 2, 5 cu oricare altele. Singurul lucru impus a fost ca $\gcd(5, 7) = 1$.

Iată generalizarea:

**Teorema 11.1**: (Teorema restului chinezesc) Fie $m$, $n$ relativ prime si fie $a$, $b$ arbitrare. Perechea de ecuatii $x \equiv a \pmod m$, $x \equiv b \pmod n$ are o solutie unică în $x$ mod $mn$.

Mai mult, solutia $x$ poate fi calculată eficient (ca exercitiu, cititorul poate verifica modul de a face calculul).

Teorema restului chinezesc este utilă adesea când se face aritmetică modulară cu un modul compus; dacă dorim să calculăm o valoare necunoscută modulo $mn$, un truc standard este a o calcula modulo $m$, de a o calcula modulo $n$ si apoi a deduce valoarea ei modulo $mn$ utilizând teorema restului chinezesc (CRT – *Chinese remainder theorem*).

**Teorema lui Euler**

În lectia anterioară am văzut mica teoremă a lui Fermat, care spune ceva despre ce se întâmplă cu exponentierea când modulul este prim. Putem generaliza cu usurintă la cazul când se lucrează cu modulo un produs de două numere prime.

**Teorema 11.2**: Fie $p$, $q$ două numere prime distincte. Fie $n = pq$. Atunci numărul $x^{(p-1)(q-1)} \equiv 1 \pmod n$ pentru orice $x$ care satisface conditia $\gcd(x, n) = 1$.

**Demonstratie**: Mai întâi să reducem modulo $p$ ambele părti ale congruentei din enunt (este permis deoarece $p$ divide pe $n$). Găsim că

$$x^{(p-1)(q-1)} \equiv (x^{(p-1)})^{(q-1)} \equiv (1)^{(q-1)} \equiv 1 \pmod p$$

unde s-a folosit mica teoremă a lui Fermat pentru a conchide că $x^{p-1} \equiv 1 \pmod p$. Similar avem $x^{(p-1)(q-1)} \equiv 1 \pmod q$. Se obtine un sistem de două ecuatii

$$x^{(p-1)(q-1)} \equiv 1 \pmod p$$
$$x^{(p-1)(q-1)} \equiv 1 \pmod q$$

De retinut că $\gcd(p, q) = 1$. Asadar, după teorema restului chinezesc, trebuie să existe o solutie unică pentru $x^{(p-1)(q-1)} \bmod pq$. Se poate vedea că $x^{(p-1)(q-1)} \equiv 1$

(mod $pq$) este una din solutiile posibile si în virtutea unicitătii aceasta trebuie să fie singura posibilitate. Adevărul teoremei decurge de aici.

□

Să recapitulăm. Mica teoremă a lui Fermat ne spune că $x^{p-1} \equiv 1 \pmod{p}$ si tocmai am văzut că $x^{(p-1)(q-1)} \equiv 1 \pmod{pq}$. Care este pattern-ul general aici? De unde vin acesti exponenti magici? Există vreo relatie generalizabilă între $p - 1$ si $p$ si $(p - 1)(q - 1)$ si $pq$? Da, există.

Conceptul de care avem nevoie este acela de *functia totient* a lui Euler, $\varphi(n)$. Numărul $\varphi(n)$ este definit a fi numărul de întregi pozitivi mai mici ca $n$ si mutual primi cu $n$. Se poate vedea că $\varphi(p) = p - 1$ când $p$ este prim, deoarece întregii 1, 2, …, $p - 1$ sunt toti mai mici decât $p$ si relativ primi cu $p$. Cu putin efort de numărare se poate determina si $\varphi(n)$ când $n = pq$, un produs de două numere prime. Se obtine rezultatul următor:

**Lema 11.1**: Fie $p$, $q$ două numere prime distincte si $n = pq$. Atunci $\varphi(n) = (p - 1)(q - 1)$.

**Demonstratie**: Câti întregi sunt mai mici decât $n$ si sunt relativ primi cu $n$? Ei bine, întregii $p$, $2p$, $3p$, …, $(q - 1)p$ nu se pun: ei au un factor comun cu $n$. Dintr-un motiv similar nu se numără nici $q$, $2q$, $3q$, …, $(p - 1)q$. Acestea sunt singurele exceptii si cele două liste de numere sunt disjuncte. Dacă din cele $n - 1$ numere mai mici ca $n$ eliminăm aceste exceptii se obtine cea ce urmărim. În final, prima listă contine $q - 1$ exceptii, a doua $p - 1$ exceptii. Asadar, sunt $n - 1 - (p - 1) - (q - 1) = pq - p - q + 1 = (p - 1)(q - 1)$ întregi pozitivi mai mici decât $n$ si relativ primi cu $n$.

□

La acest moment este natural a formula conjectura conform căreia functia totient a lui Euler croieste cărarea către o generalizare a micii teoreme a lui Fermat. Si, desigur, se poate demonstra următorul rezultat:

**Teorema 11.3**: (**Teorema lui Euler**) $x^{\varphi(n)} \equiv 1 \pmod{n}$ pentru orice $x$ care satisface conditia $\gcd(x, n) = 1$.

**Demonstratie**: Demonstratia va fi exact ca aceea pentru mica teoremă a lui Fermat. Se consideră o multime $\mathbf{\Phi}$ de întregi pozitivi mai mici decât $n$ si relativ primi cu $n$. Dacă alegem oricare element $x$ din $\mathbf{\Phi}$ obtinem o altă multime $\mathbf{\Phi}_x = \{ix \bmod n : i \in \mathbf{\Phi}\}$. De notat că toate elementele din $\mathbf{\Phi}_x$ sunt distincte (deoarece $x$ este inversabil) si relativ prime cu $n$, asadar $\mathbf{\Phi} = \mathbf{\Phi}_x$. În consecintă, produsele $\prod_{i \in \Phi} i$ si $\prod_{i \in \Phi_x} i$ sunt egale modulo $n$. Dar $\prod_{i \in \Phi} i \equiv \prod_{i \in \Phi_x} i \pmod{n}$, asadar $x^{|\mathbf{\Phi}|} \equiv 1 \pmod{n}$. în final, notând $|\mathbf{\Phi}| = \varphi(n)$, rezultă teorema.

□

Cititorul poate verifica mica teoremă a lui Fermat ca un caz special al teoremei lui Euler.

## RSA

În final, folosind teorema lui Euler, oferim o dovadă că RSA lucrează corect, adică decriptarea unui mesaj criptat RSA restituie mesajul initial.
Fie $n = pq$ produsul a două numere prime si fie $e$ un număr cu $\gcd(e, \varphi(n)) = 1$, astfel ca cheia publică RSA este dată de perechea $(n, e)$. Se reaminteste că o criptare a unui mesaj $m$ este definită ca

$$e(m) \equiv m^{\,e} \pmod{n}$$

Similar, decriptarea unui text cifrat $c$ este definită ca:

$$d(c) \equiv c^{\,d} \pmod{n}$$

cu $d \equiv e^{-1} \pmod{\varphi(n)}$. Vom dovedi că decriptarea este inversa criptării. Aceasta dă siguranta că receptorul poate recupera mesajul originar, criptat folosind cheia sa privată, cum trebuie să fie într-o schemă de criptare bună.
**Teorema 11.4**: $d(e(m)) \equiv m \pmod{n}$ ori de âte ori $\gcd(m, n) = 1$.
**Demonstratie**: Rememorăm că $e \equiv d^{-1} \pmod{\varphi(n)}$ astfel $ed \equiv 1 \pmod{\varphi(n)}$, sau cu alte cuvinte, $ed - 1$ este un multiplu de $\varphi(n)$, să spunem $k\varphi(n)$. Apoi calculăm

$$d(e(m)) \equiv (m^e)^d \equiv m^{ed} \equiv m^{1 + k\varphi(n)} \equiv m(m^{\varphi(n)})^k \equiv m.1^k \equiv m \pmod{n}$$

În calcul s-a utilizat teorema lui Euler pentru a conchide că $m^{\varphi(n)} \equiv 1 \pmod{n}$.
□

## Revedere a RSA

Observatia 1: Sunt suficiente numere prime pentru a facilita găsirea a două din ele; există aproximativ $x/\ln x$ numere prime între 1 si $x$.
Observatia 2: Functia de criptare si functia de decriptare pot fi aplicate în orice ordine.

## Autentificarea

O situatie de discutat: mesajul nu este necesarmente secret dar este de dorit o asigurare asupra autorului lui. Alice crează o semnătură digitală $S$ pentru mesajul ei $M$: $S = M^d \bmod n$, unde $(d, n)$ este cheia privată a Alicei; ea poate atunci să cripteze semnătura utilizând cheia publică a lui Bob. La primirea mesajului, Bob o va decripta uzând de cheia lui privată, apoi va aplica cheia publică a Alicei pentru a se asigura că aceasta vine de la ea si nu s-a produs vreo interferentă nedorită.
Ceea ce se întâmplă în realitate este că Alice semnează un *digest* (un rezumat) al mesajului $M$ format prin aplicarea unei functii *hash*[5] mesajului $M$, apoi atasează versiunea mai scurtă semnată a mesajului, la mesajul $M$ real; Bob aplică aceeasi functie *hash* lui $M$ pentru a obtine *digest*-ul, apoi aplică semnăturii cheia publică a lui Alice si compară rezultatul cu $h(M)$. De notat că

[5] O functie *hash* este o amprentă numerică redusă produsă/rezultată din orice gen de date. Functiile hash sunt utilizate si în criptografie.

Alice semnează în esentă orice mesaj care se rezumă prin *hash* la acelasi digest; acest lucru reprezintă un risc de securitate.
Recent, SHA-1 (**S**ecure **H**ash **A**lgorithm – 1, unul dintr-o serie de algoritmi siguri de obtinere de versiuni *hash*, SHA-1, SHA-224, SHA-256, SHA-384 si SHA-512), o functie hash pentru acest scop, a fost compromisă; s-a găsit o functie mai simplă care produce aceleasi valori ca si SHA-1.
Un ceritificat dă asigurarea că cheia publică a lui Alice este corectă; el adaugă o semnătură (de genul Verisign) la cheia publică care verifică autenticitatea ei. Există de asemenea autorităti *time-stamp* pentru verificarea timpului la care mesajul a fost compus/trimis.

**Utilizarea practică a RSA**

Algoritmii cu cheie privată sunt mult mai rapizi (de cca. 1000 de ori). În general, exponentul public este de obicei mult mai mic decât exponentul privat, ceea ce înseamnă că decriptarea unui mesaj este mai rapidă decât criptarea si verificarea unei semnături este mai rapidă decât semnarea; aceasta-i foarte bine deoarece semnarea si criptarea sunt făcute numai o dată, în timp ce verificarea unei semnături date se poate repeta de mai multe ori. RSA este utilizată deobicei pentru a coda o cheie care poate fi apoi utilizată cu un sistem de cheie privată.

**SSH (*S*ecure *Sh*ell)**

Programul ssh-keygen generează o pereche de chei publică/privată prin alegerea aleatoare de numere prime mari; această operatie consumă câteva minute pe un minicomputer nova.
În particular, cheia privată este depusă în .ssh2/id_dsa_2048_a; cheia publică este depusă în .ssh2/id_dsa_2048_a.pub. Procedura de logare din contul A la contul B: SSH de pe A trimite un identificator de sesiune semnat (cunoscut numai clientului si serverului, criptat cu cheia privată a lui A) serverului SSH al lui B. B se asigură că cheia publică a lui A se află într-un fisier denumit authorized_keys si este corectă. Dacă asa este, serverul SSH al lui B permite logarea la B.
SSH mentine si verifică totodată o bază de date care contine informatii de identificare pentru fiecare *host* cu care a lucrat vreodată; dacă un host se schimbă, utilizatorul este avertizat. Toate comunicatiile în orice directie sunt criptate cu o metodă de cheie privată.

**Alte aplicatii înrudite**

*Jocul de poker pe Internet* ("Mental Poker", by SRA, în The Mathematical Gardner, editat de David Klarner, publicat de Wadsworth în 1981).
Alice si Bob convin asupra unor functii de criptare/decriptare $E$ si $D$ pentru care:

$E_k(X)$ este versiunea criptată a mesajului $X$ cu cheia $K$
$D_k(E_k(X)) = X$ pentru orice mesaj $X$ si cheia $K$
$E_k(E_j(X)) = E_j(E_k(X)$ pentru orice mesaj $X$ si cheile $J$ si $K$
Fiind dat $X$ si $E_k(X)$, nu se poate deriva $K$, oricare ar fi $X$ si $K$
Fiind date mesajele $X$ si $Y$, nu se pot găsi cheile $J$ si $K$ astfel încât $E_j(X) = E_k(Y)$

Alice si Bob aleg cheile secrete $A$ si $B$, respectiv.
Bob criptează cele 52 de mesaje, "2 de treflă", "3 de treflă", …, "As de pică" cu cheia lui secretă $B$. Apoi rearanjează aleator pachetul criptat si-l trimite integral lui Alice.
Alice selectează cinci cărti (mesaje) la întâmplare si le trimite înapoi lui Bob, necriptate (ele au fost criptate deja de Bob). Bob decriptează aceste mesaje pentru a afla care este mâna/formatia sa.
Acum Alice selectează alte cinci mesaje, le criptează cu cheia ei secretă $A$ si le trimite lui Bob. Nici unul din aceste mesaje nu este criptat dublu, o dată de Bob si încă o dată de Alice. Bob le decriptează (utilizând comutativitatea functiei de criptare/decriptare), apoi le trimite retur la Alice. După decriptarea acelor mesaje, Alice stie care este cartea ei.
La finalul jocului, ambii jucători devoalează cheile lor secrete. Acum fiecare jucător poate verifica faptul că celălalt a "distribuit realmente" cărtile pe care ea sau el pretinde că le au pe durata jocului. Din punct de vedere al calculului este dificil a devoala cheia falsă.
*Amprentarea*. Se presupune că este de trimis un fisier spre lună. Apar când si când biti eronati. Ne-ar plăcea să verificăm corectitudinea fisierului ajuns acolo. Am putea retrimite fisierul a doua oară dar lărgimea de bandă către lună este foarte costisitoare, astfel încât ar fi de dorit o solutie mai bună.
Solutia este amprentarea. Se presupune că stim că mesajul $x$ de expediat este lung de $n$ biti. Alegem în avans un număr prim $p$, la întâmplare dintre numerele prime mai mici decât $n^3$. Îl scriem pe $x$ ca un număr $0 \le x < 2^n$ în modul cunoscut. Apoi se evaluează amprenta care este $F_p(x) = x \bmod p$. Trimitem $F_p(x)$ odată cu $x$. De observat că lungimea lui $F_p(x)$ este de $3\log n$ biti, si este mult mai scurtă decât $x$.
Care este probabilitatea ca o eroare aleatoare de transmitere să rămână nedetectată? Ei bine, să presupunem că receptorul primeste $(y, c)$ în loc de $(x, F_p(x))$ si admitem că $y$ este diferit de $x$. Atunci, probabilitatea ca un bit eronat să treacă neobservat/nedetectat este probabilitatea ca amprenta să se prezinte ca validă:

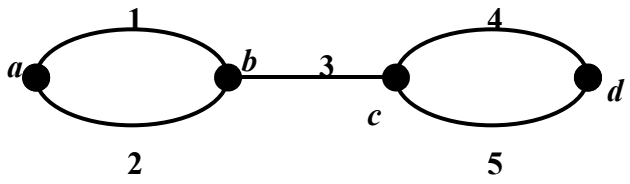
$$\Pr[F_p(y) = c] = \Pr[p|(y-c)] \le$$
$$\le (\text{numărul de divizori primi al lui } (y-c))/(\text{numărul de prime inferioare lui } n^3) \le$$
$$\le n/(\text{numărul de prime inferioare lui } n^3) \approx n/(n^3/(\ln n^3)) \le 1/n$$

Asadar, probabilitatea aceasta este foarte scăzută, adevăr care se mentine indiferent de numărul de biti eronati care ar putea apărea în transmisie. De observat că si rapiditatea calculului amprentei este mare, $O(n\log n)$, si numărul de biti suplimentari transmisi o dată cu mesajul de lungime $n$ este redus, $O(\log n)$.

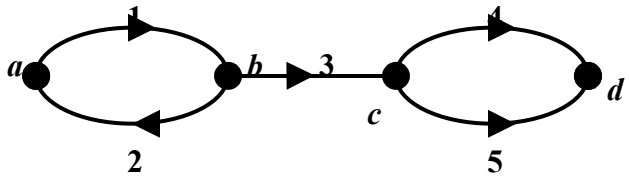
# Lecția 12

## Grafuri – introducere

**Definitia 12.1**: Un *graf simplu* **G** = (**V**, **E**) constă într-o *multime nevidă* **V** de noduri si o multime **E** de *perechi neordonate* de elemente distincte din **V** numite arce.
Un graf simplu este asemănător unui graf orientat cu deosebirea că arcele nu au specificat un sens, de la un nod la un altul.
Uneori sunt de modelat conexiuni multiple între noduri, ceea ce este imposibil cu grafuri simple. În aceste cazuri trebuie utilizate *multigrafuri*.
**Definitia 12.2**: Un *multigraf* **G** = (**V**, **E**) constă într-o multime **V** de noduri, o multime **E** de arce si o functie $f$: $\mathbf{E} \rightarrow \{(u, v) | u, v \in \mathbf{V}, u \neq v\}$.
Arcele $e_1$ si $e_2$ sunt denumite arce multiple sau arce paralele dacă $f(e_1) = f(e_2)$.
De notat: Buclele nu sunt admise în multigrafuri ($u \neq v$).
Exemplu: Un multigraf **G** cu nodurile **V** = {$a$, $b$, $c$, $d$}, arcele {1, 2, 3, 4, 5} si functia $f$ cu $f(1) = (a, b)$, $f(2) = (a, b)$, $f(3) = (b, c)$, $f(4) = (c, d)$ si $f(5) = (c, d)$:




Dacă sunt de definit (si) bucle, este necesar un alt fel de graf:
**Definitia 12.3**: Un *pseudograf* **G** = (**V**, **E**) este o multime de noduri **V**, o multime de arce **E** si o functie $f$ de la **E** la $\{(u, v) | u, v \in \mathbf{V}\}$.
Un arc este buclă dacă $f(e) = (u, u)$ pentru un $u$ din **V**.
Iată acum un graf de un tip deja cunoscut.
**Definitia 12.4**: Un *graf orientat* **G** = (**V**, **E**) este o multime **V** de noduri si o multime **E** de arce care sunt perechi *ordonate* de elemente din **V**.
Acest tip de graf duce la alt tip de graf:
**Definitia 12.5**: Un *multigraf orientat* **G** = (**V**, **E**) este o multime **V** de noduri, o multime **E** de arce si o functie $f$ de la **E** la $\{(u, v) | u, v \in \mathbf{V}, u \neq v\}$.
Arcele $e_1$ si $e_2$ se numesc arce multiple dacă $f(e_1) = f(e_2)$.

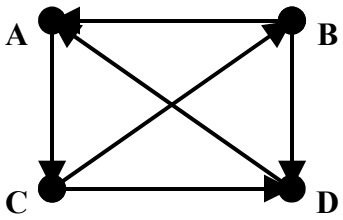



Un exemplu: Un multigraf **G** cu nodurile **V** = {*a*, *b*, *c*, *d*}, arcele {1, 2, 3, 4, 5} si functia *f* cu *f*(1) = (*a*, *b*), *f*(2) = (*b*, *a*), *f*(3) = (*b*, *c*), *f*(4) = (*c*, *d*) si *f*(5) = (*c*, *d*): Se poate acum sintetiza într-un tabel tipurile de grafuri si proprietătile lor:

| Tipul | Arce | Arce multiple | Bucle |
|---|---|---|---|
| Graf simplu | Fără orientare | Nu | Nu |
| Multigraf | Fără orientare | Da | Nu |
| Pseudograf | Fără orientare | Da | Da |
| Graf orientat | Orientate | Nu | Da |
| Multigraf orientat | Orientate | Da | Da |

**Modele de tip graf**

Exemplul 1: Reprezentarea unei retele de căi ferate. Statiile sunt noduri ale grafului, legătura directă între două statii este un arc al grafului. Graful nu este orientat, circulatia trenurilor se face în ambele sensuri pe fiecare tronson de cale ferată.
Exemplul 2: Rezultatele unui turneu sportiv în care competitorii (echipele) se întâlnesc fiecare-cu-fiecare si rezultatul egal este exclus se pot reprezenta într-un graf orientat ca acela de mai jos:

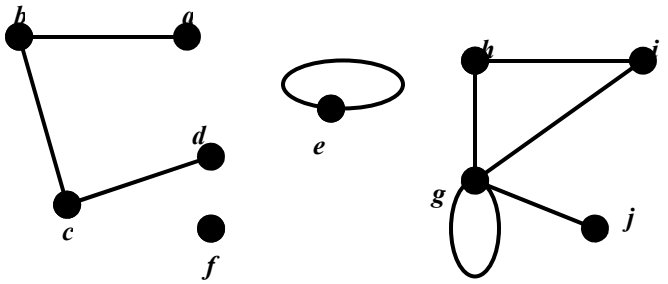



Arcul (B, A) indică faptul că B l-a învins pe A.

**Terminologie**

**Definitia 12.6**: Două noduri *u* si *v* dintr-un graf neorientat **G** sunt calificate ca *adiacente* (sau *vecine*) în **G** dacă (*u*, *v*) este arc în **G**.

Dacă $e = (u,v)$, arcul $e$ se numeste *incident* cu nodurile $u$ si $v$. Se mai spune că arcul $e$ *conectează* nodurile $u$ si $v$.
Nodurile $u$ si $v$ se numesc *capetele* arcului $(u, v)$.
**Definitia 12.7**: *Gradul* unui nod al unui graf fără orientare este numărul de arce incidente cu acel nod. O buclă contribuie de două ori la gradul nodului în jurul căruia este construită. Cu alte cuvinte, gradul unui nod poate fi stabilit în cazul unui graf desenat prin *numărarea liniilor* care îl vizitează.
Gradul unui nod $v$ se notează cu $\deg(v)$.
Un nod de gradul zero este un *nod izolat* deoarece nu este adiacent nici unui alt nod.
Notă: Un nod cu o buclă în el are cel putin gradul 2 si, prin definitie, nu este izolat chiar dacă nu este adiacent vreunui alt nod.
Un nod cu gradul 1 este un nod *pandant*. El este adiacent exact unui nod.
Exemplu: Care din nodurile grafului următor sunt izolate, care sunt pandante si care este gradul maxim? Cu ce tip de graf avem de-a face?




Solutia: Nodul $f$ este izolat, nodurile $a$, $d$, $j$ sunt pandante. Gradul maxim este $\deg(g) = 5$. Acest graf este un pseudograf (este neorientat, are bucle).
**Teorema 12.1** (**a strângerii de mână**): Fie **G** = (**V**, **E**) un graf neorientat cu $e$ arce. Are loc relatia:

$$2e = \Sigma_{v \in V} \deg(v)$$

Această teoremă este adevărată chiar dacă sunt prezente arce multiple si/sau bucle.
Exemplu: Câte arce sunt într-un graf cu 10 noduri, fiecare de gradul 6?
Solutie: Suma gradelor tuturor nodurilor este 6.10 = 60. Conform teoremei $2e = 60$ de unde $e = 30$ de arce.
**Teorema 12.2**: Un graf neorientat are un număr par de noduri de grad impar.
Idei pentru **demonstratie**: Sunt trei posibilităti de a adăuga un arc pentru a lega două noduri într-un graf:

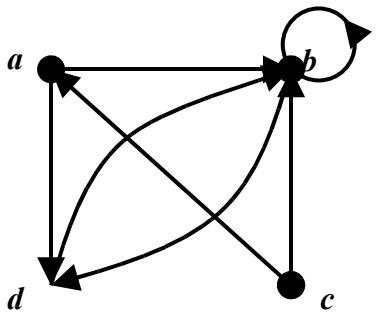
| Înainte de adăugare | | După adăugare |
|---|---|---|
| Ambele noduri au gradul par | → | Ambele noduri au gradul impar |
| Ambele noduri au gradul impar | → | Ambele noduri au gradul par |

| | | |
|---|---|---|
| Cele două noduri au grade de parități diferite | → | Cele două noduri au grade de parități diferite |

Sunt două posibilități de a adăuga o buclă pe un nod din graf:

| Înainte de adăugare | | După adăugare |
|---|---|---|
| Nodul are gradul par | → | Nodul are gradul par |
| Nodul are gradul impar | → | Nodul are gradul impar |

Asadar, dacă există un număr par de noduri de grad impar, după adăugarea unui arc numărul va fi tot par.
Dar un graf neorientat cu zero arce are un număr par de noduri de grad impar (zero), acelasi fapt trebuie să fie adevărat si pentru orice graf neorientat.
**Definitia 12.8**: Dacă ($u$, $v$) este un arc într-un graf **G** cu arcele orientate, se spune că $u$ este *adiacent la* $v$ si $v$ este *adiacent din* $u$.
Nodul $u$ este *nodul initial* al arcului ($u$, $v$) si $v$ este *nodul terminal* al arcului. Pentru o buclă, nodul initial si cel terminal coincid.
**Definitia 12.9**: Într-un graf cu arcele orientate, gradul imergent al unui nod $v$, notat $\deg^-(v)$ este numărul de arce care au pe $v$ ca nod terminal. Gradul emergent al nodului $v$, notat $\deg^+(v)$ este numărul de arce care au pe $v$ ca nod initial.
La întrebarea "cum se modifică gradele imergent si emergent al unui nod prin adăugarea în acel nod a unei bucle?" răspunsul este "gradul imergent si gradul emergent cresc cu câte o unitate".
Exemplu: Care sunt gradele imergente si emergente ale nodurilor $a$, $b$, $c$, $d$ în graful alăturat?



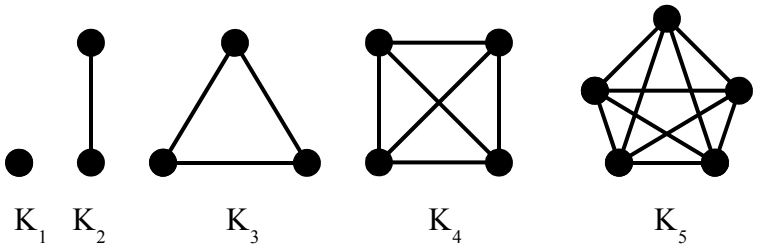

Răspunsul: $\deg^-(a) = 1$, $\deg^-(b) = 4$, $\deg^-(c) = 0$, $\deg^-(d) = 2$ si $\deg^+(a) = 2$, $\deg^+(b) = 2$, $\deg^+(c) = 2$, $\deg^+(d) = 1$.
**Teorema 12.3**: Fie **G** = (**V**, **E**) un graf cu arcele orientate. Atunci:

$$\Sigma_{v \in V} \deg^-(v) = \Sigma_{v \in V} \deg^+(v) = |\mathbf{E}|$$

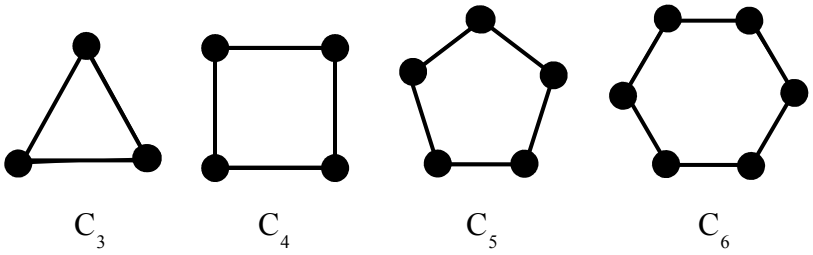
Acest fapt este usor de înteles deoarece fiecare nou arc face să crească cu o unitate atât suma pe gradele imergente cât si suma pe gradele emergente.

**Grafuri speciale**

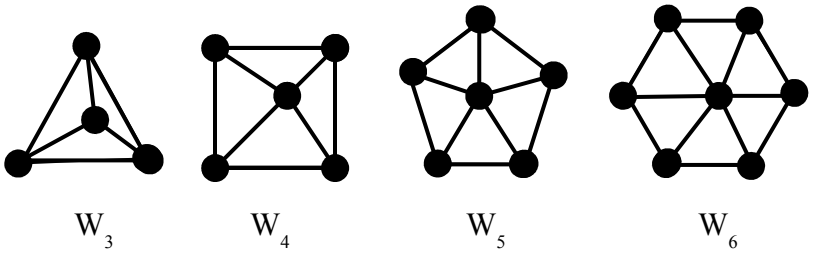
**Definitia 12.10**: Un graf complet pe $n$ noduri, notat cu $\mathbf{K}_n$, este un graf simplu care contine exact un arc între fiecare pereche de noduri distincte.



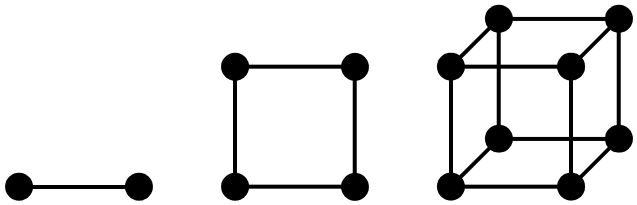

**Definitia 12.11**: Ciclul $\mathbf{C}_n$, $n \geq 3$, constă în $n$ noduri $v_1$, $v_2$, …, $v_n$ si arcele $(v_1, v_2)$, $(v_2, v_3)$, …, $(v_{n-1}, v_n)$, $(v_n, v_1)$,.



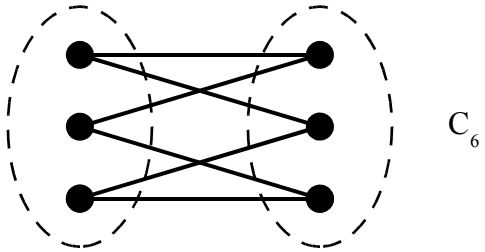

**Definitia 12.12**: Volantul $\mathbf{W}_n$ constă în ciclul $\mathbf{C}_n$ ($n \geq 3$) la care se adaugă un nod suplimetar care se conectează cu fiecare din cele $n$ noduri ale ciclului, prin adăugare de arce.



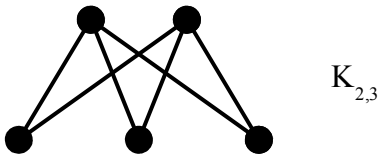

**Definitia 12.13**: Cubul $n$-dimensional, notat $\mathbf{Q}_n$ este graful care are ca noduri cele $2^n$ secvente distincte care se pot alcătui cu $n$ biti. Două noduri sunt adiacente dacă si numai dacă sirurile de biti pe care le reprezintă diferă prin exact un bit.



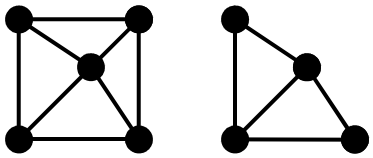

**Definitia 12.14**: Un graf simplu este *bipartit* dacă multimea nodurilor sale **V** poate fi partitionată în două multimi disjuncte nevide, $\mathbf{V}_1$ si $\mathbf{V}_2$, astfel încât orice arc din graf leagă un nod din $\mathbf{V}_1$ cu un nod din $\mathbf{V}_2$ (nici un arc nu leagă două noduri din $\mathbf{V}_1$ sau două noduri din $\mathbf{V}_2$).
De exemplu, graful care are ca noduri persoanele (căsătorite) dintr-o localitate si ca arce relatia de căsătorie este un graf bipartit. Este bipartit pentru că arcele leagă un nod din submultimea bărbatilor cu un nod din submultimea (disjunctă) a femeilor (se consideră aici căsătoria traditională).
Exemplul 1. Este $\mathbf{C}_3$ bipartit? Nu, deoarece nu există vreun mod de a partitiona multimea nodurilor în două submultimi fără arce care să lege noduri din aceeasi submultime.
Exemplul 2. Este $\mathbf{C}_6$ bipartit? Da, deoarece se poate rearanja ca în figura alăturată.




**Definitia 12.15**: Un graf bipartit complet, $\mathbf{K}_{m,n}$ este un graf care are multimea nodurilor partitionată în două submultimi de $m$ si $n$ noduri. Două noduri sunt conectate dacă si umai dacă ele sunt în submultimi diferite.

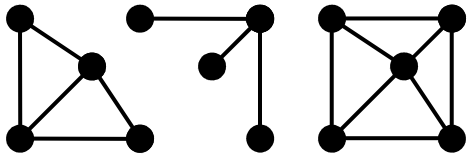


**Operatii cu grafuri**



$W_4$ subgraf al lui $W_4$

**Definitia 12.16**: Un subgraf al unui graf **G** = (**V**, **E**) este un graf **H** = (**W**, **F**) în care **W** ⊆ **V** si **F** ⊆ **E**. Desigur, **H** este un graf valid astfel că nu se pot elimina capetele unor arce încă prezente. Un exemplu, în figura de mai sus.
**Definitia 12.17**: Reuniunea a două grafuri simple $\mathbf{G}_1 = (\mathbf{V}_1, \mathbf{E}_1)$ si $\mathbf{G}_2 = (\mathbf{V}_2, \mathbf{E}_2)$ este graful simplu cu multimea de noduri $\mathbf{V}_1 \cup \mathbf{V}_2$ si cu multimea de arce $\mathbf{E}_1 \cup \mathbf{E}_2$. Reuniunea celor două grafuri se notează $\mathbf{G}_1 \cup \mathbf{G}_2$.
Exemplu:



$G_1$ $G_2$ $W_4 = G_1 G_2$

**Reprezentarea grafurilor**

Reprezentarea tabelară a unui graf fără orientare specifică adiacenta fiecărui nod. De pildă, tabelul următor

| Noduri | Noduri adiacente |
|---|---|
| a | b, c, d |
| b | a, d |
| c | a, d |
| d | a, b, c |

specifică multimea de noduri si conexiunile fiecărui nod cu alte noduri (ca exercitiu, versiunea grafică a acestui graf).
Reprezentarea tabelară a unui graf orientat specifică multimea arcelor prin nodurile de plecare si de sosire ale fiecăruia. De pildă, tabelul următor

| Noduri initiale | Noduri terminale |
|---|---|
| a | c |

| | |
|---|---|
| b | a |
| c | |
| d | a, b, c |

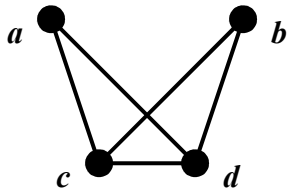
specifică multimea de arce, în număr de 5, care pleacă din fiecare nod cu destinatie alt nod, alte noduri. Nodul **c** nu este originea nici unui arc. Se propune ca exercitiu, reprezentarea geometrică a acestui graf.

**Definitia 12.18**: Fie **G** = (**V**, **E**) un graf simplu cu |**V**| = $n$. Se presupune că nodurile grafului sunt listate într-o ordine arbitrară $v_1$, $v_2$, …, $v_n$. Matricea de adiacentă $A$ (sau $A_{\mathbf{G}}$) a grafului în raport cu această listă este o matrice $n \times n$ cu valori 1 si 0, cu 1 în pozitia ($i$, $j$) ori de câte ori $v_i$ si $v_j$ sunt adiacente, cu 0 în pozitia ($i$, $j$) în celealte cazuri. Cu alte cuvinte, matricea de adiacentă $A$ are elementele

$a_{ij}$ = 1 dacă ($v_i$, $v_j$) este arc în G

$a_{ij}$ = 0 altminteri

Exemplu: Matricea de adiacentă a grafului din figură



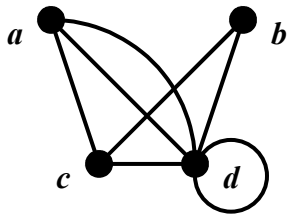

este

$$A_G = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

cu liniile si coloanele ordonate $a$, $b$, $c$, $d$.

De observat că matricea de adiacentă a unui graf fără orientare este totdeauna simetrică.

Pentru reprezentarea grafurilor cu arce multiple, în loc de valorile 0 si 1 se folosesc numere naturale: în pozitia ($i$, $j$) se pune numărul de arce asociate cu perechea de noduri ($v_i$, $v_j$). De exemplu, pentru graful din figură



matricea de adiacentă este

$$A_G = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 \end{bmatrix}$$

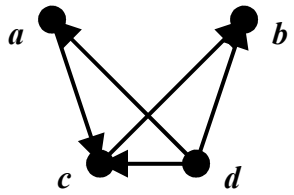
Simetria matricei se păstrează deoarece graful este (tot) fără orientare.

**Definitia 12.19**: Fie **G** = (**V**, **E**) un graf orientat cu |**V**| = $n$. Se presupune că nodurile grafului sunt listate într-o ordine arbitrară $v_1$, $v_2$, …, $v_n$. Matricea de adiacentă $A$ (sau $A_{\mathbf{G}}$) a grafului în raport cu această listă este o matrice $n \times n$ cu valori 1 si 0, cu 1 în pozitia $(i, j)$ ori de câte ori există un arc de la $v_i$ la $v_j$ si cu 0 în pozitia $(i, j)$ în celelalte cazuri. Cu alte cuvinte, matricea de adiacentă $A$ are elementele

$a_{ij}$ = 1 dacă ($v_i$, $v_j$) este un arc în G

$a_{ij}$ = 0 altminteri

Exemplu: Pentru ordinea alfabetică, normală, $a$, $b$, $c$, $d$, matricea de adiacentă a grafului din figură




este

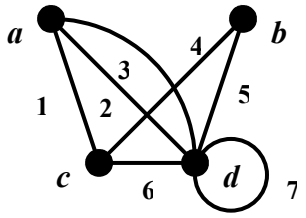
$$A_G = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

**Definitia 12.20**: Fie **G** = (**V**, **E**) un graf neorientat cu |**V**| = $n$ si |**E**| = $m$. Se admite o ordine a nodurilor, $v_1$, $v_2$, …, $v_n$ si, respectiv, o ordine a arcelor $e_1$, $e_2$, …, $e_m$. Matricea de incidentă a grafului G în raport cu aceste liste ordonate de noduri si arce este o matrice $n \times m$ cu elementele zerouri si unităti, cu 1 pe pozitia $(i, j)$ dacă arcul $e_j$ este incident nodului $v_i$ si cu 0 în acea pozitie, în alte situatii. Cu alte cuvinte, pentru matricea de incidentă $M_{\mathbf{G}}$

$m_{ij}$ = 1 dacă arcul $e_j$ este incident nodului $v_i$

$m_{ij}$ = 0 altminteri.

Exemplu: Pentru graful următor



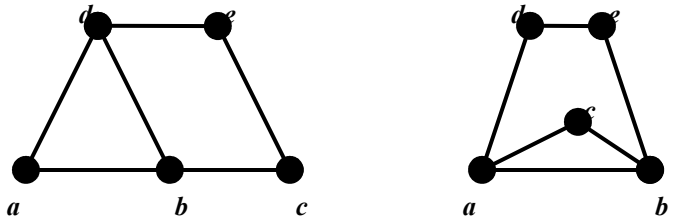

matricea de incidentă noduri-arce este

$$M_G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

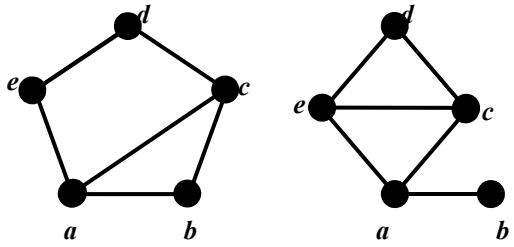
Se observă că matricea de incidentă contine două unităti pe coloanele arcelor care unesc două noduri si un singur de 1 pentru bucle.

**Izomorfisme de grafuri**

**Definitia 12.21**: Grafurile simple $\mathbf{G}_1 = (\mathbf{V}_1, \mathbf{E}_1)$ si $\mathbf{G}_2 = (\mathbf{V}_2, \mathbf{E}_2)$ sunt izomorfe dacă există o bijectie *f* de la $\mathbf{V}_1$ la $\mathbf{V}_2$ cu proprietatea că nodurile *a* si *b* sunt adiacente în $\mathbf{G}_1$ dacă si numai dacă *f*(*a*) si *f*(b) sunt adiacente în $\mathbf{G}_2$ pentru orice *a* si *b* din $\mathbf{V}_1$. O asemenea functie *f* se numeste izomorfism.
În alti termeni, grafurile $\mathbf{G}_1$ si $\mathbf{G}_2$ sunt izomorfe dacă nodurile lor pot fi ordonate astfel încât matricile lor de adiacentă $M_{G1}$ si $M_{G2}$ să coincidă.
Din punct de vedere vizual, grafurile $\mathbf{G}_1$ si $\mathbf{G}_2$ sunt izomorfe dacă pot fi aranjate astfel încât înfătisarea lor să fie identică (desigur, fără a schimba adiacenta).
Din păcate, pentru două grafuri simple, fiecare cu câte *n* arce, sunt *n*! functii posibile care ar trebui verificate dacă există un izomorfism.
Este mult mai usor de arătat că două grafuri nu sunt izomorfe atunci când ele nu sunt astfel.
Pentru aceasta se pot verifica anumiti invarianti, proprietăti pe care cele două grafuri dacă sunt izomorfe trebuie să le aibă. De exemplu, numărul de noduri, numărul de arce, si gradele nodurilor care trebuie să fie aceleasi dacă grafurile sunt izomorfe. Orice două grafuri care diferă fie si prin unul din acesti invarianti nu pot fi izomorfe. Dar două grafuri care se potrivesc la toti invariantii enumerati nu sunt în mod necesar izomorfe.
Un exemplu: Două grafuri izomorfe.




Ele pot fi aranjate să arate identic. Functia $f$, izomorfismul este $f(a) = c, f(b) = b, f(c) = e, f(d) = a, f(e) = d$.

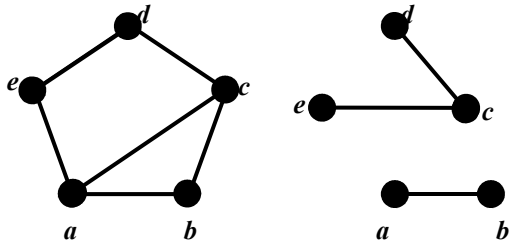



Un alt exemplu: Două grafuri care nu sunt izomorfe si nu pot fi izomorfe deoarece diferă în gradele nodurilor. Nodul $b$ al grafului din dreapta are gradul 1 si nu există un asemenea nod în graful din stânga.

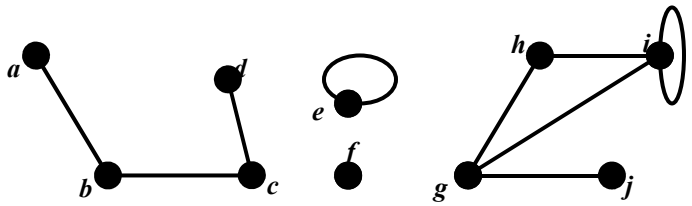
**Conectivitate**

**Definitia 12.22**: Într-un graf neorientat, un *drum* de lungime $n$ de la $u$ la $v$, cu $n$ un întreg pozitiv, este o secventă de arce ale grafului, $e_1, e_2, \ldots, e_n$ astfel încât $f(e_1) = (x_0, x_1), f(e_2) = (x_1, x_2), \ldots, f(e_n) = (x_{n-1}, x_n)$ cu $x_0 = u$ si $x_n = v$.
Când graful este simplu, drumul poate fi notat prin secventa lui de noduri $x_0, x_1, \ldots, x_n$ deoarece aceasta defineste în mod unic drumul.
Un drum este un *circuit* dacă începe si se sfârseste în acelasi nod; în definitia de mai sus $u = v$.
Drumul sau circuitul *trece* prin nodurile $x_1, x_2, \ldots, x_{n-1}$. Un drum sau un circuit este simplu dacă nu contine vreun acelasi arc de mai multe ori.
**Definitia 12.23**: Într-un graf orientat, un *drum* de lungime $n$ de la $u$ la $v$, cu $n$ un întreg pozitiv, este o secventă de arce ale grafului, $e_1, e_2, \ldots, e_n$ astfel încât $f(e_1) = (x_0, x_1), f(e_2) = (x_1, x_2), \ldots, f(e_n) = (x_{n-1}, x_n)$ cu $x_0 = u$ si $x_n = v$.
Dacă pe drum nu există arce multiple, drumul poate fi definit unic prin secventa de noduri care-i apartin, $x_0, x_1, \ldots, x_n$.
Drumul este un circuit dacă nodul de început si nodul de final coincid, $u = v$. Si în cazul grafurilor orientate drumul este simplu dacă nici un arc nu este parcurs de două sau de mai multe ori.
Alte fapte relativ la grafuri:

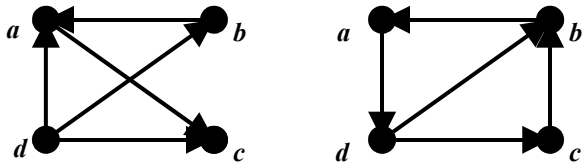
**Definitia 12.24**: Un graf fără orientare este *conex* dacă între oricare două noduri distincte ale grafului există un drum.
Într-o retea de calculatoare, modelabilă cu un graf, oricare două calculatoare pot comunica între ele dacă există un drum între ele.
Prin definitie, un graf cu un singur nod este conex deoarece nu există nici o pereche de noduri.
Exemple, în figurile alăturate: primul graf este conex, al doilea graf nu este conex.



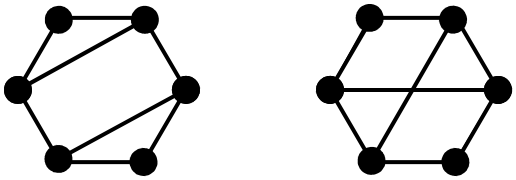

**Teorema 12.4**: Într-un graf conex neorientat există un drum simplu între oricare două noduri distincte ale grafului.
**Demonstratie**: Existenta unui drum este o certitudine. Dacă drumul acesta nu are bucle teorema este dovedită. Dacă există bucle, prin eliminarea lor pe secventa de noduri continută, într-un număr finit de pasi se ajunge la un drum simplu.
**Definitia 12.25**: Un graf neconex este reuniunea mai multor subgrafuri conexe; subgrafurile în perechi nu au noduri comune. Aceste subgrafuri conexe disjuncte sunt numite *componentele conexe* ale grafului.
Exemplu: Componentele conexe ale grafului următor




sunt $\{a, b, c, d\}$, $\{e\}$, $\{f\}$, $\{g, h, i, j\}$.
**Definitia 12.26**: Un graf orientat este *tare* conex dacă există un drum de la $a$ la $b$ si un drum de la $b$ la $a$ oricare ar fi nodurile $a$, $b$ ale grafului.
**Definitia 12.27**: Un graf orientat este *slab* conex dacă există un drum între oricare două noduri ale grafului.
Exemple: Dintre grafurile de mai jos unul este slab conex, primul, unul este tare conex, cel de al doilea (de ce?).




Numărul si dimensiunile componentelor conexe si circuitele sunt alti invarianti, argumente legate de (non)existenta izomorfismului grafurilor simple. De exemplu, grafurile următoare



nu sunt izomorfe deoarece graful din stânga are circuite de lungime 3 pe când graful din dreapta nu are asemenea circuite.

**Problema celui mai scurt drum**

Arcelor unui graf li se pot atasa ponderi. Ponderile pot fi, de pildă, distante între orase, pe sosea sau pe calea ferată, în cazul când una sau ambele retele apartinând de infrastructura unei regiuni sunt modelate prin grafuri (noduri – localitătile, arce – tronsoane de sosele sau de cale ferată).
Tot prin grafuri ponderate se pot modela si retelele de calculatoare, cu ponderi timpul de răspuns sau costul legăturii între noduri.
Si într-un caz si în altul, si în multe altele, apare problema drumului celui mai scurt între două noduri, drumul cu suma ponderilor arcelor minimă. În exemplele date asta ar însemna ruta cea mai scurtă între două orase sau cea mai rapidă conexiune între două noduri ale unei retele de calculatoare.

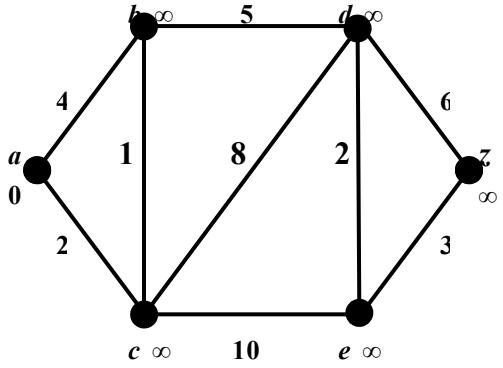
**Algoritmul lui Dijkstra**

Algoritmul lui Dijkstra este o procedură iterativă care găseste cel mai scurt drum între două noduri *a* si *z* dintr-un graf ponderat. Porneste prin a găsi cel mai scurt drum de la nodul *a* la noduri succesive prin adăugarea acestora la o multime specială de noduri **S**. Algoritmul se încheie atunci când nodul *z* este atins.

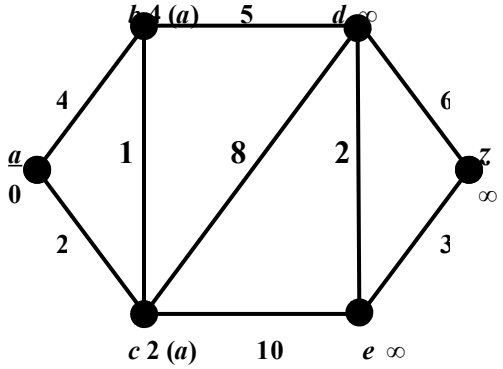
***Procedure*** Dijkstra (**G**: graf simplu conex ponderat cu vârfurile $a = v_0, v_1, \ldots, v_n = z$ si cu ponderile pozitive $w(v_i, v_j)$, cu $w(v_i, v_j) = \infty$ dacă $(v_i, v_j)$ nu este arc în **G**)

***for*** $i$ : = 1 ***to*** $n$
$L(v_i)$ : = ∞
$L(a)$ : = 0
**S** : = Ø
{sunt initializate etichetele nodurilor, $a$ cu eticheta 0, celelalte cu eticheta ∞; multimea **S** a nodurilor speciale este deocamdată vidă}
***while*** $z \notin$ **S**
***begin***
$u$ : = nodul care nu este în **S** cu $L(u)$ minim
**S** : = **S** ∪ {u}
***for*** toate nodurile $v$ care nu sunt în **S**
***if*** $L(u) + w(u, v) < L(v)$ ***then*** $L(v) : = L(u) + w(u, v)$
{se adaugă un nod la **S**, cel cu etichetă minimă si se actualizează etichetele nodurilor care nu sunt în **S**}
***end*** {$L(z)$ = lungimea celui mai scurt drum de la $a$ la $z$}

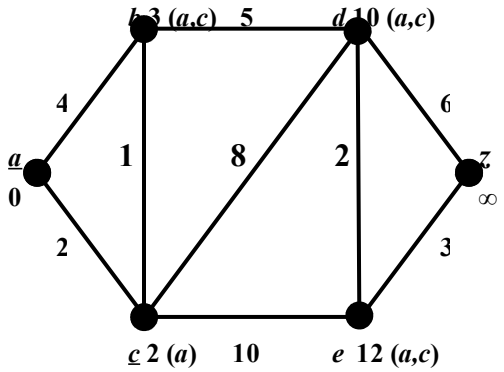
Exemplu:



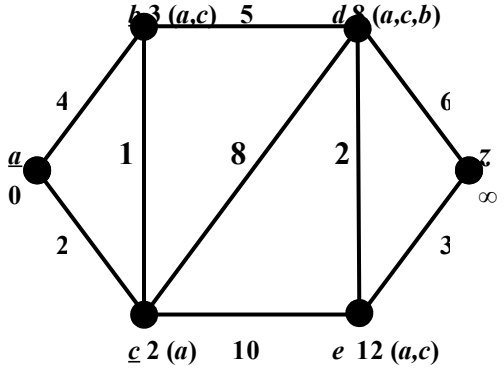

Pasul 0



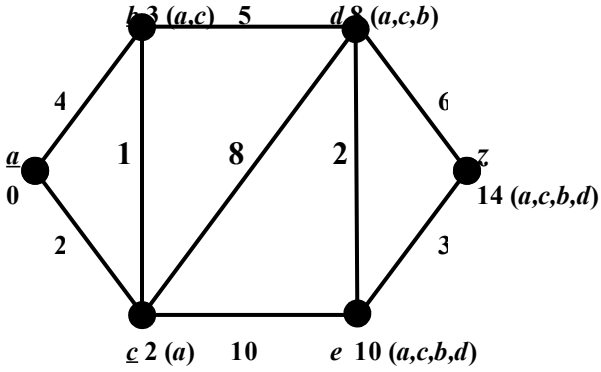

Pasul 1



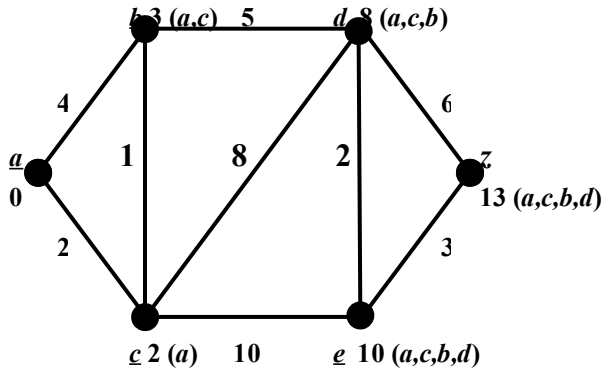

Pasul 2



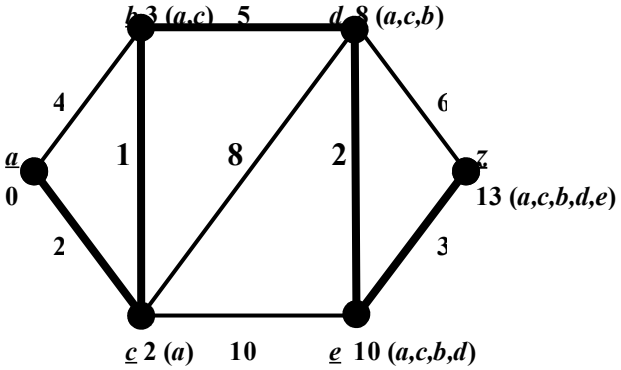

Pasul 3




Pasul 4




Pasul 5



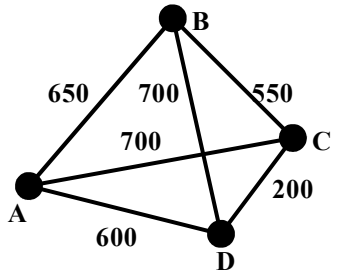

Pasul 6

**Teorema 12.5**: Algoritmul lui Dijkstra stabileste lungimea unui cel mai scurt drum între două noduri ale unui graf simplu, neorientat, conex, cu ponderi.
**Teorema 12.6**: Pentru stabilirea unui cel mai scurt drum între două noduri ale unui graf simplu, neorientat, conex, cu ponderi, algoritmul lui Dijkstra utilizează $O(n^2)$ operatii (adunări si comparări).
Se propune cititorului numărarea acestor operatii si verificarea acestui $O(n^2)$.

**Problema voiajorului comercial**

Problema voiajorului comercial este una din problemele clasice din stiinta calculatoarelor. Un voiajor comercial are de vizitat un număr de localităti cu revenire în localitatea de plecare. Desigur, parcursul trebuie făcut cu economie de timp si energie si de aceea drumul cel mai scurt este cel ideal. Localitătile si distantele dintre ele pot fi reprezentate ca un graf neorientat, complet si cu ponderi pe arcele sale. Problema este a stabili un circuit cu suma ponderilor minimă, circuit care să treacă prin toate nodurile grafului.

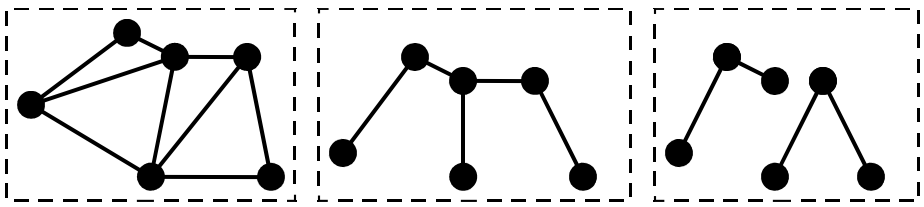
Un exemplu: Care este circuitul cel mai scurt între orasele A, B, C, D legate prin sosele de lungimile mentionate pe arcele grafului din figură?




Răspunsul: A, D, C, B, A, 2000 de unităti.
Se pune întrebarea: Fiind date $n$ noduri, câte cicluri diferite $C_n$ se pot forma prin conectarea acestor noduri prin arce? Răspunsul se obtine printr-o operatie de numărare: se alege un nod de plecare; sunt $(n - 1)$ alegeri pentru al doilea nod pe circuit, rămân $(n - 2)$ alegeri pentru al treilea nod s.a.m.d. astfel încât până la urmă sunt $(n - 1)!$ cicluri posibile. Sunt însă printre acestea cicluri identice, diferite numai prin sensul de parcurgere. Asadar, numărul de cicluri diferite $C_n$ este $(n - 1)!/2$.
Din nefericire, până acum nu s-a realizat un algoritm de rezolvare a problemei voiajorului comercial într-un timp în cel mai rău caz polinomial. Asta înseamnă că rezolvarea problemei pentru număr de noduri mare este nepractic(abil)ă. În aceste cazuri se pot utiliza algoritmi de aproximare, care să determine un drum a cărui lunigme ar putea fi întrucâtva mai mare decât cea minimă dar poate fi calculată la o complexitate temporală polinomială. Retelele neuronale artificiale pot executa o asemenea aproximare eficientă.

**Arbori**

**Definitia 12.28**: Un *arbore* este un graf conex neorientat care nu are circuite simple. Deoarece un arbore nu poate avea circuite, un arbore nu poate avea bucle sau arce multiple. Asadar, un arbore nu poate fi decât un graf simplu.
**Teorema 12.7**: Un graf neorientat este un arbore dacă si numai dacă între oricare două noduri există un drum simplu unic.
Exemplele de mai jos, în număr de trei fac diferenta dintre un arbore si structuri care sunt grafuri dar nu sunt arbori: singurul arbore este graful din centru.



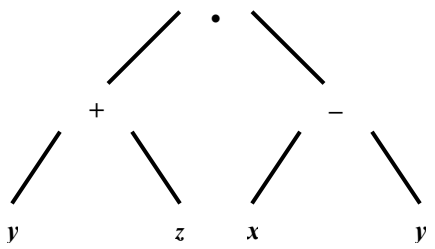
**Definitia 12.29**: Un graf neorientat care nu contine circuite simple si nu este neapărat conex este o *pădure.*
Arborii sunt utilizati de obicei pentru a reprezenta structuri ierarhice. Uneori, un anumit nod este calificat ca *rădăcină*. Deoarece există un drum unic de la rădăcină la fiecare nod al grafului, fiecare arc poate fi orientat de la rădăcină spre alte noduri. Astfel, un arbore si rădăcina sa produc un graf orientat denumit *arbore cu rădăcină*.

**Terminologia relativă la arbori**

Dacă *v* este un nod al unui arbore cu rădăcină, diferit de rădăcină, nodul unic care îl precede *u* este *părintele* lui *v*. Există, asadar, un arc orientat de la *u* la *v*.
Dacă *u* este părintele lui *v*, atunci *v* este *copilul* (urmasul, succesorul) lui *u*.
Nodurile cu acelasi părinte sunt denumite *surori* (frati).
*Înaintasii* (ancestorii) unui nod, altii decât rădăcina, sunt nodurile de pe drumul de la rădăcină la acel nod, excluzând acel nod dar incluzând rădăcina.
*Decendentii* uni nod *v* sunt acele noduri care îl au pe *v* ca înaintas.
Un nod dintr-un arbore cu rădăcină, care nu are descendenti este *frunză*.
Nodurile care au copii (succesori) sunt denumite noduri *interne*.
Dacă un nod *a* este într-un arbore, subarborele cu *a* ca rădăcină este subgraful din arbore care contine toti descendentii lui *a* si toate arcele incidente acestor descendenti.
*Nivelul* unui nod *v* într-un arbore cu rădăcină este lungimea drumului unic de la rădăcină la acel nod.
Nivelul rădăcinii este definit ca nivelul *zero*.
*Înăltimea* unui arbore este nivelul maxim al nodurilor sale.

**Exemple de arbori**

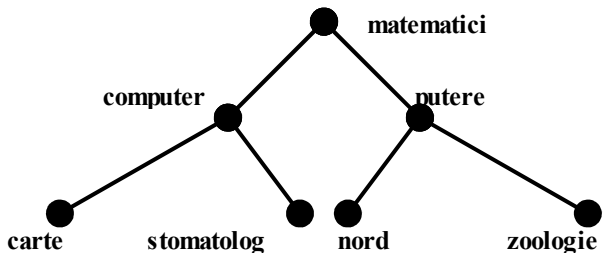
Arborii genealogici, arborele unui sistem de foldere (folder, foldere în folder, foldere în folder, …, fisiere), expresiile aritmetice.
Nu este greu de imaginat un arbore genealogic sau o structură arborescentă de foldere si fisiere. Pentru o expresia aritmetică, cum este de pildă $(y + z)\bullet(x - y)$, arborele arată ca în figura alăturată.



**Definitia 12.30**: Un arbore cu rădăcină este un arbore *m*-ar dacă oricare nod intern are nu mai mult de *m* urmasi. Un arbore este un arbore *m*-ar *complet* dacă fiecare nod intern are exact *m* descendenti. Pentru $m = 2$, arborii acestia sunt binari.
**Teorema 12.8**: Un arbore cu *n* noduri are $(n - 1)$ arce.

### Arbori de căutare binari

În cazul în care trebuie identificat un articol într-o listă cuprinzătoare, s-ar putea ca aranjarea acestei liste de articole în forma unui *arbore binar de căutare* să faciliteze identificarea urmărită.
Un arbore de căutare binar este un arbore binar în care fiecare descendent al unui nod este desemnat ca *de stânga* sau *de dreapta* si fiecare nod este etichetat cu o cheie care este în fond un articol.
Când se construieste arborele, nodurilor li se atribuie chei astfel încât cheia unui nod este mai mare decât cheile tuturor nodurilor din subarborele lui din stânga si mai mică decât cheile tuturor nodurilor din subarborele său din dreapta.
Exemplu: Constructia unui arbore binar de căutare pentru sirurile matematici, computer, putere, nord, zoologie, stomatolog, carte.




Pentru a efectua o căutare a articolului *x*, se porneste din nodul rădăcină prin compararea cheii lui *x* cu cheia nodului de pronire. Dacă *x* este mai mic decât cheia rădăcinii, mergem spre descendentul din stânga; dacă *x* este mai mare decât cheia rădăcinii mergem la descendentul din dreapta. Procedura este repetată până când articolul este identificat sau până când continuarea căutării devine imposibilă.
Într-un arbore echilibrat care reprezintă *n* articole, căutarea poate fi făcută în cel mult $\lceil \log(n+1) \rceil$ pasi.

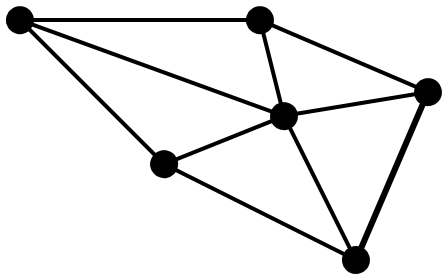
### Aplicatii ale arborilor

Aplicatiile arborilor sunt numeroase si importante. Iată câteva dintre ele:
- Optimizarea retelelor cu arborele acoperitor minimal
- Rezolvarea de probleme prin parcursul invers al arborilor de decizie

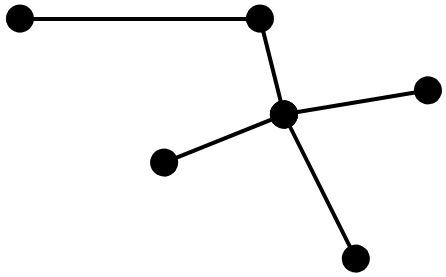
- Compresia de date cu coduri Huffman

**Arbori acoperitori**

**Definitia 12.31**: Fie **G** un graf simplu. Un arbore acoperitor pentru **G** este un subgraf al lui **G** care este un arbore cu toate nodurile grafului incluse.
Un arbore acoperitor pentru graful **G** = (**V**, **E**) este un graf conex pe **V** cu un număr minim de arce (|**V**| – 1). Sunt, desigur, mai multi arbori acoperitori.
Exemplu: O retea de alei între câteva puncte de interes cultural dintr-un parc



si un arbore acoperitor



Dacă s-ar pune problema acoperirii unor alei pentru a proteja vizitatorii pe vreme rea, acoperirea aleilor figurate în arborele acoperitor ar fi fără îndoială mai ieftină decât acoperirea tuturor aleilor si vizitatorii ar fi protejati în deplasările lor fără să rateze vreunul din punctele parcului.
Dacă se asociază lungimile aleilor ca ponderi ale arcelor, se poate formula cerinta de a realiza o lungime minimă a aleilor acoperite, un arbore de lungime minimă. Un astfel de arbore se numeste arbore acoperitor minimal.
Cum se poate stabili un arbore acoperitor minimal? Există, de pildă
**Algoritmul lui Prim**

- Se alege un arc cu cea mai mică lungime care este inclus primul în arborele minimal
- Se adaugă succesiv la arborele minimal arce de pondere minimă care sunt incidente nodurilor deja trecute îm arborele minimal, care nu fac cicluri simple cu arcele din arborele minimal

- Se opreste operatia când s-au adunat în arborele minimal ($n - 1$) arce ($n$ este numărul de noduri din graf).

**Algoritmul lui Kruskal**

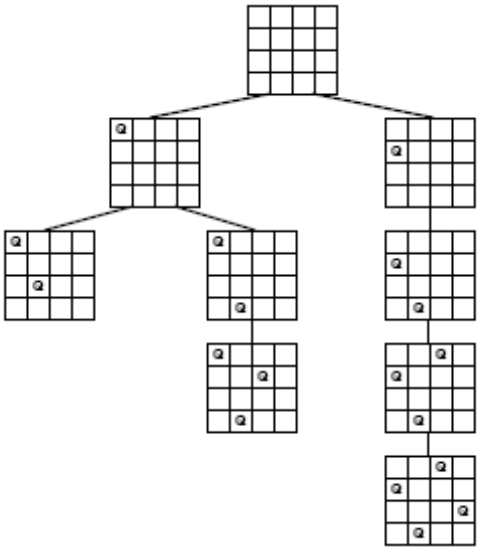
Algoritmul lui Kruskal este aproape identic cu algoritmul lui Prim. Diferenta constă în absenta cerintei de a introduce arce noi care să fie incidente unor noduri deja incluse în arborele minimal.
Ambii algoritmi sunt capabili să producă un arbore acoperitor minimal garantat.

**Parcurgerea inversă a arborilor de decizie**

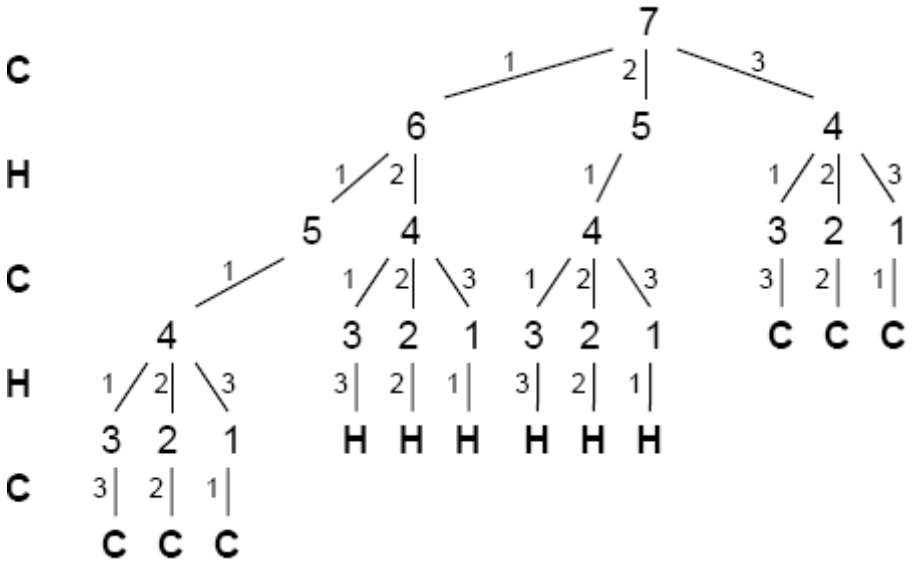
Un arbore decizional este un arbore cu rădăcină în care fiecare nod interior corespunde unei decizii, cu un subarbore al acestor noduri pentru fiecare rezultat posibil al deciziei. Arborii de decizie pot fi utilizati pentru a modela probleme care printr-o serie de decizii conduc la o solutie (a se compara exemplele cu problema monedei contrafăcute si cu arborele de căutare binar).
Solutiile posibile ale problemei corespund unui drum de la rădăcină la frunzele arborelui decizional.
Sunt probleme care pentru solutionare cer o căutare exhaustivă pe toate secventele de decizii. Aceste probleme se pot rezolva prin construirea unui arbore decizional complet în care apoi se stabileste un drum de la rădăcină la una din frunze. În multe cazuri, eficienta acestei proceduri poate fi îmbunătătită dramatic printr-o tehnică numită a parcursului invers.
Ideea este de a pleca de la rădăcină si de a merge în jos, adică a face o secventă de decizii până când fie se obtine o solutie, fie se atinge o situatie de la care nici o solutie nu mai poate fi atinsă prin orice secventă de decizii. În situatia din urmă, se reia parcursul înapoi (backtrack) spre părintele nodului curent si se continuă în jos din acel nod. Dacă toate drumurile din acel nod au fost deja explorate, se merge încă mai înapoi la părintele nodului de mai devreme. Se continuă astfel până ce se găseste o solutie sau se stabileste că nu există solutie (nu mai sunt drumuri de încercat).
Exemplu: Problema celor $n$ regine.
Cum se pot amplasa $n$ regine pe o tablă de sah $n$x$n$ astfel încât să nu existe vreo pereche de regina care să se poată captura reciproc.
O regină se poate deplasa oricâte pătrate pe orizontală, pe verticală si pe directiile diagonale ale tablei.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | x | | x | | x | |
| | | | x | x | x | | |
| x | x | x | x | Q | x | x | x |
| | | | x | x | x | | |
| | | x | | x | | x | |
| | x | | | x | | | x |
| x | | | | x | | | |
| | | | | x | | | |

Este aproape evident că solutia problemei celor *n* regine trebuie să fie cu exact o regină în fiecare coloană a tablei. Asadar, solutia acestei probleme poate fi descrisă ca o secventă de decizii:
Decizia 1: Se asează o regină în prima coloană
Decizia 2: Se asează o regină în a doua coloană
. . . . . .
Decizia *n*: Se asează o regină în a *n*-a coloană
Iată cum se rezolvă problema pentru *n* = 4.



Parcursul invers poate fi implementat ca un program destinat a juca împotriva unui oponent uman iată un joc extrem de simplu si de aceea nu foarte atractiv.



La începutul jocului sunt pe masă sapte monede. Jucătorul 1 face prima miscare, apoi jucătorul 2, din nou jucătorul 1 s.a.m.d. O miscare înseamnă a lua de pe masă 1, 2 sau 3 monede. Jucătorul care face ultima miscare este câstigător.

Să presupunem că prima mutare o are calculatorul. Atunci jocul poate fi considerat o succesiune de decizii cu prima decizie făcută de calculator, a doua de jucătorul uman, a treia de calculator s.a.m.d până când monedele sunt epuizate. Calculatorul "vrea" să ia decizii care să-l ducă la câstigarea partidei în acest joc rudimentar. Se admite tacit că jucătorul uman face totdeauna miscarea optimă.

Conform arborelui din figură (C – mutări ale calcualtorului, H – mutări ale omului (*human*)), calculatorul va începe cu a lua trei monede ceea ce îi garantează câstigul.

Pentru cazul jocurilor mai complexe, cum este de pildă sahul, este extrem de greu a verifica fiecare secventă de mutări posibilă. Jucătorul-calculator examinează în avans numai un anumit număr de mutări si estimează sansa de câstig pentru fiecare secventă posibilă.

# Lecția 13

## Introducere în probabilități

Subiectul sectiunii a treia a acestui curs, o sectiune majoră, îl constituie probabilitătile. Se urmăreste întelegerea unor declaratii ca acelea care urmează:

1. "Sunt 30% sanse ca un cutremur de pământ de peste 6 pe scara Richter să se petreacă în Vrancea în următorii 5 ani"
2. "Durata medie între două defectări ale sistemului este de circa 5 zile"
3. "Sansa de a avea o culoare la jocul de poker cu 5 cărti este de circa 1 la 500"
4. "În această schemă de echilibrare a sarcinii, probabilitatea ca un procesor să trateze mai mult de 12 solicitări este neglijabilă"

În toate aceste declaratii si în multe altele asemănătoare se sesizează implicit un ceva anume subiacent probabilităților. Acesta poate fi rezultatul unui model al lumii reale pe care îl confectionăm (ca în cazurile 1 si 2) sau al unui experiment pe care tot noi îl efectuăm (ca în cazurile 3 si 4). Nici una din declaratiile de mai sus nu are sens fără a specifica spatiul probabilităților la care se referă: pentru acest motiv, declaratiile ca aceea de tipul 1 (tipic făcută fără a crea acest context) sunt aproape independente de context.

## Spatii probabilistice

Un spatiu probabilistic se bazează pe un *experiment* aleator: aruncarea unui zar, amestecarea unui pachet de cărti si extragerea uneia dintre ele, extragerea unui număr la o loterie, atribuirea unui job unor procesoare, functionarea unui sistem etc. În loc de a încerca definirea directă a unui experiment, se defineste mai curând multimea de rezultate posibile ale acelui experiment, pe care o vom numi *spatiu al evenimentelor*. De retinut că rezultatele care interesează în primul rând sunt cele mutual exclusive elementare (atomice) si ele trebuie să acopere toate posibilitătile.

**Definitia 13.1** (**spatiul evenimetelor**): Spatiul evenimentelor relative la un experiment este multimea tuturor rezultatelor posibile ale acelui experiment.

**Definitia 13.2** (**spatiul probabilistic**): Un spatiu al probabilităților este un spatiu al evenimentelor atomice $\Omega$ împreună cu o *probabilitate* $\Pr(\omega)$ pentru fiecare eveniment $\omega$ astfel ca

- $0 \le \Pr(\omega) \le 1$ pentru orice $\omega \in \Omega$
- $\sum_{\omega \in \Omega} \Pr(\omega) = 1$, adică suma probabilităților rezultatelor $\omega$ este 1.

Strict vorbind, ce s-a definit mai sus este o multime restrânsă de spatii probabilistice cunoscute ca spatii *discrete*: asta înseamnă că multimea rezultatelor este finită sau numerabilă (cum sunt multimea numerelor naturale, multimea numerelor rationale, dar nu multimea numerelor reale). Mai departe se va vorbi sumar si despre spatiile probabilistice continue, dar deocamdată vorbim numai de spatii discrete.

Iată câteva exemple de spatii probabilistice discrete:

1. Aruncarea unei monede. Spatiul evenimentelor este $\mathbf{\Omega} = \{S, R\}$ – Stemă si Revers – si Pr(S) = Pr(R) = 1/2, admitând că moneda este corectă.
2. Aruncarea unei monede de trei ori. Aici $\mathbf{\Omega} = \{(t_1, t_2, t_3): t_i \in \{S, R\}\}$ cu $t_i$ rezultatul aruncării $i = 1, 2, 3$. Multimea $\mathbf{\Omega}$ are 8 elemente, fiecare are probabilitatea de 1/8. Mai general, dacă moneda este aruncată de $n$ ori, spatiul evenimentelor are dimensiunea $2^n$ (corespunzătoare tuturor cuvintelor de lungime $n$ alcătuite pe alfabetul $\{S, R\}$) si fiecare eveniment are probabilitatea $1/2^n$.
3. Aruncarea unei monede incorecte. Se presupune că dezechilibrul este de 2 la 1 adică stema apare cu probabilitatea 2/3, reversul apare cu probabilitatea 1/3. Spatiul evenimentelor este acelasi ca în exemplul anterior, dar probabilitătile sunt diferite. De exemplu, $\Pr(SSS) = (2/3)^3 = 8/27$, iar $\Pr(RSS) = (1/3)(2/3)^2 = 4/27$ (Notă: Am multiplicat aici într-o veselie probabilităti. Explicatia pentru această procedură va fi dată mai departe. Nu totdeauna procedura este corectă). Mai general, dacă aruncăm de $n$ ori în secventă o monedă incorectă, cu probabiluitătile $p$ si $1 - p$ pentru stemă si pentru reversul stemei, probabilitatea unei anumite secvente cu $r$ aparitii ale stemei este $p^r(1 - p)^{n-r}$. Secventa aruncării unei monede incorecte apare în multe contexte: de pildă, ea poate modela comportarea în $n$ încercări a unui sistem cu defecte, sistem care poate reactiona gresit, de fiecare dată cu o probabilitate $p$.
4. Aruncarea a două zaruri (unul rosu, altul albastru). Multimea $\mathbf{\Omega} = \{(i, j): 1 \leq i, j \leq 6\}$. Fiecare din cele 36 de rezultate au aceeasi probabilitate: 1/36.
5. Aruncarea a două zaruri indistincte (ambele albastre, de pildă). Aici $\mathbf{\Omega} = \{(i, j): 1 \leq i \leq j \leq 6\}$. Sunt 21 de rezultate (de ce?); cele care sunt de tipul $(i, i)$ au probabilitatea 1/36, celelalte au probabilitatea 2/36 = 1/18 (Cum s-au obtinut aceste probabilităti?). Când zarurile nu se deosebesc, suntem obligati a utiliza alt spatiu al evenimentelor, cel indicat. Acelasi spatiu se poate utiliza si la modelarea jocului de table, chiar dacă zarurile sunt colorate diferit, deoarece interesează numerele si nu ordinea lor.
6. Cărti amestecate. Un pachet de cărti amestecat are $|\mathbf{\Omega}| = 52!$ adică toate permutările de 52 de obiecte (cărti). Fiecare permutare are probabilitatea 1/52!. Remarcă: Aici vorbim de un model matematic ideal al unui pachet de cărti care poate fi amestecat perfect; în realitate, totdeauna există putin *bias* (iregularitate sistematică) în procesul de amestecare. Modelul este totusi destul de apropiat de realitate, deci este util si utilizabil.

7. Mâini la poker. Amestecarea urmată de distribuirea une mâini de poker. Multimea $\mathbf{\Omega}$ constă din toate formatiile de 5 cărti posibile (se admite din nou o amestecare ideală). Numărul de formatii total este de $C_{52}^5$ (alte notatii deplin echivalente: $\binom{52}{5}$ sau C(52, 5)), adică numărul de moduri în care se pot extrage 5 cărti din pachet fără a tine seamă de ordinea extragerii. Dar $C_{52}^5 = \frac{52!}{5!47!} = 2.598.960$. Experientele de acest gen în care un număr de elemente distincte sunt extrase dintr-o multime de cardinal $n$ este cunoscută uzual ca "sondare/explorare fără înlocuire".
8. Extragerea de premii. $n$ competitori intră într-o tragere la sorti; sunt luati la întâmplare trei câstigători distincti în ordine (primul, al doilea, al treilea). Aici, multimea $\mathbf{\Omega}$ este alcătuită din triple de forma $(i, j, k)$ cu $i, j, k \in \{1, \ldots, n\}$ si distincte. Cardinalul $|\mathbf{\Omega}|$ este egal cu numărul de moduri în care se pot extrage trei elemente din cele $n$, în ordine: acest număr s-a notat într-o lectie anterioară cu $P(n, 3)$ si se calculează cu relatia $P(n, k) = n!/(n - k)!$ ceea ce înseamnă în cazul în discutie $n(n - 1)(n - 2)$. Este un alt exemplu de "sondare fără înlocuire", cu deosebirea că de data aceasta contează si ordinea.
9. Bile si cutii. Se aruncă 20 de bile distincte în 10 cutii distincte, astfel încât fiecare bilă are sanse egale de a ateriza în oricare cutie, independent de ceea ce se întâmplă cu celelalte bile. Aici $\mathbf{\Omega} = \{(b_1, b_2, \ldots, b_{20}): 1 \le b_i \le 10\}$; componenta $b_i$ reprezintă cutia în care aterizează bila $i$. Sunt $10^{20}$ rezultate posibile (de ce?), fiecare cu probabilitatea $1/10^{20}$. Mai general, dacă se aruncă $m$ bile în $n$ cutii, spatiul evenimentelor are dimensiunea $n^m$. (Un caz special este Exemplul 2 de mai sus cu $m = 3$, $n = 2$). Acela este un exemplu de "sondare *cu* înlocuire" deoarece alegerile făcute de bile nu trebuie să fie distincte. Cum se va vedea, sistemul bile-si-cutii este un alt spatiu probabilistic care apare foarte frecvent în stiinta calculatoarelor: de exemplu, ne putem gândi la echilibrarea încărcării cu sarcini de calcul a mai multor procesoare, prin repartizarea aleatoare a job-urilor pe procesoare.
10. Bile si cutii cu bilele nedistincte. Se presupune că se aruncă $m$ bile în $n$ cutii, dar bilele sunt asemănătoare, chiar identice. Asta înseamnă că după aruncarea bilelor nu interesează decât numărul bilelor în fiecare cutie, nu care anume bile au aterizat într-o cutie sau alta. Astfel, punctul din spatiul probelor este un $n$-uplu $(m_1, m_2, \ldots, m_n)$, în care $m_i$ este numărul de bile în cutia $i$, adică $\mathbf{\Omega}$ constă în astfel de $n$-uple în care $m_i$ sunt întregi nenegativi care însumati dau $m$. Câte puncte sunt în $\mathbf{\Omega}$? Atâtea câte moduri de partitionare a celor $m$ obiecte (bile) în $n$ grupe sunt. Despre aceste partitii se poate gândi astfel: se figurează $m$ puncte în linie (care reprezintă obiectele); apoi se iau $n - 1$ linii verticale care se intercalează cumva cu punctele. Orice aranjament de puncte si linii de acest gen reprezintă o partitie unică după cum urmează: prin lectura de la stânga la dreapta, numărul de obiecte din

prima grupă, $m_1$, este numărul de puncte de la stânga primei linii. Numărul de obiecte din a doua grupă, $m_2$, este numărul de puncte dintre linia primă si linia a doua s.a.m.d. Ar trebui să fie destul de clar că fiecare partitie este reprezentată de unul si numai de unul dintre aranjamentele puncte-linii. Asadar, numărul de partitii este egal cu numărul de astfel de aranjamente. Dar numărul de aranjamente poate fi calculat ca $C_{m+n-1}^{n-1}$ deoarece avem de ales pozitiile celor $n - 1$ linii din totalul de $m + n - 1$ pozitii posibile (de reflectat asupra acestei afirmatii).

În acest spatiu al probelor, probabilitătile punctelor *nu* sunt toate egale. De exemplu, dacă sunt două bile si două cutii, aranjamentul (2, 0) cu ambele bile în cuita 1 se poate întâmpla într-un singur mod pe când aranjamentul (1, 1) se poate întâmpla în două moduri diferite (schimbând cele două bile ajungem la acelasi aranjament). În general, în câte moduri se poate obtine aranjamentul ($m_1$, $m_2$, …, $m_n$)? Prin generalizarea unei demonstratii date mai demult relativ la numărul de combinări se poate formula răspunsul: $\frac{m!}{m_1!m_2!...m_n!}$ moduri. În consecintă, probabilitatea acestui punct din spatiul probelor este $\Pr(\omega) = \frac{1}{n^m}\frac{m!}{m_1!m_2!...m_n!}$

11. Problema Monty Hall. Într-un joc de prin anii ’60 numit “Let’s Make a Deal”, (gazdă era un anume Monty Hall) unui concurent i se arătau trei usi; dincolo de una din usi era un premiu consistent si dincolo de celelalte două erau nimicuri. Concurentul alegea o usă dar n-o deschidea si Monty deschidea una din celelalte două usi demascând unul din nimicuri. Concurentului i se oferea posibilitatea să rămână la prima optiune sau să schimbe la cealaltă usă închisă. El câstiga dacă si numai dacă usa aleasă era cea corectă. Întrebarea este: avea concurentul o sansă mai mare de a câstiga dacă schimba optiunea?

    Care este aici spatiul probelor? Rezultatele jocului pot fi descrise (până în punctul unde concurentul face decizia finală) folosind o triplă de forma ($i$, $j$, $k$) cu $i, j, k \in \{1, 2, 3\}$. Valorile $i$, $j$, $k$ numesc respectiv usa dincolo de care este premiul, usa initială aleasă de concurent, usa deschisă de Monty. De observat că unele triple nu sunt posibile, de pildă (1, 2, 1), deoarece Monty nu deschidea niciodată usa cu premiul cel mare. Gândind spatiul probelor ca o structură arborescentă în care se alege mai întâi $i$, apoi $j$ si în final $k$ (depinzând de $i$ si $j$) vom descoperi exact 12 puncte.

    Atribuirea de probabilităti punctelor spatiului cere fixarea unor presupuneri:

    - Cu probabilităti egale, premiul poate fi în spatele oricărei usi
    - Initial, concurentul poate alege oricare dintre cele trei usi cu probabilităti egale
    - Dacă de întâmplă ca un concurent să aleagă usa cu premiul (astfel rămân pentru Monty două usi posibil de deschis), Monti alege una din ele cu sanse egale.

Acum se pot atribui probabilități fiecărei punct al spatiului probelor. De pildă, punctul (1, 2, 3) corespunde plasării premiului în spatele usii 1 (cu probabilitatea 1/3), concurentul alege usa 2 (cu probabilitatea 1/3) si monty deschide usa 3 (cu probabilitatea 1 deoarece nu are altă alegere). Astfel

$$\Pr[(1,2,3)] = \frac{1}{3}.\frac{1}{3}.1 = \frac{1}{9}$$

(Notă: Din nou se face produsul unor probabilități fără o justificare solidă). Sunt sase rezultate de tipul acesta, caracterizat prin $i \neq j$ (si cu $k$ diferit de ambele). Pe de altă parte avem

$$\Pr[(1,1,3)] = \frac{1}{3}.\frac{1}{3}.\frac{1}{2} = \frac{1}{18}$$

si sunt sase rezultate de acest tip, cu $i = j$. Acestea sunt toate rezultatele posibile, asa că spatiul probabilistic este deplin definit. Pentru a verifica aritmetica noastră, se face suma tuturor probabilităților care trebuie să fie 1 si este egală cu 1.

$$\left(6.\frac{1}{9}\right) + \left(6.\frac{1}{18}\right) = 1$$

**Evenimente**

În problema lui Monty, ne interesează probabilitatea ca premiul să fie câstigat de concurent. Acesta nu este un rezultat unic (concurentul poate câstiga în mai multe moduri diferite) ci *o multime* de rezultate. De aici:

**Definita 13.3** (**eveniment**): Un eveniment $A$ din spatiul $\mathbf{\Omega}$ este orice submultime $A \subseteq \mathbf{\Omega}$.

Cum trebuie definită probabilitatea unui eveniment $A$? Natural, trebuie adunate probabilitătile punctelor spatiului $\mathbf{\Omega}$ cuprinse în $A$.

**Definitia 13.4** (**probabilitatea unui eveniment**): Pentru orice eveniment $A \subseteq \mathbf{\Omega}$, probabilitatea se defineste ca

$$\Pr[A] = \sum_{\omega \in A} \Pr[\omega]$$

O privire asupra unor exemple recoltate din spatiile probabilistice exemplificate mai devreme.

1. Moneda corectă. Fie $A$ evenimentul "la aruncare rezultă stema". $\Pr[A] = 1/2$.
2. Trei monede corecte. Fie $A$ evenimentul în care cele trei monende afisează aceeasi fată. În acest caz, $\Pr[A] = \Pr[SSS] + \Pr[RRR] = 1/8 + 1/8 = 1/4$.
3. Monede cu defect. Fie A acelasi eveniment ca în exemplul anterior. În acest caz diferit, $\Pr[A] = \Pr[SSS] + \Pr[RRR] = 8/27 + 1/27 = 1/3$. Ca un al doilea exemplu, fie $B$ evenimentul în care sunt exact două steme. Se stie că probabilitatea oricărui rezultat cu două si numai două steme este $(2/3)^2.(1/3) = 4/27$. Câte asemenea rezultate sunt în total? Sunt $C_3^2 = 3$ moduri de a alege ordinea stemelor si aceste alegeri epuizează complet specificarea

secventei. Astfel, Pr[$B$] = 3.(4/27) = 4/9. Mai general, probabilitatea de a avea exact $r$ steme din $n$ aruncări ale monedei cu defect este $C_n^r p^r (1-p)^{n-r}$, cu $p$ probabilitatea unei steme la o aruncare.

4. Zarul. Fie $A$ evenimentul ca suma a două zaruri să fie cel putin 10 si $B$ evenimentul ca să apară cel putin un 6. Pr[$A$] = 6/36 = 1/6 si Pr [$B$] = 11/36. În acest exemplu (ca si în 1 si 2 de mai devreme) spatiul probabilităților este uniform, adică toate punctele spatiului probelor au *aceeasi* probabilitate (care trebuie să fie 1/|**Ω**|, valoarea reciprocă a cardinalului multimii **Ω**). În asemenea împrejurări, probabilitatea unui eveniment $A$ este cu claritate Pr[$A$] = (# de atomi din $A$)/(# de atomi din **Ω**) = |$A$|/|**Ω**| (atomi este totuna cu puncte în spatiul probelor).
   Asadar, pentru spatii uniforme, caclulul probabilităților se reduce la *numărarea* atomilor!
6. Cărti amestecate. Fie $A$ evenimentul constând în aceea ca deasupra să fie un as. Din remarca anterioară putem scrie:
   Pr[$A$] = (# de permutări cu un as deasupra)/52!
   Câte permutări au un as deasupra? Sunt patru alegeri pentru as; odată asul ales si asezat deasupra, cărtile de sub el pot fi în 51! de asezări posibile. Astfel, numărul de permutări cerut este de 4.51! si probabilitatea asociată este Pr[$A$] = (4.51!)/52! = 4/52 = 1/13.
   Fie $B$ evenimentul descris ca "două cărti de deasupra sunt de aceeasi culoare". Câte permutări de acest gen sunt? Sunt patru culori posibile si odată culoarea aleasă sunt 13 cărti din care să facem alegerea pentru prima carte si 12 din care să alegem a doua carte. Restul de 50 de cărti pot fi asezate în 50! moduri. Astfel, Pr[$B$] = (4.13.12.50!)/52! = 12/51.
7. Mâini de poker. Care este probabilitatea ca o mână la poker să fie "culoare" (adică toate cărtile să fie de aceeasi culoare; pică, sau cupă, sau caro, sau treflă)? Trebuie calculat câte posibilităti de "culoare" sunt. Sunt 13 cărti în fiecare culoare, sunt $C_{13}^5$ mâini posibile în acea culoare. Numărul total de "culori" este asadar $4C_{13}^5$. Si mai departe Pr["culoare"] = $4C_{13}^5 / C_{52}^5$ ≈ 0,002.
8. Extragerea de premii. Sunt unul din cei $n$ competitori. Fie $A$ evenimentul constând în a câstiga eu, nu altcineva, unul din premii. Câte puncte ale spatiului probelor sunt în $A$? Sunt trei premii pe care le pot câstiga; fiecare din acestea lasă două premii pentru alti doi competitori (în ordine) din ceilalti $n$ – 1 si sunt P($n$ – 1, 2) = ($n$ – 1)($n$ – 2) posibilităti. Astfel, |$A$| = 3($n$ – 1)($n$ – 2). Asa se ajunge la Pr[$A$] = |$A$|/|**Ω**| = [3($n$ – 1)($n$ – 2)]/[ $n$($n$ – 1)($n$ – 2)] = 3/$n$.
9. Bile si cutii. Fie $A$ evenimentul definit ca "o cutie anumită (de pildă prima) rămâne goală". Din nou, se face operatia de numărare a rezultatelor cu această proprietate. Si acestea sunt exact numărul modurilor în care cele 20 de bile pot cădea în celelalte 9 cutii, adică $9^{20}$. Prin urmare, Pr[$A$] = $9^{20}/10^{20}$ = $(9/10)^{20}$ ≈ 0,12. Care este probabilitatea ca prima cutie să conțină cel putin

o bilă? Este usor de calculat: se pune $\overline{A}$ *contrarul* (*complementul*) lui $A$, adică evenimentul "în prima cutie cade cel putin o bilă" si contine exact ceea ce nu contine $A$. Asadar, $\Pr[\overline{A}] = 1 - \Pr[A] \approx 0{,}88$. Mai general, dacă se aruncă $m$ bile în $n$ cutii

$$\Pr[\text{"prima cutie rămâne goală"}] = \left(\frac{n-1}{n}\right)^m = \left(1-\frac{1}{n}\right)^m$$

11. Problema Monty Hall. Revenim la această problemă si la intentia de a investiga meritul relativ al schimbării sau mentinerii strategiei după deschiderea unei usi. Să admitem că decizia concurentului este de a schimba alegerea de usă. Evenimentul $A$ care interesează este cel care-l face pe concurent câstigător. Care puncte $(i, j, k)$ sunt în $A$? Deoarece concurentul schimbă usa, alegerea sa initială $j$ nu poate fi egală cu usa premiului, care este $i$. Si toate rezultatele de acest tip corespund câstigului deoarece Monty trebuie să deschidă a doua usă fără premiu, lăsând concurentul să comute la usa cu premiul. Astfel, $A$ constă în toate rezultatele de primul tip conform analizei făcute mai devreme; sunt sase astfel de rezultate, fiecare cu probabilitatea de 1/9. Asadar, $\Pr[A] = 6/9 = 2/3$, adică concurentul câstigă cu probabilitatea 2/3! Ar trebui să fie intuitiv clar (si usor de verificat formal – a se încerca!) că sub strategia non-comutării probabilitatea de a câstiga este 1/3 (în cazul acesta, concurentul alege realmente aleator numai o usa). Astfel, prin comutare, concurentul îsi măreste considerabil sansa de a câstiga.

    Este unul din exemplele care ilustrează cât de important este a calcula sistematic probabilităti si nu "intuitiv".

Recapitulăm pasii din calculele noastre:

- Care este spatiul probelor (experimentul si posibilele lui rezultate)?
- Care este probabilitatea fiecărui rezultat (fiecărui eveniment atomic)?
- Care este evenimentul care interesează (care submultime a spatiului evenimentelor elementare)?
- Calculul probabilitătii evenimentului prin adunarea de probabilităti ale evenimentelor elementare componente.

Ori de câte ori apare o problemă de probabilităti, trebuie mers înapoi la aceste elemente fundamentale pentru a evita orice capcană. Chiar si cei mai experimentati cercetători pot gresi când uită acest parcurs, marturie stau multele "demonstratii" eronate emise de matematicieni către ziarele vremii, conform cărora strategia schimbării în problema Monty Hall nu ar îmbunătăti sansele de câstig.

# Lecția 14

## Probabilități conditionate

O companie farmaceutică face marketingul unui nou produs de testare a unei anumite conditii de sănătate. Potrivit încercărilor clinice, testul cu produsul respectiv are proprietătile următoare:

1. Când se aplică unei persoane afectate, testul se dovedeste pozitiv în 90% din cazuri si negativ în 10% din cazuri (acestea din urmă sunt denumite "false negative")
2. Când se aplică persoanelor sănătoase testul apare negativ în 80% si pozitiv în 20% din cazuri (acestea sunt numite "false pozitive")

Se presupune că incidenta afectiunii în populatia unei tări este de 5%. Când o persoană luată la întâmplare este testată pozitiv, care este probabilitatea ca acea persoană să aibă conditia indicată de test? (De notat că aceasta este de presupus a nu fi aceeasi cu simpla probabilitate ca o persoană la întâmplare să aibă conditia, probabilitate care este de numai 1/20).

Probabilitatea cerută este un exemplu de *probabilitate conditionată*: suntem interesati în probabilitatea ca o persoană să aibă conditia (evenimentul $A$) dat fiind faptul că testul are rezultat pozitiv (evenimentul $B$). Asta se scrie $\Pr[A|B]$.

Cum se evaluează această probabilitate? Deoarece evenimentul $B$ se întâmplă garantat, trebuie examinat nu întreg spatiul probelor ci numai o parte a lui care constă în punctele din $B$. Care trebuie să fie probabilitătile acestor puncte? Dacă ele mostenesc pur si simplu probabilitătile din $\mathbf{\Omega}$, atunci suma acestor probabilități va fi $\Sigma_{\omega \in B} \Pr[\omega] = \Pr[B]$ care este în general mai mică decât 1. Astfel, scara fiecărui eveniment trebuie modificată cu $1/\Pr[B]$, adică pentru fiecare punct $\omega \in B$ noua probabilitate devine

$$\Pr[\omega|B] = \Pr[\omega]/\Pr[B]$$

Acum este clar cum se calculează $\Pr[A|B]$: se însumează aceste probabilități scalate pentru toate punctele care sunt si în $A$ si în $B$

$$\Pr[A \mid B] = \sum_{\omega \in A \cap B} \Pr[\omega \mid B] = \sum_{\omega \in A \cap B} \frac{\Pr[\omega]}{\Pr[B]} = \frac{\Pr[A \cap B]}{\Pr[B]}$$

**Definitia 14.1** (**probabilitatea conditionată**): Pentru evenimentele $A$, $B$ din acelasi spatiu probabilistic, cu $\Pr[B] > 0$, probabilitatea lui $A$ conditionată de $B$ este

$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]}$$

Să revenim la exemplul testării medicale. Spatiul probelor constă din toti cetătenii tării, în număr de $N$. Populatia se constituie din patru submultimi disjuncte:
$TP$ – cei adevărat pozitivi: 90% din $N/20$, adică $9N/200$;
$FP$ – cei fals pozitivi: 20% din $19N/20$, adică $19N/100$;
$TN$ – cei adevărat negativi: 80% din $19N/20$, adică $76N/100$;
$FN$ – cei fals negativi: 10% din $N/20$ adică, $N/200$.
Acum, fie $A$ evenimentul "o persoană aleasă la întâmplare este afectată" si $B$ evenimentul "acea persoană este testată cu rezultat pozitiv". De notat că $B$ este reuniunea multimilor disjuncte $TP$ si $FP$, asa că

$$|B| = |TP| + |FP| = 9N/200 + 19N/100 = 47N/200$$

si

$$\Pr[A] = 1/20,\ \Pr[B] = 47N/200$$

Acum, când se face conditionarea prin evenimentul $B$, focalizăm evaluarea pe spatiul restrâns de probe constând în acei $47N/200$ indivizi care s-au testat cu rezultat pozitiv. Pentru a calcula $\Pr[A|B]$, trebuie evaluată $\Pr[A \cap B]$ (partea din $A$ care se află în $B$). Dar $A \cap B$ este multimea persoanelor care sunt si afectate si au dat si test pozitiv, adică $A \cap B = TP$. Asadar, avem:

$$\Pr[A \cap B] = |TP|/N = 9/200$$

Finalmente, conchidem din definita 14.1 că

$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{9/200}{47/200} = \frac{9}{47} \approx 0{,}19$$

Asta arată destul de rău: dacă cineva are testul pozitiv, sunt numai cca. 19% sanse ca persoana respectivă să aibă în realitate conditia. Asta sună mai rău decât pretentiile initiale ale companiei farmaceutice, dar de fapt este numai un alt mod de a vedea aceleasi date.
Incidental, a se nota că

$$\Pr[B \mid A] = \frac{\Pr[A \cap B]}{\Pr[A]} = \frac{9/200}{1/20} = \frac{9}{10}$$

astfel că cele două probabilităti conditionate, $\Pr[A|B]$ si $\Pr[B|A]$ pot fi foarte diferite. Probabilitatea din urmă este probabilitatea ca o persoană să aibă test pozitiv si să aibă si conditia, ceea ce se stie de la început că este de 90%.
Pentru a completa imaginea, care este probabilitate (fără conditionare) ca testul să dea un rezultat corect (pozitiv sau negativ) când este aplicat unei persoane la întâmplare? Notăm acest eveniment cu $C$, si

$$\Pr[C] = \frac{|TP| + |TN|}{N} = \frac{9}{200} + \frac{76}{100} = \frac{161}{200} \approx 0{,}8$$

Asadar testul este în general eficient în proportie de cca. 80%, ceea ce dă o impresie mult mai bună.
Dar cât de bună este această impresie? Să presupunem că ignorăm testul si ne pronuntăm că toată lumea e sănătoasă. Afirmatia e corectă pentru 95% din populatie (pentru cei sănătosi) si gresită pentru cei 5% afectati. Adică acest test banal, trivial este eficient 95%! Se pune întrebarea dacă merită ca testul să fie efectuat. Ce crede cititorul?

Urmează acum alte câteva exemple de probabilităti conditionate, în legătură cu spatii ale probelor descrise în lectia anterioară.

1. **Bile si cutii**. Presupunem că aruncăm $m = 3$ bile în $n = 3$ cutii; spatiul acesta este uniform si are $3^3 = 27$ de puncte. Se cunoaste deja că probabilitatea ca prima cutie să fie goală este $(1 - 1/3)^3 = (2/3)^3 = 8/27$. Care este probabilitatea acestui eveniment *stiut fiind că* a doua cutie este goală? Să notăm aceste evenimente $A$, respectiv $B$. Pentru a calcula $\Pr[A|B]$ trebuie evaluată probabilitatea $\Pr[A \cap B]$. Dar intersectia celor două evenimente, prima si a doua cutie să fie goale, înseamnă ca toate cele trei bile să cadă în cutia a treia. Prin urmare, $\Pr[A \cap B] = 1/27$ (de ce?) si
$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{1/27}{8/27} = \frac{1}{8}$$
Observatia că fractia 1/8 este ceva mai mică decât 8/27 nu este o surpriză: faptul cunoscut că în cutia 2 nu este nici o bilă face semnificativ mai putin probabil ca nici în cutia 1 să nu fie vreo bilă.
2. **Zaruri**. Se aruncă două zaruri corecte. Fie $A$ evenimentul descris ca "suma punctelor afisate este pară" si $B$ evenimentul "primul zar afisează un număr par de puncte". Prin simetrie, este usor de observat că $\Pr[A] = 1/2$ si $\Pr[B] = 1/2$. Mai mult, putină numărătoare si ajungem la $\Pr[A \cap B] = 1/4$. Si
$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{1/4}{1/2} = \frac{1}{2}$$
În acest caz, $\Pr[A|B] = \Pr[A]$, adică conditionarea prin $B$ nu schimbă probabilitatea lui $A$.

**Evenimente independente**

**Definitia 14.2** (**independenta**): Două evenimente $A$, $B$ din acelasi spatiu probabilistic sunt independente dacă $\Pr[A|B] = \Pr[A]$.

De observat că independenta este simetrică: dacă $\Pr[A|B] = \Pr[A]$, atunci si $\Pr[B|A] = \Pr[B]$. Pentru a argumenta această afirmatie recurgem la definitia probabilităților conditionate
$$\Pr[B \mid A] = \frac{\Pr[A \cap B]}{\Pr[A]} = \frac{\Pr[A \cap B]}{\Pr[B]} \times \frac{\Pr[B]}{\Pr[A]} = \frac{\Pr[A \mid B]}{\Pr[A]} \times \Pr[B] = \Pr[B]$$
În ultima fază a demonstratiei am folosit faptul că $\Pr[A|B] = \Pr[A]$. (S-a presupus tot tot timpul că $\Pr[A]$ si $\Pr[B]$ sunt nenule, altminteri ar apărea nedeterminări).

În exemplele de mai sus, în cazul "bile si cutii" $A$ si $B$ nu sunt evenimente independente; în cazul "zaruri" $A$ si $B$ sunt independente.

A sti că evenimetele sunt independente este totdeauna foarte util deoarece

**Teorema 14.1**: Dacă evenimentele $A$, $B$ sunt independente atunci $\Pr[A \cap B] = \Pr[A].\Pr[B]$.

**Demonstratie**: Din definitia probabilităților conditionate avem
$$\Pr[A \cap B] = \Pr[A|B].\Pr[B] = \Pr[A].\Pr[B]$$
unde în pasul al doilea al secventei de egalități s-a utilizat independenta.

□

De retinut că relatia din teorema 14.1 se mentine *dacă si numai dacă* evenimentele $A$ si $B$ sunt independente. Uneori această relatie este dată ca definitie a independentei.

Cele de mai sus se generalizează la orice multime finită de evenimente:

**Definitia 14.3** (**independenta mutuală**): Evenimentele $A_1, \ldots, A_n$ sunt mutual independente dacă pentru orice $1 \le i \le n$ si pentru orice $I \subseteq \{1, \ldots, n\} - \{i\}$

$$\Pr[A_i \mid \bigcap_{j \in I} A_j] = \Pr[A_i]$$

adică probabilitatea lui $A_i$ nu depinde de nici o combinatie de alte evenimente.

**Teorema 14.2**: Dacă evenimentele $A_1, \ldots, A_n$ sunt mutual independente, atunci

$$\Pr[A_1 \cap \ldots \cap A_n] = \Pr[A_1] \times \Pr[A_2] \times \ldots \times \Pr[A_n]$$

Nu se va da o demonstratie a acestei teoreme deoarece ea este un caz special al teoremei 14.3, mai generale, care va fi demostrată mai jos. De retinut că se pot construi trei evenimente $A$, $B$, $C$ independente *în perechi* dar care *nu* sunt toate trei mutual independente.

## Combinatii de evenimente

În multe aplicatii ale probabilităților în automatizări si în stiinta calculatoarelor, apare interesant a calcula lucruri precum $\Pr[\bigcap_{i=1}^{n} A_i]$ sau $\Pr[\bigcup_{i=1}^{n} A_i]$ cu $A_i$ evenimente simple adică evenimente pentru care este usor a calcula $\Pr[A_i]$. Intersectia corespunde unui SI logic al evenimentelor $A_i$, reuniunea corespunde unui SAU logic. De pildă, $A_i$ poate fi un eveniment constând în defectarea într-un anumit mod a unei părti a unui sistem, reuniunea poate fi căderea totală a sistemului.

În general, calculul probabilităților unor astel de combinatii poate fi foarte dificil. Aici se discută unele situatii în care calculul poate fi făcut.

## Intersectia de evenimente

Din definitia probabilităților conditionate rezultă imediat următoarea *regulă a produsului* (uneori numită *regula lantului*) pentru calculul probabilității unei intersectii de evenimente.

**Teorema 14.3** (**Regula produsului**): Pentru evenimentele $A$, $B$ avem

$$\Pr[A \cap B] = \Pr[A].\Pr[B|A]$$

Mai general, pentru evenimetele $A_1, \ldots, A_n$

$$\Pr[\bigcap_{i=1}^{n} A_i] = \Pr[A_1] \times \Pr[A_2 \mid A_1] \times \Pr[A_3 \mid A_1 \cap A_2] \times \ldots \times \Pr[A_n \mid \bigcap_{i=1}^{n-1} A_i]$$

**Demonstratie**: Prima afirmatie rezultă direct din definitia probabilitătii conditionate (si este de fapt un caz special al afirmatiei secunde pentru $n = 2$).

Pentru a dovedi afirmatia a doua se foloseste inductia simplă după $n$, numărul de evenimente.

Cazul de bază este $n = 1$ si corespunde afirmatiei $\Pr[A] = \Pr[A]$, ceea ce este un adevăr trivial.
În pasul inductiv, fie $n > 1$ si presupunerea (ipoteza inductivă) că

$$\Pr[\bigcap_{i=1}^{n-1} A_i] = \Pr[A_1] \times \Pr[A_2 \mid A_1] \times \ldots \times \Pr[A_{n-1} \mid \bigcap_{i=1}^{n-2} A_i]$$

Acum aplicăm relatia de definitie a probabilitătii conditionate pentru două evenimente, $A_n$ si $\bigcap_{i=1}^{n-1} A_i$ , pentru a deduce că

$$\Pr[\bigcap_{i=1}^{n} A_i] = \Pr[A_n \cap \bigcap_{i=1}^{n-1} A_i] = \Pr[A_n \mid \bigcap_{i=1}^{n-1} A_i] \times \Pr[\bigcap_{i=1}^{n-1} A_i] =$$

$$= \Pr[A_n \mid \bigcap_{i=1}^{n-1} A_i] \times \Pr[A_1] \times \Pr[A_2 \mid A_1] \times \ldots \times \Pr[A_{n-1} \mid \bigcap_{i=1}^{n-2} A_i]$$

unde în ultima etapă s-a folosit ipoteza inductivă. Demonstratia este completă.
□
Teoremele 14.1 si 14.2 sunt cazuri speciale ale regulii produsului pentru evenimente *independente*.

**Secvente de încercări**

Multe experimente pot fi privite ca *secvente* de experiente simple sau de *încercări*. În aceste cazuri, este mai natural a defini spatiul probabilistic în termeni de probabilităti conditionate, uzând de regula produsului. Ca o ilustrare, se consideră spatiul $\Omega$ al aruncărilor de $n$ ori a unei monede cu defect, caz discutat în lectia anterioară. $\Omega$ se poate scrie ca un produs $\Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$, în care $\Omega_i = \{S, R\}$ este spatiul aruncării $i$ a acelei monede (avem aici multimea $n$-tuplelor ordonate de rezultate ale aruncărilor succesive ale monedei).
Cum se definesc probabilitătile în $\Omega$? Punctele în spatiul $\Omega$ sunt $n$-tuplele $\omega = (\omega_1, \omega_2, \ldots, \omega_n)$ unde $\omega_i \in \Omega_i$ este rezultatul celei de a $i$ aruncări. Utilizând regula produsului, trebuie să aibă loc

$$\Pr[\omega] = \Pr[\omega_1] \times \Pr[\omega_2|\omega_1] \times \ldots \times \Pr[\omega_n|\omega_1, \omega_2, \ldots, \omega_{n-1}]$$

Astfel, dacă se definesc toate probabilitătile conditionate $\Pr[\omega_i|\omega_1, \omega_2, \ldots, \omega_{i-1}]$, se defineste de fapt întreg spatiul probabilistic.
Acum elementul cheie în acest exemplu este acela că aruncările monedei sunt presupuse *independente* astfel încât fiecare probabilitate conditionată este exact aceea fără conditionare. Ecutia de mai sus devine

$$\Pr[\omega] = \Pr[\omega_1] \times \Pr[\omega_2] \times \ldots \times \Pr[\omega_n]$$

Asadar, probabilitatea unui punct $\omega$ din $\Omega$ este $\Pr[\omega] = p^r(1-p)^{n-r}$, cu $r$ numărul de steme în $\omega$. Asa s-a si definit spatiul evenimentelor elementare în lectia anterioară, dar acum redefinirea are o bază rațională, o justificare: este o consecintă inevitabilă a independentei aruncărilor monedei.
Acum, câteva exemple.

1. **Aruncări de monedă**. Aruncarea de trei ori a unei monede corecte. Fie $A$ evenimentul care constă în aparitia de trei ori a stemei. $A = A_1 \cap A_2 \cap A_3$, cu $A_i$ rezultatul "stemă" pentru fiecare din aruncări. Avem

$$\Pr[A] = \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2] = \Pr[A_1] \times \Pr[A_2] \times \Pr[A_3] = (1/2)^3 = 1/8$$

Versiunea a doua a expresiei decurge din faptul că aruncările sunt mutual independente. Desigur, se stie deja din lectia anterioară că $\Pr[A] = 1/8$. Aici este o confirmare a faptului că acel spatiu se comportă cum ne asteptam.

Dacă moneda este incorectă, cu probabilitatea stemei $p \neq 1/2$, utilizând independenta se obtine din nou

$$\Pr[A] = \Pr[A_1]\times\Pr[A_2]\times\Pr[A_3] = p^3$$

si, în general, probabilitatea oricărei secvente de $n$ aruncări care contine $r$ steme si $n - r$ reversuri este $p^r(1-p)^{n-r}$. Aceasta este ratiunea pentru care am definit astfel acest spatiu în lectia anterioară: am definit probabilitatea punctului ca si când aruncările monedei se petrec independent.

2. **Bile si cutii**. Spatiul evenimentelor elementare este aici produsul $\Omega = \Omega_1\times\Omega_2\times\ldots\times\Omega_m$ cu $\Omega_i = \{1, 2, \ldots, n\}$ multimea celor $n$ cutii care pot fi "alese" de bila $i$ (sunt $m$ bile si $n$ cutii). Deoarece s-a precizat că bilele sunt aruncate independent, rationând ca mai devreme, trebuie să avem $\Pr[\omega] = \Pr[\omega_1]\times\Pr[\omega_2]\times\ldots\times\Pr[\omega_m] = (1/n)^m$ pentru oricare punct $\omega$ al spatiului. Din nou apare concordanta cu definitia spatiului probabilistic din lectia anterioară.

   Regula produsului dă totodată un nouă cale de calcul al probabilităților unor evenimente. Fie $A$ evenimentul "cutia 1 este goală". Se poate scrie $A = \bigcap_{i=1}^{m} A_i$ cu $A_i$ evenimentul "bila $i$ ratează cutia 1". În mod clar, $\Pr[A_i] = 1 - 1/n$ pentru orice $i$. De asemenea, evenimentele $A_i$ sunt mutual independente prin însăsi constructia spatiului. Astfel, prin regula produsului

$$\Pr[A] = \Pr[A_1]\times\Pr[A_2 \mid A_1]\times\ldots\times\Pr[A_n \mid \bigcap_{i=1}^{n-1} A_i] =$$

$$= \Pr[A_1]\times\Pr[A_2]\times\ldots\times\Pr[A_n] = \left(1 - \frac{1}{n}\right)^m$$

   Asta se potriveste cu răspunsul obtinut în lectia anterioară prin numărarea punctelor-evenimente elementare.

3. **Cărti amestecate**. Fiecare punct $\omega$ din spatiul probelor poate fi văzut ca o secventă de 52 de alegeri, $\omega = (\omega_1, \omega_2, \ldots, \omega_{52})$, unde $\omega_i$ este cartea $i$ din pachet. Mai întâi se ia cartea $\omega_1$ din 52, apoi cartea $\omega_2$ din 51, apoi $\omega_3$ din 50 si tot asa (nu putem pune aici $\Omega = \Omega_1\times\Omega_2\times\ldots\times\Omega_{52}$ deoarece spatiul $\Omega_i$ depinde de $(\omega_1, \omega_2, \ldots, \omega_{i-1})$, de pildă $\Omega_2$ depinde de $\omega_1$). Probabilitatea de a alege un $\omega_1$ este clar 1/52; cu $\omega_1$ stabilit, probabilitatea de a alege un $\omega_2$ este uniformă, 1/51; cu $\omega_1$ si $\omega_2$ alese, probabilitatea la alegerea lui $\omega_3$ este 1/50 s.a.m.d. Aceste probabilităti conditionate cu regula produsului conduc la

$$\Pr[\omega] = \Pr[\omega_1]\times\Pr[\omega_2|\omega_1]\times\ldots\times\Pr[\omega_{52}|\omega_1, \omega_2, \ldots, \omega_{51}] = 1/52!$$

   adică la produsul fractiilor 1/52, 1/51, …, 1/2, 1/1. Se verifică ceea ce s-a stabilit în lectia anterioară prin numărarea permutărilor.

   Fie $A$ evenimentul "cartea de deasupra este un as". Vedem imediat că la prima alegere a lui $\omega_1$, $\Pr[A] = 4/52 = 1/13$. Dacă $B$ este evenimentul "primele două cărti sunt de aceeasi culoare" atunci putem considera pe $B$ ca

evenimentul care produce la a doua alegere aceeasi culoare ca la prima alegere. Aceasta se calculează astfel:

$$\Pr[\text{culoare}(\omega_1) = \text{culoare}(\omega_2)] =$$

$$= \sum_{\omega_1} \Pr[\omega_1] \times \Pr[\text{culoare}(\omega_1) = \text{culoare}(\omega_2) \mid \omega_1] = \sum_{\omega_1} \Pr[\omega_1] \times \frac{12}{51} = \frac{12}{51}$$

4. **Mâini la poker**. Din nou se foloseste o vedere a spatiului probelor ca o secventă de alegeri. Să alegem mai întâi una din cărti (a se nota că nu este "prima" carte, deoarece cărtile din mână nu au o ordine) uniform din cele 52. Apoi să alegem alta din cele 51 rămase s.a.m.d. Pentru orice mână de poker, probabilitatea de a o alege este, conform regulii produsului

$$\frac{5}{52} \times \frac{4}{51} \times \frac{3}{50} \times \frac{2}{49} \times \frac{1}{48} = \frac{1}{C_{52}^5}$$

ca si altădată. De unde vin numărătorii fractiilor? Din faptul că prima carte aleasă poate fi oricare din cele cinci, cinci alegeri din 52. A doua poate fi oricare din cele patru rămase, patru din 51. La fel în continuare. Ordinea cărtilor în mână nu are relevanță.

Să calculăm probabilitatea unei formatii numită "culoare" într-un mod diferit. Clar, aceasta este $4 \times \Pr[A]$, în care $A$ este, de pildă, evenimentul "culoare de pică". Se poate scrie $A = \bigcap_{i=1}^{5} A_i$ cu $A_i$ evenimentul "cartea $i$ este o pică". Avem astfel

$$\Pr[A] = \Pr[A_1] \times \Pr[A_2 \mid A_1] \times \ldots \times \Pr[A_5 \mid \bigcap_{i=1}^{4} A_i]$$

$\Pr[A_1] = 13/52 = 1/4$. Dar $\Pr[A_2|A_1]$? Deoarece avem o conditionare de $A_1$ (prima carte este o pică), sunt numai 51 de posibilităti rămase si 12 dintre ele sunt pici. Asadar, $\Pr[A_2|A_1] = 12/51$. Similar $\Pr[A_3|A_1 \cap A_2] = 11/50$ etc. Se obtine

$$4 \times \Pr[A] = 4 \times \frac{13}{52} \times \frac{12}{51} \times \frac{11}{50} \times \frac{10}{49} \times \frac{9}{48}$$

ceea ce coincide cu rezultatul calculat în lectia anterioară.

5. **Monty Hall**. Revenim cu definirea probabilităti unui punct din spatiul probelor ca un produs de probabilităti pentru o secventă de alegeri. Pentru problema Monty Hall

$$\Pr[1, 1, 2] = \frac{1}{3} \times \frac{1}{3} \times \frac{1}{2} = \frac{1}{18}$$

Ratiunea pentru care s-a dat această definitie este că se cunosc (din modelul problemei) probabilitătile conditionate de evenimentele anterioare. Astfel, fractia 1/2 din produsul de mai sus este probabilitatea ca Monty să deschidă usa 2 conditionată de "premiul se află dincolo de usa 1" si de "alegerea de către competitor a usii 1". Din nou, se folosesc probabiltăti conditionate pentru a defini probabilitătile din spațiul probelor specific.

Avem acum două metode de a calcula probabilitătile în multe spatii de evenimente. Este util a tine la îndemână ambele metode, atât ca o verificare a

răspunsurilor noastre cât si pentru că în unele ocazii una din metode este mai usor de utilizat decât cealaltă.

**Reuniuni de evenimente**

Sunteti în Las Vegas si observati un joc nou care are regulile date imediat.
Se alege un număr între 1 si 6. Apoi se aruncă trei zaruri. Se câstigă dacă si numai dacă numărul ales apare pe cel putin unul din zaruri.
Cazinoul pretinde că sansele de câtig sunt de 50% si utilizează următorul argument. Fie $A$ evenimentul "câstig". Se poate scrie $A = A_1 \cup A_2 \cup A_3$ cu $A_i$ evenimentul "pe zarul $i$ apare numărul ales". Este clar că probabilitatea lui $A_i$ este 1/6. Asadar

$$\Pr[A] = \Pr[A_1 \cup A_2 \cup A_3] = \Pr[A_1] + \Pr[A_2] + \Pr[A_3] = 3x(1/6) = 1/2$$

Este acest calcul corect? Ei bine, să admitem că se aruncă sase zaruri si, din nou, jucătorul câstigă dacă numărul ales apare pe măcar unul din zaruri. Un rationament analog, duce la probabilitatea 6x(1/6) = 1, adică la certitudine. Dacă sunt aruncate mai mult de sase zaruri se ating probabilităti supraunitare!
Problema este că evenimentele $A_i$ *nu sunt disjuncte*: sunt unele puncte care se situează în mai mult de unul din evenimentele $A_i$ (cu mai mult noroc se poate obtine numărul ales pe două sau chiar pe toate trei zarurile). Astfel, dacă se adună probabilitătile $\Pr[A_i]$ se numără unele puncte din saptiul probelor mai mult de o dată.
Din fericire există o formulă pentru asta cunoscută ca *principiul includerii/ excluderii*:
**Teorema 14.4** (**a includerii/excluderii**): Pentru evenimentele $A_1$, …, $A_n$ din unele spatii probabilistice, avem

$$\Pr[\bigcup_{i=1}^{n} A_i] = \sum_{i=1}^{n} \Pr[A_i] - \sum_{\{i,j\}} \Pr[A_i \cap A_j] + \sum_{\{i,j,k\}} \Pr[A_i \cap A_j \cap A_k] - \ldots \pm \Pr[\bigcap_{i=1}^{n} A_i]$$

cu sumele pe multimi $\{i, j\}$, $\{i, j, k\}$ etc. din elemente distincte, fără vreo ordine precizată.

Conform formulei, pentru a calcula $\Pr[\bigcup_{i=1}^{n} A_i]$, se însumează probabilitătile $\Pr[A_i]$, apoi se scad probabilitătile intersectiilor pe perechi, apoi se adună probabilitătile intersectiilor triple s.a.m.d.
Se poate verifica formula pentru $n = 3$, prin figurarea unei diagrame Venn si prin verificarea pe această diagramă a fiecărui punct prin numărare.
Formula generală se poate demonstra prin inductie, într-o manieră similară celei de la teorema 14.3.
Luând formula de bună, care este probabilitatea de câstig în jocul de la Las Vegas?

$$\Pr[A_1 \cup A_2 \cup A_3] = \Pr[A_1] + \Pr[A_2] + \Pr[A_3]$$
$$- \Pr[A_1 \cap A_2] - \Pr[A_1 \cap A_3] - \Pr[A_2 \cap A_3] + \Pr[A_1 \cap A_2 \cap A_3]$$

Partea frumoasă este că evenimentele $A_i$ sunt mutual independente (rezultatul citit pe un zar nu depinde de ceea ce apare pe celelalte) astfel încât $\Pr[A_i \cap A_j] =$

$\Pr[A_i]\ \Pr[A_j] = (1/6)^2 = 1/36$ si, similar, $\Pr[A_1 \cap A_2 \cap A_3] = \Pr[A_1]\Pr[A_2]\Pr[A_3] = (1/6)^3 = 1/216$. Asadar,

$$\Pr[A_1 \cup A_2 \cup A_3] = 3(1/6) - 3(1/36) + 1/216 = 91/216 \approx 0{,}42$$

Iată deci că sansa clamată de cazinou este cam prea mare fată de sansa reală!

Când $n$ este mare (reuniuni de încă mai multe evenimente), formula includerii/excluderii este în esentă inutil(izabil)ă deoarece implică evaluarea probabilitătilor pentru toate intersectiile de evenimente continute în fiecare submultime nevidă de evenimente si sunt $2^n - 1$ astfel de submultimi-evaluări. Uneori este suficientă o privire asupra primilor câtiva termeni cu ignorarea celorlalti: termenii succesivi dau alternativ o supraestimare, apoi o subestimare a răspunsului si aceste estimări merg din ce în ce mai bine, sunt din ce în ce mai precise pe măsură ce mergem mai departe.

Totusi, în multe situatii o solutie bună este dată chiar de primii termeni:

1. **Evenimente disjuncte**. Dacă evenimentele $A_i$ sunt toate *disjuncte* (nici o pereche nu are în comun puncte de probă – denumite si evenimente *mutual exclusive*), atunci

   $$\Pr[\bigcup_{i=1}^{n} A_i] = \sum_{i=1}^{n} \Pr[A_i]$$

   Acest fapt a fost utilizat deja în exemplele de mai devreme, de pildă când am spus că probabilitatea unei culori este de patru ori cea a culorii de pică – culorile diferite fiind evident evenimente disjuncte.
2. **Limita unei reuniuni**. Totdeauna

   $$\Pr[\bigcup_{i=1}^{n} A_i] \le \sum_{i=1}^{n} \Pr[A_i]$$

   Prin însumarea probabilitătilor $\Pr[A_i]$ se obtine o valoare care nu poate decât să supraestimeze probabilitatea evenimentului reuniune. Grosieră cum pare, această relatie este utilizată cu eficacitate în stiinta calculatoarelor.

# Lecția 15

## Două aplicatii killer

Iată mai departe două aplicatii "killer" ale probabilităților elementare în stiinta calculatoarelor.

1. Se presupune că o functie *hash* distribuie chei în mod egal pe o tabelă de dimensiunea $n$. Câte chei alese la întâmplare se pot distribui *hash* înainte ca probabilitatea unei coliziuni să depăsească (de pildă) 0,5?
2. Se consideră următorul scenariu simplu de a echilibra încărcarea. Sunt date $m$ joburi si $n$ masini; se alocă fiecare job unei masini în mod uniform la întâmplare si independent de toate celelalte joburi. Cât este de probabilă o valoare pentru încărcarea maximă a oricărei masini?

Cum se va vedea, ambele probleme pot fi tratate printr-o analiză a spatiului probabilistic bile-si-cutii despre care s-a mai discutat.

## Aplicatia 1: functii *hash*

Conform unor cunostinte anterioare, un tabel *hash* este o structură de date care suportă depozitarea unor seturi de chei dintr-un univers (cuprinzător) $U$ (de pildă numele celor 22 de milioane de locuitori ai României). Operatiile suportate sunt ADD (adăugarea) unei chei la un set, DELETE (stergerea) unei chei dintr-un set si verificarea MEMBER a (apartenentei) unei chei la un set. Functia *hash* $h$ aplică $U$ pe un tabel $T$ de dimensiune moderată. Pentru a adăuga (ADD) o cheie la setul în discutie, se evaluează $h(x)$ (adică se aplică functia hash pe cheie) si se depune $x$ în locatia $h(x)$ din tabelul $T$. Toate cheile din setul nostru care sunt aplicate pe aceeasi locatie din tabelul $T$ sunt stocate într-o listă simplă *link*-ată. Operatiile DELETE si MEMBER sunt implementate într-o manieră similară, prin evaluarea lui $h(x)$ si căutând în lista *link*-ată la $h(x)$. De observat că eficienta functiei *hash* constă în a avea cât mai putine coliziuni, altfel spus chei care se aplică pe aceeasi locatie. Aceasta pentru că timpul de căutare pentru operatiile DELETE si MEMBER este proportional cu lungimea listei *link*-ate corespunzătoare.

Întrebarea care interesează aici este următoarea: presupunând că tabelul *hash* $T$ are dimensiunea $n$ si că functia *hash* $h$ distribuie $U$ egal peste $T$ (adică $|U| = \alpha n$ – dimensiunea lui $U$ este un multiplu întreg de factor $\alpha$ al dimensiunii lui $T$ – si pentru fiecare $y$ din $T$ numărul de chei $x$ din $U$ pentru care $h(x) = y$ este exact $\alpha$), presupunând că acele chei de stocat sunt alese uniform la întâmplare si

independent din universul $U$, care este cel mai mare număr $m$ de chei care se pot stoca înainte ca probabilitatea unei coliziuni să atingă 0,5?

Să începem prin a vedea cum se poate pune această problemă în tiparul cutii-si-bile. Bilele vor fi cele $m$ chei de stocat si cutiile vor fi cele $n$ locatii din tabelul *hash* $T$. Deoarece cheile sunt alese uniform si independent din $U$ si deoarece functia *hash* distribuie cheile egal peste tabel, se poate vedea fiecare cheie (bilă) ca alegerea unei locatii (cutii) uniform si independent în tabelul $T$. Astfel, spatiul probabilistic care corespunde acestui experiment de *hashing* este exact acelasi cu spatiul bilelor si cutiilor.

Interesează acel eveniment $A$ care nu are nici o coliziune, sau echivalent, toate cele $m$ bile aterizează în cutii diferite. Clar, Pr[$A$] va descreste cu cresterea lui $m$ (cu $n$ fixat). Telul nostru este acela de a găsi cea mai mare valoare a lui $m$ astfel încât Pr[$A$] să rămână deasupra lui 0,5. [Notă: în realitate aici se lucrează cu spatii ale probelor (evenimentelor) diferite, unul pentru fiecare valoare a lui $m$. Astfel ar fi mult mai corect a scrie $\Pr_m$ si nu simplu Pr, pentru a spune clar despre ce spatiu al probelor este vorba. Totusi, acest detaliu va fi omis si în continuare.]

Să fixăm pe $m$ si să încercăm evaluarea lui Pr[$A$]. Deoarece spatiul probabilităților este uniform (fiecare rezultat are probabilitatea $1/n^m$), este suficient numai a inventaria numărul de rezultate din $A$. În câte moduri se pot aranja $m$ bile în $n$ cutii astfel încât nici o cutie să nu conțină mai mult de o bilă? Ei bine, sunt $n$ locuri în care poate fi pusă prima bilă, apoi $n - 1$ locuri rămase pentru a doua bilă (deoarece ea nu poate fi, nu trebuie să fie pusă în aceeasi cutie cu prima), $n - 2$ locuri pentru bila a treia s.a.m.d. Astfel, numărul total de astfel de aranjamente este

$$n \times (n-1) \times (n-2) \times \ldots \times (n-m+2) \times (n-m+1)$$

formulă validă atât timp cât $m \le n$. Dacă $m > n$ atunci clar, răspunsul este diferit, este zero. De aici înainte se consideră $m \le n$.

Acum se poate calcula probabilitatea lipsei de coliziune:

$$\Pr[A] = \frac{n(n-1)(n-2)\cdots(n-m+1)}{n^m} = \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{m-1}{n}\right) \qquad (1)$$

Înainte de a merge mai departe, o pauză scurtă pentru a observa că probabilitatea Pr[$A$] poate fi calculată si în alt mod. Spatiul probabilităților este privit acum ca o secventă de alegeri, una pentru fiecare bilă. Pentru $1 \le i \le m$, fie $A_i$ evenimentul constând în faptul că bila numărul $i$ aterizează într-o cutie diferită de bilele 1, 2, …, $i - 1$. Atunci

$$\Pr[A] = \Pr\left[\bigcap_{i=1}^{m} A_i\right] =$$

$$= \Pr[A_1] \times \Pr[A_2 \mid A_1] \times \Pr[A_3 \mid A_1 \cap A_2] \times \cdots \times \Pr\left[A \mid \bigcap_{i=1}^{m-1} A_i\right] =$$

$$= 1 \times \frac{n-1}{n} \times \frac{n-2}{n} \times \cdots \times \frac{n-m+1}{n} = \left(1-\frac{1}{n}\right) \times \left(1-\frac{2}{n}\right) \times \cdots \times \left(1-\frac{m-1}{n}\right)$$

Rezultatul este, din fericire, acelasi! [Asigurati-vă că ati înteles cum s-au obtinut probabilitătile conditionate din expresia de mai sus. De exemplu, Pr[$A_3$|

$A_1 \cap A_2$] este probabilitatea ca a treia bilă să aterizeze într-o cutie diferită de cele ocupate de primele două bile, fiind precizat faptul că acele două bile au aterizat de asemenea în cutii diferite. Aceasta înseamnă că a treia bilă are $n - 2$ alegeri posibile din totalul de $n$.]

În esentă, problema este încheiată: ecuatia (1) de mai sus dă formula exactă pentru probabilitatea lipsei de coliziune când $m$ chei sunt *hashed*. Tot ce mai trebuie făcut acum este să se introducă în acea ecuatie valori $m = 1, 2, 3, \ldots$ până când probabilitatea Pr[$A$] scade sub 0,5. Valoarea corespunzătoare a lui $m$ (minus 1) este cea care se urmăreste.

Dar aceasta nu este o cale realmente satisfăcătoare: ar fi mult mai util a avea o formulă care să dea valoarea "critică" a lui $m$ direct si nu prin calcularea probabilitătii Pr[$A$] pentru $m = 1, 2, 3, \ldots$ De observat că această evaluare trebuie repetată separat pentru fiecare valoare diferită a lui $n$, adică ori de câte ori se schimbă dimensiunea tabelului *hash*.

Asa că ce rămâne de făcut este a rostogoli pe toate părtile ecuatia (1) astfel ca ea să spună valoarea lui $m$ pentru care Pr[$A$] scade sub 0,5. Pentru asta se logaritmează: este un lucru bun de făcut deoarece transformă produsul în sumă, ceea ce este mai usor de manevrat. Se obtine:

$$\ln\{\Pr[A]\} = \ln\left(1 - \frac{1}{n}\right) + \ln\left(1 - \frac{2}{n}\right) + \cdots + \ln\left(1 - \frac{m-1}{n}\right) \qquad (2)$$

în care logaritmii sunt cei naturali. Acum se poate face uz de aproximarea standard pentru logaritmi si anume, dacă $x$ este mic, atunci $\ln(1 - x) \approx -x$. Această aproximare vine din dezvoltarea Taylor

$$\ln(1 - x) = -x - (x^2/2) - (x^3/3) - \ldots$$

Prin înlocuirea lui $\ln(1 - x)$ prin $-x$ se comite o eroare de cel mult $[(x^2/2) + (x^3/3) + \ldots]$ care este de cel mult $2x^2$ dacă $x \leq 0{,}5$. Cu alte cuvinte

$$-x - 2x^2 \leq \ln(1 - x) \leq -x$$

Si dacă $x$ este mic atunci termenul de eroare $2x^2$ este mult mai mic decât termenul principal $-x$. În loc de a purta în calcule termenul $2x^2$, în continuare se va folosi aproximarea $\ln(1 - x) \approx -x$, cu constiinta faptului că aproximarea poate fi făcută mai precisă dacă este cazul.

Acum, dacă se introduce aproximarea preconizată în relatia (2) se obtine

$$\ln\{\Pr[A]\} \approx -\frac{1}{n} - \frac{2}{n} - \cdots - \frac{m-1}{n} = -\frac{1}{n}\sum_{i=1}^{m-1} i = -\frac{m(m-1)}{2n} \approx -\frac{m^2}{2n} \qquad (3)$$

De notat că s-a utilizat aproximarea pentru $\ln(1 - x)$ pentru $x = 1/n, 2/n, 3/n, \ldots, (m - 1)/n$. Astfel, aproximarea trebuie să fie bună dat fiind că toate aceste numere sunt mici, adică dat fiind că $n$ este destul de mare si $m$ este întrucâtva mai mic decât $n$. Odată terminat calculul se poate vedea că aproximarea este destul de bună chiar pentru $n$ moderat ca mărime.

Acum se poate inversa logaritmul din expresia (3) pentru a obtine expresia probabilitătii Pr[$A$]:

$$\Pr[A] \approx e^{-\frac{m^2}{2n}}$$

Pasul ultim este destinat evaluării acelei valori a lui $m$ care face această probabilitate să treacă sub 0,5. Cel mai mare $m$ pentru care $e^{-\frac{m^2}{2n}} \geq \frac{1}{2}$ se obtine din

$$-\frac{m^2}{2n} \geq \ln\left(\frac{1}{2}\right) = -\ln 2 \qquad (4)$$

ceea ce este echivalent cu

$$m \leq \sqrt{(2\ln 2)n} \approx 1{,}177\sqrt{n}$$

Astfel linia de jos este aceea că se pot repartiza *hash* aproximativ $m = \lfloor 1{,}177\sqrt{n} \rfloor$ chei înainte de a obtine o probabilitate de 0,5 pentru unicitatea asocierii cheie-pozitie în tabel.

De retinut că acest calcul a fost aproximativ; este acum posibil un calcul exact pentru a face vizibilă eroarea. Se utilizează, desigur, ecuatia (1). Se calculează pentru câteva valori ale lui $n$ valoarea lui $m = m_0$ corectă, valoare care face ca probabilitatea Pr[$A$] să treacă de 0,5. Tabelul alăturat prezintă valorile exacte si valorile estimate.

| $n$ | 10 | 20 | 50 | 100 | 200 | 365 | 500 | $10^3$ | $10^4$ | $10^5$ | $10^6$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\frac{1{,}177}{\sqrt{n}}$ | 3,7 | 5,3 | 8,3 | 11,8 | 16,6 | 22,5 | 26,3 | 37,3 | 118 | 372 | 1177 |
| $m_0$ exact | 4 | 5 | 8 | 12 | 16 | 22 | 26 | 37,3 | 118 | 372 | 1177 |

Din tabel se vede că aproximarea este foarte bună chiar pentru valori ale lui $n$ mici. Când $n$ este mare diferentele devin neglijabile.

**De ce 0,5?**

Problema de *hashing* enuntată se referă la o probabilitate de coliziune de 0,5. Este oare ceva special cu acest 0,5? Răspunsul: nu! S-a încercat un calcul (aproximativ) al Pr[$A$] (probabilitatea lipsei coliziunilor) ca o functie de $m$ si apoi s-a stabilit cea mai mare valoare a lui $m$ pentru care estimarea este sub 0,5. Dacă interesa o probabilitate a coliziunii de (să spunem) 0,95 (sau 95%) atunci se înlocuia 0,5 cu 0,05 în ecuatia (4). Cu putină prelucrare algebrică se ajunge la valoarea critică $m = \sqrt{(2\ln 20)n} \approx 2{,}45\sqrt{n}$. Asadar, indiferent ce probabilitate de încredere este specificată, valoarea critică va fi totdeauna $m = c\sqrt{n}$, cu $c$ o constantă (dependentă de nivelul de încredere probabilistic impus).

**Zile de nastere**

Iată o problemă faimoasă, denumită adesea "paradoxul zilelor de nastere" desi nu este vorba de nici un paradox. Aveti prieteni si vreti să-i invitati la o

petrecere. Câti trebuie să invitati pentru a avea o sansă bună (să spunem de cel putin 50%) ca doi dintre ei să aibă aceeasi zi de nastere? Este exact problema *hashing*-coliziune, cu $n = 365$ (numărul zilelor unui an – de fapt mai sunt si ani bisecti si în plus mai si presupunem că zilele de nastere sunt independente si uniform distribuite pe durata anului ceea ce nu este foarte exact; numărul de nasteri fluctuează pe durata anului cu unele vârfuri, de pildă la nouă luni de la sărbătorile de iarnă). Din tabelul de mai devreme, se poate vedea că 23 de invitati sunt suficienti (de ce?). Dacă doriti să mergeti mai la sigur si să aveti o probabilitate mai înaltă, de pildă 95%, trebuie să invitati 47 de persoane (a se verifica acest număr).

**Aplicatia 2: Echilibrarea încărcării**

Una din cele mai presante probleme de ordin practic în calculul paralel este aceea a distribuirii încărcării între procesoare. Este o problemă uriasă, încă nerezolvată în cazul general. Aici este examinat un scenariu extrem de simplu care este fundamental si stabileste o linie de bază fată de care pot fi judecate metodele mai sofisticate.

Să presupunem că avem $m$ joburi identice si $n$ procesoare identice. Sarcina noastră este a atribui joburile pe procesoare astfel încât nici un procesor să nu fie foarte încărcat. Desigur, există aici o solutie optimă: se împart joburile atât de egal pe cât se poate, astfel încât fiecare procesor să primească fie $\lceil m/n \rceil$, fie $\lfloor m/n \rfloor$ joburi. Cu toate acestea, solutia aceasta reclamă foarte mult control centralizat si/sau foarte multă comunicare: încărcarea trebuie echilibrată fie printr-un planificator central foarte puternic care "vorbeste" tuturor procesoarelor, fie prin schimb intens de mesaje între joburi si procesoare. Acest tip de operatii este foarte costisitor în cele mai multe sisteme de calcul distribuit. Asadar apare întrebarea: ce se poate face cu putin overhead (sau deloc) în costul planificării si al comunicatiei?

Prima ideie la îndemână este… mingi si cutii! Adică, fiecare job selectează simplu un procesor uniform aleator si independent de toate celelalte si merge la acel procesor. (Trebuie să ne asigurăm că spatiul probabilistic al acestui experiment este acelasi cu cel al problemei mingilor si cutiilor.) Această schemă nu cere vreo comunicare. Cu toate acestea, e de prezumat că metoda nu va atinge în general o echilibrarea optimă. Fie $X$ încărcarea maximă a unui procesor în schema aleatoare prezentată. Se observă usor că $X$ nu este un număr fix: valoarea sa depinde de rezultatul experimentului mingi si cutii (este de fapt o variabilă aleatoare, lucru care-l vom defini în sectiunea următoare). Astfel, ca proiectanti sau utilizatori ai acestei scheme de echilibrare, care ar fi problema noastră?

Întrebare: Care este acea valoare $k$ pentru care $\Pr[X \geq k] \leq 0{,}5$.

Dacă valoarea $k$ este judicios potrivită, atunci vom avea o probabilitate bună (de cel putin 1/2) ca încărcarea maximă pe oricare dintre procesoare să nu depăsească acel $k$. Asta dă o idee corectă asupra performantei sistemului.

Desigur, ca si în aplicatia *hashing* de mai devreme, valoarea 1/2 nu are nimic special: s-a utilizat numai pentru ilustrare. Cum se poate verifica în esentă, aceeasi analiză poate fi utilizată pentru a stabili valori $k$ pentru niveluri de încredere diferite, de exemplu astfel încât $\Pr[X \geq k] \leq 0{,}05$ (adică la un nivel de încredere de 95%). Desigur, se pot găsi valori $k$ si pentru alte niveluri de încredere si prin aceasta se poate construi o imagine mai detaliată a comportării schemei propuse. Pentru simplificare, se poate presupune de aici încolo si că $m = n$ (adică numărul de joburi este egal cu cel al procesoarelor). Cu un surplus de muncă, analiza se poate generaliza la alte valori ale lui $m$.

Din aplicatia 1 se stie că o coliziune are loc cu o probabilitate mai mare de 0,5 dacă $m \approx 1{,}177\sqrt{n}$. Asadar, dacă $m = n$ sarcina maximă va fi (aproape) cu certitudine mai mare ca 1. Deoarece încărcarea uneia dintre cutii depinde de încărcările celorlalte, chestiunea devine mult mai plină de neprevăzut. Trebuie o analiză a încărcării unei cutii, altfel oarecare, să spunem cutia 1: asta-i destul de usor. Notăm cu $X_1$ încărcarea cutiei 1 (si aceasta o variabilă aleatoare). Ce este de făcut este a stabili un $k$ astfel încât

$$\Pr[X_1 \geq k] \leq \frac{1}{2n} \qquad (5)$$

Deoarece toate cutiile sunt identice, se cunoaste atunci că, pentru acelasi $k$,

$$\Pr[X_i \geq k] \leq \frac{1}{2n} \qquad \text{pentru } i = 1, 2, \ldots, n,$$

în care $X_i$ este încărcarea cutiei $i$. Dar acum, deoarece evenimentul $X \geq k$ este exact reuniunea evenimentelor $X_i \geq k$ (de ce?), se poate utiliza "*Union Bound*" din lectia precedentă:

$$\Pr[X \geq k] = \Pr\left[\bigcup_{i=1}^{n} (X_i \geq k)\right] \leq \sum_{i=1}^{n} \Pr[X_i \geq k] = n\frac{1}{2n} = 0{,}5$$

Este important a zăbovi asupra a ceea ce am făcut aici: am dorit ca $\Pr[A] \leq 1/2$ cu $A$ evenimentul $X \geq k$. Nu s-a putut analiza $A$ direct, dar se stie că $A = \bigcup_{i=1}^{n} A_i$ pentru evenimentele mult mai simple $A_i$ (si anume, $A_i$ este evenimentul $X_i \geq k$). Deoarece sunt $n$ evenimente $A_i$ si toate au aceleasi probabilităti, este suficient a arăta că $\Pr[A_i] \leq 1/2n$; limitarea pentru reuniuni (*union bound*) ne garantează atunci că $\Pr[A] \leq 1/2$. Această manieră de a rationa este foarte obisnuită în aplicatii ale probabilitătile în stiinta calculatoarelor.

Acum, înapoi la problema echilibrării. Misiunea a fost redusă la a afla un $k$ astfel încât

$$\Pr[X_1 \geq k] \leq \frac{1}{2n}$$

cu $X_1$ încărcarea cutiei numărul 1. Nu este dificil a scrie o expresie exactă pentru $\Pr[X_1 = f]$, probabilitatea ca încărcarea cutiei 1 să fie exact $f$:

$$\Pr[X_1 = j] = C_n^j \left(\frac{1}{n}\right)^j \left(1 - \frac{1}{n}\right)^{n-j} \qquad (6)$$

Aceasta poate fi interpretată astfel: fiecare bilă este o aruncare de monedă măsluită: "stema" ar corespunde bilei care aterizează în cutia 1, "reversul" fiind asociat cu celelalte rezultate. Probabilitatea "stemei" este $1/n$ si aruncările monedei sunt mutual independente. Cum s-a văzut în lectiile anterioare, (6) dă probabilitatea ca exact $j$ rezultate "stemă" să apară în $n$ aruncări.
Astfel se obtine

$$\Pr[X_1 \geq k] = \sum_{j=k}^{n} \Pr[X_1 = j] = \sum_{j=k}^{n} C_n^j \left(\frac{1}{n}\right)^j \left(1 - \frac{1}{n}\right)^{n-j} \qquad (7)$$

Acum, într-un anume sens am isprăvit: trebuie numai să introducem valori $k = 1, 2, \ldots$ în (7) până când probabilitatea scade sub $1/2n$. Cu toate acestea, ca în exemplul *hashing*, ar fi mult mai util a face relatia (7) mai imediată, capabilă a oferi valori $k$ mai direct. Pentru a face aceasta se înlocuieste ecuatia exactă (7) cu o aproximare obtinută din limitarea reuniunii: se va vedea că această aproximare este destul de bună în practică.
Fie $B$ evenimentul "$X_1 \geq k$" si pentru fiecare submultime $S \subseteq \{1, 2, \ldots, n\}$ de exact $k$ bile, fie $B_S$ evenimentul care constă în căderea tuturor bilelor în cutia 1. Clar, $B$ este reuniunea evenimentelor $B_S$ deoarece $B$ se produce dacă si numai dacă cel putin unul din evenimentele $B_S$ se produce. Astfel, utilizând din nou limitarea reuniunii se poate scrie

$$\Pr[X_1 \geq k] \equiv \Pr[B] = \Pr\left[\bigcup_S B_S\right] \leq \sum_S \Pr[B_S] \qquad (8)$$

Ce este $\Pr[B_S]$? Ei bine, pentru orice $S$ este tocmai probabilitatea ca un set particular de $k$ bile să cadă în cutia 1, ceea ce este exact $(1/n)^k$. Si numărul de asemenea seturi este $C_n^k$. Astfel, suma din (8) poate fi scrisă

$$\Pr[X_1 \geq k] \leq \sum_S \Pr[B_S] = C_n^k \left(\frac{1}{n}\right)^k \qquad (9)$$

Cu această expresie este mult mai simplu de lucrat decât cu suma din ecuatia (7).
Singurul lucru deranjant rămas în ecuatia (9) este coeficientul binomial $C_n^k$. Cu acesta se poate lucra mai usor prin utilizarea aproximării $\left(\frac{n}{k}\right)^k \leq C_n^k \leq \left(\frac{ne}{k}\right)^k$ care dă

$$\Pr[X_1 \geq k] \leq \left(\frac{ne}{k}\right)^k \left(\frac{1}{n}\right)^k = \left(\frac{e}{k}\right)^k \qquad (10)$$

Paranteză: aproximarea de mai sus pentru $C_n^k$ face parte din bagajul de trucuri matematice pe care-l poartă matematicienii si specialistii în stiinta computerelor; nu este dificil a dovedi mărginirea inferioară; mărginirea superioară este ceva mai încurcată si face uz de altă aproximare pentru $n!$ cunoscută ca *aproximarea lui Stirling* care spune că factorialul lui $k$ este mai mare sau egal fată de $(k/e)^k$; detaliile nu se dau aici. Sfârsit de paranteză.

De notat că, chiar dacă s-a recurs la câteva aproximări, inegalitatea (10) este deplin validă: toate aproximările utilizate sunt de tipul "≤", asa încât totdeauna există o limitare superioară pentru Pr[$X_1 \geq k$]. O revenire la calculul de mai sus poate lămuri această afirmatie.

Recitirea relatiei (5) readuce în prim plan scopul urmărit: a face probabilitatea din (10) mai mică decât $1/2n$. Se poate asigura aceasta prin alegerea lui $k$ astfel încât

$$\left(\frac{e}{k}\right)^k \leq \frac{1}{2n} \qquad (11)$$

Acum se pot face afirmatii hotărîte: fiind dată o valoare a numărului $n$ (joburi/procesoare) trebuie găsită valoarea cea mai mică a lui $k = k_0$ care satisface relatia (11). Vom sti atunci că, cu probabilitatea de cel putin 1/2 sarcina maximă a oricărui procesor este de cel mult $k_0$. Tabelul care urmează arată aceste valori $k_0$ pentru câteva valori ale lui $n$. Ca exercitiu, cititorul este îndemnat a executa experimentul si a compara aceste valori $k_0$ cu ce se întâmplă în practică.

| $n$ | 10 | 20 | 50 | 100 | 500 | $10^3$ | $10^4$ | $10^5$ | $10^6$ | $10^7$ | $10^8$ | $10^{15}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_0$ exact | 5 | 6 | 6 | 7 | 8 | 8 | 9 | 10 | 11 | 12 | 13 | 19 |
| ln(2$n$) | 3 | 3,7 | 4,6 | 5,3 | 6,9 | 7,6 | 9,9 | 12,2 | 14,5 | 16,8 | 19,1 | 35,2 |
| 2ln$n$/lnln$n$ | 5,6 | 5,4 | 5,8 | 6 | 6,8 | 7,2 | 8,2 | 9,4 | 10,6 | 11,6 | 12,6 | 20 |

Se poate da o formulă pentru $k_0$ ca functie de $n$ (cum s-a făcut la problema de *hashing*)? Ei bine, luând logaritmul expresiei (11) se obtine

$$k(\ln k - 1) \geq \ln(2n) \qquad (12)$$

Din aceasta, se poate ghici o aproximare destul de bună pentru $k_0$: $k = \ln(2n)$. Introducerea aceastei valori $k$ face partea dreaptă a expresiei (12) egală cu $\ln(2n)(\ln\ln(2n) - 1)$ care este cu sigurantă mai mare decât $\ln(2n)$ deoarece $\ln(2n) \geq 2$, adică $n \geq \frac{1}{2}e^{e^2} \approx 810$. Astfel, pentru $n \geq 810$ se poate spune că sarcina maximă este (cu probabilitate de cel putin 0,5) nu mai mare de $\ln(2n)$. Tabelul de mai devreme pune alături valorile lui $\ln(2n)$ pentru comparatie cu $k_0$. Cum era de asteptat estimarea este destul de bună pentru $n$ mic dar devine pesimistă pentru $n$ mare.

Pentru valori mari ale lui $n$ se poate face o evaluare mai bună. Dacă introducem în partea stângă a relatiei (12) valoarea $k = \ln n/\ln\ln n$, aceasta devine

$$\frac{\ln n}{\ln\ln n}(\ln\ln n - \ln\ln\ln n - 1) = \ln n\left(1 - \frac{\ln\ln\ln n}{\ln\ln n}\right) \qquad (13)$$

Acum, dacă $n$ este mare, aceasta este usor mai mică decât partea dreaptă, $\ln(2n)$. De ce? Deoarece al doilea termen din paranteză tinde la zero când $n \to \infty$ (Pentru a vedea aceasta, de notat că este de forma $(\ln z + 1)/z$ cu $z = \ln\ln n$ si, desigur, $\ln z/z \to 0$ dacă $z \to \infty$) si deoarece $\ln(2n) = \ln n + \ln 2$, care este foarte apropiat de $\ln n$ când $n$ este mare ($\ln 2$ este o constantă mică). Se poate conchide astfel că pentru valori mari ale lui $n$ cantitatea $\ln n/\ln\ln n$ poate fi o estimare

destul de bună pentru $k_0$. În realitate, pentru ca această estimare să devină bună, $n$ trebuie să devină astronomic (literal) de mare. Pentru valori mai civilizate ale lui $n$, se obtin estimatii mai bune prin luarea lui $k = 2\ln n/\ln\ln n$. Factorul suplimentar 2 ajută la stergerea termenilor de ordin inferior (adică termenul secund din paranteza din (13) si logaritmul ln2) mai rapidă. Tabelul de mai sus arată comportarea acestei estimări pentru valori diferite ale lui $n$.

În final, iată o linie în plus pentru aplicatia 2. Se presupune că populatia SUA este de 250 de milioane. Se presupune că expediem 250 de milioane de mesaje *junk* de e-mail, fiecare la o adresă din SUA luată la întâmplare. Atunci (conform tabelului de mai sus), cu o probabilitate de cel putin 50%, nici o persoană nu va primi mai mult de circa o duzină de mesaje.

# Lecția 16

## Variabile aleatoare si medii

**Problemă**: Lucrările de casă elaborate de 20 de sudenti sunt colectate, amestecate si returnate studentilor. Câti studenti primesc propria lor lucrare?
Pentru a răspunde la acestă întrebare trebuie mai întâi precizat spatiul probelor: el ar trebui să conțină toate cele 20! permutări ale lucrărilor, fiecare cu probabilitatea 1/20! (se aseamănă cu spatiul atasat pachetului de cărti amestecat, cu deosebirea că acolo se amestecau 52 de obiecte). E de folos a avea o imagine a unei permutări. Imaginati-vă 20 de cărti aliniate pe un raft, numerotate de la stânga la dreapta cu 1, 2, …, 20. O permutare $\pi$ este o altă ordonare a cărtilor si, de fapt, o altă listă a numerelor lor citite de la stânga la dreapta. Să notăm $\pi_i$ eticheta numerică a cărtii din pozitia $i$. Ne interesează numărul de cărti care sunt încă în pozitia lor initială, adică numerele $i$ pentru care $\pi_i = i$. Acestea sunt numite adesea *puncte fixe* ale permutării.
Desigur, întrebarea din introducere nu are un răspuns numeric simplu (de pildă 6) deoarece numărul depinde de permutarea particulară aleasă (depinde de punctul-probă). Să notăm numărul de puncte fixe cu $X$. Pentru a face lucrurile mai simple, să scădem pentru moment dimensiunea problemei de la 20 la 3. Tabelul care urmează

| Permutarea $\pi$ | Valoarea lui $X$ |
|---|---|
| 123 | 3 |
| 132 | 1 |
| 213 | 1 |
| 231 | 0 |
| 312 | 0 |
| 321 | 1 |

dă o listă completă a spatiului probelor (3! = 6) alături de valorile corespunzătoare ale lui $X$ pentru fiecare punct. $X$ ia valorile 0, 1 sau 3 depinzând de punct. O cantitate ca aceasta, care ia o valoare numerică în fiecare punct-probă este numită *variabilă aleatoare* pe spatiul probelor.

**Definitia 16.1** (**variabilă aleatoare**): O *variabilă aleatoare X* pe spatiul Ω al probelor este o functie care atribuie fiecărui punct $\omega \in \Omega$ un număr real $X(\omega)$.
Deocamdată ne concentrăm asupra variabilelor aleatoare *discrete*, în particular acelea pentru care $|\Omega|$ este finit.
Variabila aleatoare $X$ din exemplul cu permutări este complet specificată prin tabelul de mai sus (de pildă, $X(123) = 3$ etc.)
Uneori interesează nu atât valorile în fiecare punct $\omega$ cât multimea de puncte în care variabila aleatoare ia anumite valori. Fie $a$ un număr real. Multimea $\{\omega \in \Omega: X(\omega) = a\}$ este un eveniment în spatiul evenimentelor (de ce?). Se scrie uneori acest eveniment ca "$X = a$". Dacă acesta este un eveniment atunci se poate vorbi de $\Pr[X = a]$. Aceste probabilităti, pentru toate valorile $a$ posibile, puse laolaltă alcătuiesc *distributia* variabilei aleatoare $X$.
**Definitia 16.2** (**distributie**): *Distributia* unei variabile aleatoare discrete $X$ este colectia de valori $[(a, \Pr[X = a]): a \in \mathcal{A}]$ cu $\mathcal{A}$ multimea tuturor valorilor posibile la lui $X$.
Distributia variabilei X din tabelul de mai devreme este

$$\Pr[X = 0] = 1/3 \quad \Pr[X = 1] = 1/2 \quad \Pr[X = 3] = 1/6$$

cu $\Pr[X = a] = 0$ pentru orice altă valoare $a$.
De remarcat că suma probabilităților $\Pr[X = a]$ peste toate valorile $a$ este exact unitatea. Acesta este cazul *totdeauna*, pentru orice variabilă aleatoare, deoarece o variabilă aleatoare trebuie să ia o valoare si numai una, $X(\omega)$ în fiecare punct $\omega$. $X = a$ partitionează spatiul probelor si când se însumează probabilitătile evenimentelor $X = a$, se însumează de fapt probabilitătile tuturor punctelor din saptiul probelor.

**Medii sau sperante matematice**

În multe aplicatii, distributia completă a unei variabile aleatoare este foarte greu de calculat: de exemplu, revenind la exemplul celor 20 de lucrări de casă, am numărat în principiu $20! \approx 2{,}4.10^{18}$ puncte, evenimente elementare. A calcula valoarea variabilei $X$ în fiecare din ele si a număra punctele în care $X$ ia aceeasi valoare este o sarcină cel putin descurajantă. În plus, calculul distributiei complete a unei variabile aleatoare nu este totdeauna foarte pretioasă informativ.
Din aceste motive, se încearcă o comprimare a distributiei în forme mai compacte, mai convenabile, care mai sunt si usor de evaluat. Dintre acestea cea mai larg utilizată este media (sau speranta matematică).
**Definitia 16.3** (**media**): Media unei variabile aleatoare discrete $X$ este definită ca

$$E(X) = \sum_a a \Pr[X = a]$$

cu suma luată pentru toate valorile $a$ pe care $X$ le poate lua.
Pentru cazul permutărilor de 3 obiecte enuntat mai sus

$$E(X) = 0x(1/3) + 1x(1/2) + 3x(1/5) = 1$$

adică numărul asteptat (media) de puncte fixe într-o permutare este 1.
Media poate fi privită într-un anume sens ca o valoare tipică a variabilei aleatoare (desi în realitate variabila poate să nu ia niciodată valoarea mediei). Problema cât de tipică este media pentru o variabilă aleatoare dată este importantă si se va reveni asupra ei.
Iată deocamdată câteva exemple simple de calcul al mediei.

1. **Zarul**. Se aruncă zarul presupus corect. Fie $X$ numărul de puncte afisat pe fata de deasupra: $X$ ia valorile 1, 2, …, 6, fiecare cu probabilitatea 1/6 asa încât
$$E(X) = (1/6)(1 + 2 + 3 + 4 + 5 + 6) = 21/6 = 3{,}5$$
De observat că $X$ nu ia niciodată valoarea 3,5.
2. **Două zaruri**. Fie acum două zaruri corecte. Fie $X$ suma celor două numere afisate. Distributia lui $X$ este

| $a$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\Pr[X = a]$ | $\frac{1}{36}$ | $\frac{2}{36}$ | $\frac{3}{36}$ | $\frac{4}{36}$ | $\frac{5}{36}$ | $\frac{6}{36}$ | $\frac{5}{36}$ | $\frac{4}{36}$ | $\frac{3}{36}$ | $\frac{2}{36}$ | $\frac{1}{36}$ |

Media este
$$E(X) = (1/36)(1x2 + 2 \text{ x } 3 + 3x4 + 4x5 + 5x6 + 6x7 + 5x8 + 4x9 + 3x10 + 2x11 + 1x12) = 7$$
3. **Ruleta**. O ruletă este pusă în miscare. Pariati 1 leu pe negru. Dacă rezultatul este un număr negru primiti miza si 1 leu, altminteri pierdeti miza. Fie $X$ câstigul net la un tur. Câstigul poate fi + 1 leu sau – 1 leu cu probabilitătile 18/38, respectiv 20/38 (ruleta are un disc cu 38 de sectoare: numerele 1, 2, …, 36 sunt alternativ colorate rosu sau negru si mai sunt alte două sectoare 0 si 00, colorate în verde). Asadar
$$E(X) = 1x(18/38) + (–1)x(20/38) = – 1/19$$

Adică este de asteptat o pierdere de cca. 5 bani per joc. Balanta este înclinată în favoarea cazinoului…

**Liniaritatea mediei**

Până acum s-au calculat medii prin tratare mai curând brută: s-au scris distributii complete si s-au însumat contributiile tuturor valorilor posibile ale variabilei aleatoare. Valoarea reală a mediilor în exemple din realitate constă în faptul că ele se pot calcula mai usor pe o cale mai simplă. Calea aceasta este dată de

**Teorema 16.1**: Pentru orice două variabile aleatoare $X$ si $Y$ definite pe acelasi spatiu probabilistic avem
$$E(X + Y) = E(X) + E(Y)$$
Tot asa, pentru orice constantă $c$
$$E(cX) = cE(X).$$

**Demonstratie**: Pentru a facilita demonstratia, să rescriem definitia mediei într-o formă convenabilă. Din definitia 16.3

$$E(X) = \sum_a a \Pr[X = a]$$

În termenul curent $\Pr[X = a]$ este prin definitie suma probabilităților $\Pr[\omega]$ pentru toate punctele $\omega$ în care $X(\omega) = a$. Se stie că fiecare punct $\omega \in \Omega$ se asociază cu exact una din valorile $a$. Asta înseamnă că relatia de definitie de mai sus se mai poate scrie si altfel

$$E(X) = \sum_{\omega \in \Omega} X(\omega) \Pr[\omega]$$

Această definitie echivalentă a mediei face demonstratia, cum s-a spus, mai usoară desi este mai incomodă pentru calcule efective.
Se scrie

$$\begin{aligned} E(X+Y) &= \sum_{\omega \in \Omega} (X+Y)(\omega) \times \Pr[\omega] \\ &= \sum_{\omega \in \Omega} (X(\omega) + Y(\omega)) \times \Pr[\omega] \\ &= \sum_{\omega \in \Omega} X(\omega) \times \Pr[\omega] + \sum_{\omega \in \Omega} Y(\omega) \times \Pr[\omega] \\ &= E(X) + E(Y) \end{aligned}$$

si demonstratia este completă.
□
Teorema 16.1 este foarte puternică: ea spune că media sumei a două variabile aleatoare este suma mediilor lor luate separat, fără a face vreo presupunere suplimentară asupra variabilelor aleatoare implicate (de pildă nu se cere ca variabilele să fie independente). Teorema se poate utiliza pentru a scrie relatii de genul $E(3X - 5Y) = 3E(X) - 5E(Y)$. Important: Teorema *nu* spune că $E(XY) = E(X)E(Y)$ si nici că $E(1/X) = 1/\mathrm{E}(X)$ etc. Aceste relatii sunt în general false. Intră în cadrul teoremei numai sumele (diferentele) si multiplii cu factori constanti ai variabilele aleatoare.
Iată acum teorema 16.1 în actiune.

4. **Din nou cele două zaruri**. Iată o facilitate obtinută prin scrierea $X = X_1 + X_2$, cu $X_i$ punctele afisate de zarul $i$. Din exemplul 1 se stie că $E(X_1) = E(X_2) = 3{,}5$. Dar conform teoremei 16.1 $E(X) = E(X_1) + E(X_2) = 7$.
5. **Din nou la ruletă**. Se presupune că jocul la ruletă se prelungeste la 1000 de lansări. Fie $X$ câstigul asteptat, totuna cu câstigul mediu: $X = X_1 + X_2 + \ldots + X_{1000}$ cu $X_i$ câstigul net în jocul $i$. Se cunoaste $E(X_i) = -1/19$ pentru fiecare $i$. În 1000 de jocuri este de asteptat o pierdere $E(X) = E(X_1) + E(X_2) + \ldots + E(X_{1000}) = 1000 \times (-1/19) \approx -53$. Asadar în 1000 de jocuri, jucătorul se poate astepta la o pierdere de 53 de lei.
6. **Temele de casă**. Reluăm problema distribuirii aleatoare a temelor de casă ale celor 20 de studenti. Variabila aleatoare $X$ era numărul de studenti care primeau propria lor lucrare după amestecarea lucrărilor (sau numărul punctelor fixe dintr-o permutare). Cu teorema 16.1 $X$ se poate scrie ca o sumă de variabile aleatoare mai simple. Dar deoarece $X$ contorizează

numărul de ori în care ceva anume se întâmplă, o putem scrie ca o sumă uzând de următorul truc:

$$X = X_1 + X_2 + \ldots + X_{20} \text{ cu } X_i = \begin{cases} 1 & \textit{studentul i primeste propria lucare} \\ 0 & \textit{în celelalte cazuri} \end{cases}$$

Putină reflectie asupra acestei relatii: toti $X_i$ sunt variabile aleatoare. Ce reprezintă o relatie care include astfel de variabile? Reprezintă faptul că *în fiecare punct* $\omega$, avem $X(\omega) = X_1(\omega) + X_2(\omega) + \ldots + X_{20}(\omega)$. De ce acesta este un adevăr?

O variabilă aleatoare care ia două valori 0 si 1 precum $X_i$ este denumită o variabilă aleatoare *indicator* al evenimentului respectiv (în acest caz evenimentul este "studentul *i* obtine propria sa lucrare"). Pentru o variabilă aleatoare indicator, media este în mod particular simplu de calculat si anume:

$$E(X_i) = 0\text{x}\Pr[X_i = 0] + 1\text{x}\Pr[X_i = 1] = \Pr[X_i = 1]$$

Dar în cazul în spetă, $\Pr[X_i = 1]$ = Pr[studentul *i* să obtină propria sa lucrare] = 1/20.

Acum se poate aplica teorema 16.1

$$E(X) = E(X_1) + E(X_2) + \ldots + E(X_{20}) = 20\text{x}(1/20) = 1$$

Vedem astfel că numărul asteptat (mediu) de studenti care-si recapătă propria lucrare într-o grupă de 20 este 1. Exact acelasi rezultat s-a obtinut si pentru permutările de 3 obiecte. Si se poate dovedi usor că $E(X) = 1$ pentru orice $n$: deoarece putem scrie $X = X_1 + X_2 + \ldots + X_n$ si $E(X_i) = 1/n$ pentru orice $i$.

Asadar, numărul mediu de puncte fixe ale unei permutări aleatoare de $n$ obiecte este totdeauna 1, indiferent de $n$. Uimitor dar adevărat.

7. **Aruncarea monedei**. Se aruncă o monedă corectă de 100 de ori. Fie variabila aleatoare $X$ care reprezintă numărul de steme. Ca si în exemplul anterior, pentru a profita de avantajele teoremei 16.1, se scrie $X = X_1 + X_2 + \ldots + X_{100}$ cu $X_i$ variabla aleatoare indicator al evenimentului "la aruncarea *i* rezultatul a fost stema". Deoarece moneda este corectă, $E(X_i) = \Pr[X_i = 1] = 1/2$. Teorema 16.1 conduce la

$$\text{E(X)} = \sum_{i=1}^{100} \frac{1}{2} = 100\text{x}(1/2) = 50$$

Mai general, numărul mediu (asteptat) de steme în $n$ aruncări ale unei monede corecte este $n/2$. În cazul unei monede incorecte, numărul mediu (asteptat) de steme în $n$ aruncări este $pn$.

8. **Bile si cutii**. Se aruncă $m$ bile în $n$ cutii. Fie $X$ variabila aleatoare care numără bilele căzute în cutia 1. Variabila $X$ se comportă exact ca numărul de steme din aruncarea unei monede incorecte de $n$ ori, cu probabilitatea stemei $1/n$ (de ce?). Din exemplul imediat anterior rezultă $E(X) = m/n$. în cazul special $m = n$ numărul asteptat de bile în cutia 1 este 1. Încercarea de a calcula media direct din distributia lui $X$ este destul de complicată: apare

necesar calculul unor coeficienti binomiali, ceea ce nu este totdeauna comod (a se vedea problema încărcării din lectia precedentă).
Iată un alt exemplu pe acelasi spatiu al probelor. Fie $Y$ variabila aleatoare care numără cutiile goale. Distributia lui $Y$ este ceva oribil de analizat: pentru a simti disconfortul ar fi o cale, aceea a enumerării pentru $m = n = 3$ (3 bile, 3 cutii). Cu toate acestea, calculul mediei $E(Y)$ este facil prin utilizarea teoremei 16.1. Ca deobicei se scrie $Y = Y_1 + Y_2 + \ldots + Y_n$, unde $Y_i$ este variabila aleatoare indicator al evenimentului "cutia $i$ este goală". Ca de obicei, mediile variabilelor $Y_i$ sunt usor de evaluat

$$E(Y_i) = \Pr[Y_i = 1] = \left(1 - \frac{1}{n}\right)^m$$

conform si cu un rezultat dintr-o lectie anterioară. Aplicând acum teorema 16.1 se obtine

$$E(Y) = \sum_{i=1}^{n} E(Y_i) = n\left(1 - \frac{1}{n}\right)^m$$

o formulă foarte simplă si simplu de obtinut.
Să vedem acum ce se întâmplă în cazul particular $m = n$. în acest caz $E(Y) = n\left(1 - \frac{1}{n}\right)^n$. Pentru $n$ suficient de mare, paranteza cu exponentul ei se poate aproxima cu $1/e$, astfel încât

$$E(Y) \approx (n/e) \approx 0{,}368n$$

Finalul permite aprecierea numărului mediu de cutii goale în cazul 1000 de bile, 1000 de cutii: numărul asteptat de cutii goale este 368.

# Lecția 17

## Câteva distributii importante

**Problemă**: O monedă incorectă, cu probabilitatea stemei $p$ este aruncată repetat până apare prima stemă. Care este numărul asteptat de aruncări (media numărului de aruncări) până la aparitia primei steme?

Ca deobicei, primul pas spre solutie este definirea spatiului probelor $\Omega$. Un moment numai de gândire spune că $\Omega = \{S, RS, RRS, RRRS, \ldots\}$, adică multimea cuvintelor pe alfabetul $\{S, R\}$, care se isprăvesc cu $S$ si au celelalte caractere $R$. Este primul exemplu de spatiu al probelor infinit (desi încă discret). Care este probabilitatea unui punct, de pildă $\omega = RRS$? Deoarece aruncările succesive au rezultatele independente, $\Pr[RRS] = (1-p)(1-p)p = (1-p)^2 p$.

În general, pentru orice secventă $\omega \in \Omega$ de lungime $i$ avem $\Pr[\omega] = (1-p)^{i-1}p$. Pentru a fi siguri pe consistenta acestor probabilităti, verificăm suma acestora, sumă care ar trebui să fie 1. Într-adevăr, pentru secventele de lungime $i \geq 1$, avem

$$\sum_{\omega \in \Omega} \Pr[\omega] = \sum_{i=1}^{\infty} (1-p)^{i-1} p = p \sum_{i=1}^{\infty} (1-p)^{i-1} = p \frac{1}{1-(1-p)} = 1$$

suma unei serii geometrice.

Acum, fie $X$ variabila aleatoare care numără aruncările monedei în fiecare caz, adică $X(\omega)$ este lungimea secventei $\omega$. Care este media ei? În pofida faptului că variabila aleatoare $X$ numără ceva, nu există vreun motiv evident de a o scrie ca sumă de variabile aleatoare simple cum s-a procedat altădată, în lectia anterioară. Ca alternativă, să încercăm calculul direct. Variabila aleatoare $X$ este destul de simplă: ea ia valorile 1, 2, …, $i$, … respectiv cu probabilitătile $\Pr[X = i] = (1-p)^{i-1}p$. Din definitia mediei

$$E(X) = \sum_{i=1}^{\infty} i(1-p)^{i-1} p = p \sum_{i=1}^{\infty} i(1-p)^{i-1}$$

Seria aceasta se poate trata pe mai multe căi. Una dintre ele este dată de următoarea

**Teorema 17.1**: Fie $X$ o variabilă aleatoare care ia numai valori întregi nenegative. Atunci

$$E(X) = \sum_{i=1}^{\infty} \Pr[X \geq i]$$

**Demonstratie**: Mai întâi o conventie de notatie: $p_i = \Pr[X = i]$, pentru $i = 0, 1, 2, \ldots$. Din definitia mediei, avem

$E(X) = (0 \times p_0) + (1 \times p_1) + (2 \times p_2) + (3 \times p_3) + (4 \times p_4) + \ldots$
$= p_1 + (p_2 + p_2) + (p_3 + p_3 + p_3) + (p_4 + p_4 + p_4 + p_4) + \ldots$
$= (p_1 + p_2 + p_3 + p_4 + \ldots) + (p_2 + p_3 + p_4 + \ldots) + (p_3 + p_4 + \ldots) + (p_4 + \ldots)$
$+ \ldots = \Pr[X \geq 1] + \Pr[X \geq 2] + \Pr[X \geq 3] + \Pr[X \geq 4] + \ldots$

În linia a treia, termenii au fost regrupati în sume infinite convenabile. Se poate constata usor că linia a patra decurge din cea de a treia. Notatia "…" poate este putin ambiguă dar nu este greu de interpretat în semnificatia ei corectă.

□

Cu teorema 17.1 calculul mediei $E(X)$ define facil. Observatia cheie este aceea că pentru variabila aleatoare $X$, $\Pr[X \geq i] = (1 - p)^{i-1}$. Ce înseamnă aceasta? Înseamnă că sunt necesare cel putin $i$ aruncări pentru a se produce evenimentul "$X \geq i$". Si probabilitatea acestui eveniment este exact $(1 - p)^{i-1}$.

Acum, punând acest rezultat în teorema 17.1 se obtine

$$E(X) = \sum_{i=1}^{\infty} \Pr[X \geq i] = \sum_{i=1}^{\infty} (1 - p)^{i-1} = \frac{1}{1 - (1 - p)} = \frac{1}{p}$$

ceea ce spune că numărul mediu de aruncări ale monedei incorecte până la aparitia primei steme este $1/p$. Dacă moneda este corectă, numărul mediu este 2.

## Distributia geometrică

Distributia variabilei aleatoare $X$ care reprezintă numărul de aruncări ale monedei până la prima aparitie a stemei are un nume special: *distributia geometrică cu parametrul p* (semnificatia numărului $p$ este cunoscută).

**Definitia 17.1** (**distributia geometrică**): O variabilă aleatoare $X$ pentru care $\Pr[X = i] = (1 - p)^{i-1}p$ pentru $i = 1, 2, 3, \ldots$ se spune că are o distributie geometrică cu parametrul $p$.

Dacă se reprezintă grafic distributia variabilei aleatoare $X$ distribuită geometric, se observă o scădere a probabilitătilor $\Pr[X = i]$ pe măsură ce $i$ creste. De retinut faptele deja demonstrate, în

**Teorema 17.2**: Pentru o variabilă aleatoare $X$ distribuită geometric cu parametrul $p$ avem

1. $E(X) = 1/p$
2. $\Pr[X \geq i] = (1 - p)^{i-1}$ pentru $i = 1, 2, 3, \ldots$

Distributia geometrică apare frecvent în aplicatii deoarece este frecvent necesar a evalua cât timp trebuie asteptat înainte ca un eveniment anumit să se producă: câte utilizări înainte ca un sistem să clacheze, câte trageri înainte de a lovi o tintă, câte voturi până la primul în favoarea unui anumit partid etc. Sectiunea următoare aduce în discutie o aplicatie cu implicatii mai largi.

## Problema colectionarului de cupoane

Încercăm să colectionăm un set de $n$ carduri de baseball diferite. Obtinem cardurile prin cumpărarea de cutii cu cereale: fiecare cutie contine exact un card si este egal probabil a găsi într-o cutie oricare din cele $n$ carduri. Câte cutii trebuie cumpărate până când colectia va contine cel putin un exemplar din fiecare card?

Spatiul probelor este aici similar cu cel din exemplul precedent cu aruncarea monedei, dar ceva mai complicat. Spatiul constă aici din toate secventele $\omega$ pe alfabetul $\{1, 2, \ldots, n\}$ astfel încât

1. $\omega$ contine fiecare simbol 1, 2, …, $n$ cel putin o dată
2. simbolul final este continut numai o dată.

Pentru oricare din aceste puncte $\omega$, probabilitatea este $\Pr[\omega] = 1/n^i$, cu $i$ lungimea lui $\omega$ (de ce?). Dar nu este usor a spune câte puncte $\omega$ sunt de lungime $i$ (ca exercitiu, a se încerca pentru cazul $n = 3$). Dificil este si a stabili distributia variabilei aleatoare $X$ care este lungimea secventei, cu alte cuvinte numărul de cutii cumpărate.

Din fericire, se poate calcula media $E(X)$ foarte usor, utilizând liniaritatea mediei si faptul că este cunoscută media pentru distributia geometrică. Ca deobicei se scrie $X = X_1 + X_2 + \ldots + X_n$ pentru o variabilă aleatoare $X_i$ potrivită. Dar ce înseamnă "potrivită"? Natural, o cale de încercat este a face $X_i$ egal cu numărul de cutii de cumpărat în încercarea de a obtine cel de al $i$-lea card (care începe imediat după ce s-a obtinut cardul $(i - 1)$). Cu această definitie, a se verifica mai întâi dacă suma care reprezintă pe $X$ este cea care trebuie.

Cum arată distributia lui $X_i$? Ei bine, variabila $X_1$ este banală, trivială: orice s-ar întâmpla, totdeauna se obtine un card nou la prima cutie cumpărată pentru că nu există vreunul deja colectat. Asadar, $\Pr[X_1 = 1] = 1$ si $E(X_1) = 1$.

Dar $X_2$? Când se cumpără încă o cutie, vechiul card se poate obtine cu probabilitatea $1/n$, unul nou se poate obtine cu probabilitatea $(n - 1)/n$. Se poate gândi cumpărarea unei noi cutii ca o aruncare cu o monedă incorectă cu probabilitatea stemei $p = (n - 1)/n$. Astfel, $X_2$ este numărul de aruncări până când apare prima stemă si are o distributie geometrică cu parametrul $p$ tocmai evaluat. Se scrie deci adevărul $E(X_2) = n/(n - 1)$.

Pentru $X_3$ se rationează similar. Probabilitatea unui card de un tip diferit de cele două deja existente în colectie est $p = (n - 2)/n$ si $E(X_3) = n/(n - 2)$.

Cu o argumentatie de acelasi gen, pentru $i = 1, 2, \ldots, n$ se pot detecta distributii geometrice de parametru $p = (n - i + 1)/n$ cu mediile $E(X_i) = n/(n - i + 1)$.

În final, tinând seamă de liniaritatea mediei se obtine

$$E(X) = \sum_{i=1}^{n} E(X_i) = \frac{n}{n} + \frac{n}{n-1} + \ldots + \frac{n}{2} + \frac{n}{1} = n\sum_{i=1}^{n} \frac{1}{i}$$

care este expresia exactă a mediei $E(X)$. O expresie mai comodă si o bună aproximare se obtine prin înlocuirea

$$\sum_{i=1}^{n} \frac{1}{i} \approx \ln n + \gamma$$

cu $\gamma = 0{,}5772\ldots$ *constanta lui Euler*.

Astfel, numărul asteptat de cutii cu cereale necesar a fi cumpărate pentru a colecta $n$ carduri este de cca. $n(\ln n + \gamma)$. Această aproximare este foarte bună chiar pentru valori relativ mici ale lui $n$. De pildă, pentru $n = 100$ este necesar a cumpăra cca. 518 cutii cu produs.

**Distributia binomială**

Fie $X$ numărul de steme în $n$ aruncări ale unei monede cu defect, cu probabilitatea unei steme egală cu $p$. Este clar că variabila $X$ ia valorile 0, 1, …, $n$. Într-o etapă mai timpurie a acestui curs s-a stabilit că distributia acestei variabile aleatoare este

$$\Pr[X = i] = C_n^i p^i (1 - p)^{n-i}$$

**Definitia 17.2** (**distributia binomială**): O variabilă aleatoare care are distributia de mai sus este o variabilă aleatoare cu o *distributie binomială* de parametri $n$ si $p$.

Dintr-o lectie anterioară ne reamintim că $E(X) = np$. O diagramă a unei repartitii binomiale cu $n$ mare arată o formă asemănătoare unui clopot, cu un vârf pronuntat în apropierea mediei $np$.

**Distributia Poisson**

Aruncarea a $n$ bile în $n/\lambda$ cutii (cu $\lambda$ o constantă) produce o variabilă aleatoare $X$ – numărul de bile căzute în cutia 1 – care are o repartitie binomială cu parametrii $n$ si $p = \lambda/n$ si cu media $E(X) = np = \lambda$ (de ce?).

Facem acum o privire mai detaliată asupra distributiei lui $X$, un caz special al distributiei binomiale în care parametrul $p$ este de forma $\lambda/n$. Se notează $p_i = \Pr[X = i]$ pentru $i = 0, 1, 2, \ldots$.

Pentru $p_0$ avem

$$p_0 = \Pr[\text{toate bilele ratează cutia 1}] = \left(1 - \frac{\lambda}{n}\right)^n$$

si valoarea acestei probabilităti tinde către $e^{-\lambda}$ când $n \to \infty$. Asadar, pentru $n$ foarte mare, $p_0$ este practic valoarea constantă $e^{-\lambda}$.

Pentru celelalte probabilităti $p_i$, se stie de la distributia binomială că

$$p_i = C_n^i \left(\frac{\lambda}{n}\right)^i \left(1 - \frac{\lambda}{n}\right)^{n-i}$$

Deoarece se stie cum arată $p_0$, scriem expresia

$$\frac{p_1}{p_0} = \frac{n\frac{\lambda}{n}\left(1 - \frac{\lambda}{n}\right)^{n-1}}{\left(1 - \frac{\lambda}{n}\right)^n} = \frac{\lambda}{1 - \frac{\lambda}{n}} = \frac{n\lambda}{n - \lambda}$$

care tinde către $\lambda$ pe măsură ce $n$ creste foarte mult. De aici, pentru $n$ suficient de mare, $p_1 = \lambda e^{-\lambda}$.
Pentru evaluarea lui $p_2$, avem succesiv

$$\frac{p_2}{p_1} = \frac{C_n^2\left(\frac{\lambda}{n}\right)^2\left(1-\frac{\lambda}{n}\right)^{n-2}}{n\frac{\lambda}{n}\left(1-\frac{\lambda}{n}\right)^{n-1}} = \frac{n-1}{2}\frac{\lambda}{n}\frac{1}{1-\frac{\lambda}{n}} = \frac{n-1}{n-\lambda}\frac{\lambda}{2}$$

care tinde către $\lambda/2$ odată cu $n \to \infty$. Astfel, pentru $n$ mare, probabilitatea $p_2$ este practic $\frac{\lambda^2}{2}e^{-\lambda}$.
O procedură similară, trecând prin evaluarea raportului $p_i/p_{i-1}$ produce succesiv

$$\frac{p_i}{p_{i-1}} = \frac{C_n^i\left(\frac{\lambda}{n}\right)^i\left(1-\frac{\lambda}{n}\right)^{n-i}}{C_n^{i-1}\left(\frac{\lambda}{n}\right)^{i-1}\left(1-\frac{\lambda}{n}\right)^{n-i+1}} = \frac{n-i+1}{i}\frac{\lambda}{n}\frac{n}{n-\lambda} = \frac{n-i+1}{n-\lambda}\frac{\lambda}{i}$$

Expresia ultimă tinde către $\lambda/i$ odată cu $n \to \infty$. De aici

$$p_i \to \frac{\lambda^i}{i!}e^{-\lambda} \quad \text{dacă} \quad n \to \infty$$

(cititorul ar trebui să verifice aceasta) adică atunci când $n$ este mare fată de $i$, probabilitatea ca exact $i$ bile să cadă în cutia 1 este foarte apropiată de limita dată în ultima expresie. Acest fapt motivează definitia care urmează.
**Definitia 17.3** (**distributia Poisson**): O variabilă aleatoare $X$ pentru care

$$\Pr[X = i] = \frac{\lambda^i}{i!}e^{-\lambda} \quad \text{pentru} \quad i = 0, 1, 2, \ldots$$

este o variabilă aleatoare cu o *distributie Poisson* cu parametrul $\lambda$.
Pentru a verifica dacă această definitie este validă, trebuie verificată suma probabilităților $\Pr[X = i]$ pentru toate valorile $i$ posibile, sumă care trebuie să fie egală cu 1. Într-adevăr

$$\sum_{i=0}^{\infty}\frac{\lambda^i}{i!}e^{-\lambda} = e^{-\lambda}\sum_{i=0}^{\infty}\frac{\lambda^i}{i!} = e^{-\lambda}e^{\lambda} = 1$$

S-a utilizat în etapa a doua o serie Taylor cu suma binecunoscută.
Care este media acestei variabile aleatoare? Un calcul simplu conduce la

$$E(X) = \sum_{i=0}^{\infty} i\Pr[X = i] = \sum_{i=0}^{\infty} i\frac{\lambda^i}{i!}e^{-\lambda} = e^{-\lambda}\sum_{i=0}^{\infty} i\frac{\lambda^i}{i!} = \lambda e^{-\lambda}\sum_{i=1}^{\infty}\frac{\lambda^{i-1}}{(i-1)!} = \lambda e^{-\lambda}e^{\lambda} = \lambda$$

Asadar, media variabilei aleatoare $X$ distribuite poissonian cu parametrul $\lambda$ este $E(X) = \lambda$.
O reprezentare grafică a distributiei Poisson în functie de $i$ care parcurge multimea numerelor naturale, evidentiază o crestere, atingerea unui maxim si

apoi o descrestere. Vârful este în apropierea valorii medii, adică are loc la $i = \lfloor \lambda \rfloor$.

Am văzut că distributia Poisson rezultă ca limită a numărului de bile în cutia 1 când $n$ bile sunt aruncate în $n/\lambda$ cutii. Cu alte cuvinte, ea este limita distributiei binomiale cu parametrii $n$ si $p = \lambda/n$ când $n \to \infty$ cu $\lambda$ constant. Distributia Poisson este larg acceptată ca model pentru asa-numitele evenimente rare cum sunt apelurile telefonice de conectat, emisiile radioactive, încrucisările între cromozomi etc. Acest model este potrivit ori de câte ori evenimentele sunt presupuse întâmplătoare cu o densitate constantă $\lambda$ oarecare într-o regiune continuă în timp sau în spatiu, astfel încât evenimentele sunt în subregiuni disjuncte si sunt independente. Se poate arăta atunci că numărul de evenimente care se produc într-o regiune de dimensiune unitară se supune unei legi de distributie poissoniene cu parametrul $\lambda$.

Iată un exemplu frivol. Se presupune că un anumit soi de prăjituri este făcut dintr-un aluat care contine în medie 3 stafide la ligură. Fiecare prăjitură este făcută din două linguri de aluat. Este de asteptat ca numărul de stafide într-o prăjitură să se distribuie poissonian cu parametrul $\lambda = 6$. Iată câteva probabilităti asociate diferitelor numere posibile de stafide într-una din prăjituri luată la întâmplare:

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Pr[$X=i$]** | **0,002** | **0,015** | **0,045** | **0,089** | **0,134** | **0,161** | **0,161** | **0,138** | **0,103** | **0,069** |

De retinut că distributia Poisson rezultă natural în cel putin două contexte distincte importante. Alături de distributiile binomială si normală (care urmează a fi cunoscută), distributia Poisson este una din cele trei distributii cu care se lucrează cel mai frecvent.

# Lecția 18

## Dispersia unei variabile aleatoare

**Problemă**: La fiecare pas în timp al unei secvente de aruncări ale unei monede corecte, dacă apare stema fac un pas la dreapta, dacă apare reversul fac un pas la stânga. Cât de departe este de asteptat a mă afla după *n* pasi?
Notând pasul la dreapta cu +1 si pasul la stânga cu –1, se poate descrie un spatiu probabilistic alcătuit din toate cuvintele de lungime *n* pe alfabetul cu două caractere {±1}, fiecare cuvânt cu probabilitatea $1/2^n$. Fie variabila aleatoare $X$ pozitia relativ la punctul de plecare 0 după *n* deplasări. Asta înseamnă

$$X = X_1 + X_2 + \ldots + X_n \text{ cu } X_i = \begin{cases} +1 & \textit{pentru aruncarea } i = \textit{stema} \\ -1 & \textit{pentru aruncarea } i = \textit{reversul} \end{cases}$$

Evident că $E(X) = 0$. Cea mai directă si riguroasă cale de a demonstra această evidență constă în a calcula $E(X_i) = (1/2)(+1) + (1/2)(-1) = 0$. Din liniaritatea mediei rezultă $E(X) = 0$ si după *n* pasi este de asteptat să mă găsesc la 0. Dar această valoare nu este foarte plină de informatie si lipsa se datorează faptului că deviatile pozitive si negative se compensează reciproc.
Problema formulată mai devreme poate să însemne si altceva: care este media valorii $|X|$, distanta fată de 0? Dar în loc de variabila aleatoare $|X|$ care este întrucâtva greu de manipulat din cauza luării modulului, se ia în considerarea variabila aleatoare $X^2$. Si aceasta face din deviatiile de la pozitia 0 numere toate pozitive si dă o măsură bună a distantei parcurse, cu toate că deoarece este o distanță la pătrat, va fi necesar a lua în final rădăcina pătrată.
Să calculăm

$$E(X^2) = E((X_1 + X_2 + ... + X_n)^2)$$
$$= E\left( \sum_{i=1}^{n} X_i^2 + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} X_i X_j \right)$$
$$= \sum_{i=1}^{n} E(X_i^2) + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} E(X_i X_j)$$

În linia ultimă s-a utilizat liniaritatea mediei. Să calculăm acum $E(X_i^2)$ si $E(X_i X_j)$, aceasta din urmă pentru $i \neq j$. Deoarece $X_i = \pm 1$, pătratul acestei variabile este permanent 1 si, în consecintă, media $E(X_i^2) = 1$. Despre cealaltă medie, deoarece $X_i$ si $X_j$ sunt variabile aleatoare independente are loc relatia $E(X_i X_j) = E(X_i)E(X_j) = 0$. (Două variabile aleatoare $X$ si $Y$ sunt independente dacă evenimentele "$X = a$" si "$Y = b$" sunt independente pentru toate perechile de valori $a$, $b$. Dacă $X$, $Y$ sunt independente, atunci $E(XY) = E(X)E(Y)$; se propune ca temă demonstrarea acestei egalităti. De retinut că egalitatea mentionată este în general falsă pentru $X$, $Y$ arbitrare; un exemplu este chiar $E(X_i^2)$ din discutia de mai devreme). Introducând aceste valori în relatia de mai sus, se obtine

$$E(X^2) = (n \times 1) + 0 = n$$

Astfel s-a demonstrat că valoarea medie a pătratului distantei fată de 0 este $n$. O interpretare a acestui fapt este aceea că după $n$ pasi ne putem astepta la o distanțare medie față de 0 de cca. $\sqrt{n}$. Aici trebuie oarecare prudență: nu putem spune pur si simplu că $E(|X|) = \sqrt{E(X^2)} = \sqrt{n}$ (de ce?). Vom vedea imediat cum se pot face deductii riguroase asupra lui $|X|$ din evaluarea mediei $E(X^2)$.

Pentru moment să vedem în $E(X^2)$ o măsură intuitivă a împrăstierii, a dispersării valorilor pe care le poate lua variabila aleatoare $X$. De fapt, pentru o medie $E(X) = \mu$ mult mai generală, este interesantă media unei alte distante la pătrat, $E((X - \mu)^2)$, distanța la pătrat față de medie. În exemplul deplasării aleatoare discutat aici $\mu = 0$ si de aceea $E((X - \mu)^2)$ devine $E(X^2)$.

**Definitia 18.1** (**dispersia**): Pentru o variabilă aleatoare $X$ cu media $E(X) = \mu$, *dispersia* (varianța) lui $X$ se defineste ca

$$Var(X) = E((X - \mu)^2)$$

Rădăcina pătrată a acestei dispersii, $\sqrt{Var(X)}$ este numită *abatere medie pătratică* sau *deviație standard*.

Calculul deviației standard este un gen de operatie inversă ridicării la pătrat a distanțelor înainte de mediere. Astfel deviatia standard este o mărime pe aceeasi scală cu însăsi variabila aleatoare. Deoarece dispersia si abaterea medie pătratică sunt într-o legătură algebrică simplă nu are importantă cu care din ele alegem să operăm. Totusi, mai obisnuit se lucrează cu dispersia. Pentru

problema pasilor aleatori de mai sus, am stabilit că $Var(X) = n$ si că deviatia standard a lui $X$ este $\sqrt{n}$ .

Observatia următoare, foarte la îndemână oferă o cale usor diferită de a calcula dispersia în multe cazuri.

**Teorema 18.1**: Pentru o variabilă aleatoare $X$ cu media $E(X) = \mu$ avem $Var(X) = E(X^2) - \mu^2$.

**Demonstratie**: Din definitia dispersiei, avem

$$Var(X) = E((X - \mu)^2) = E(X^2 - 2\mu X + \mu^2) = E(X^2) - 2\mu E(X) + \mu^2 = E(X^2) - \mu^2$$

În ultima fază a demonstratiei s-a tinut seamă de liniaritatea mediei.

□

Acum, câteva exemple de calcul al dispersiei.

1. **Zarul corect**. Fie $X$ numărul de puncte afisat de un zar corect. Reamintim că $E(X) = 7/2$. Mai e nevoie de $E(X^2)$, ceea ce se obtine printr-un calcul de rutină

   $$E(X^2) = (1/6)(1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2) = 91/6$$

   Din teorema de mai devreme

   $$Var(X) = E(X^2) - (E(X))^2 = (91/6) - (49/4) = 35/12$$

2. **Moneda incorectă**. Fie $X$ numărul de aruncări din $n$ cu rezultatul "stema" care individual are probabilitatea $p$ (adică $X$ are o distributie binomială cu parametrii $n$, $p$. Am calculat deja $E(X) = np$. Scriind ca de obicei

   $$X = X_1 + X_2 + \ldots + X_n \text{ cu } X_i = \begin{cases} 1 & \text{pentru aruncarea } i = \text{stema} \\ 0 & \text{pentru aruncarea } i = \text{reversul} \end{cases}$$

   avem

   $$E(X^2) = E((X_1 + X_2 + \ldots + X_n)^2)$$

   $$= \sum_{i=1}^{n} E(X_i^2) + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} E(X_i X_j) = np + n(n-1)p^2 = n^2 p^2 + np(1-p)$$

   În partea ultimă s-au utilizat adevărurile $E(X^2) = p$ si $E(X_iX_j) = E(X_i)E(X_j) = p^2$ (deoarece $X_i$ si $X_j$ sunt variabile aleatoare independente). De observat că sunt $n(n-1)$ perechi $(i, j)$ cu $i \neq j$.

   În final, se obtine $Var(X) = E(X^2) - (E(X))^2 = np(1-p)$. Ca exemplu, pentru o monedă corectă, media aparitiilor unei fete este $n/2$, iar deviatia standard este $\sqrt{n}/2$.

   De notat că $Var(X) = \Sigma_i Var(X_i)$, ceea ce a fost adevărat si pentru cazul mersului aleator. Nu este o coincidentă ci este un adevăr mai larg valabil atât timp cât variabilele aleatoare $X_i$ sunt independente. Asadar, în cazul *independenței*, dispersiile sumelor de variabile aleatoare se pot calcula foarte usor. Nu la fel este cazul de la exemplul 4 de mai jos.

3. **Distributia Poisson**. Fie $X$ o variabilă aleatoare distribuită poissonian cu parametrul $\lambda$, adică $\Pr[X = i] = (\lambda^i/i!)e^{-\lambda}$ pentru $i = 0, 1, 2, \ldots$ Se poate dovedi că $E(X) = \lambda$ (a se revedea sectiunea despre medii). Avem totodată si

$$E(X^2) = \sum_{i=0}^{\infty} i^2 e^{-\lambda} \frac{\lambda^i}{i!} = \lambda \sum_{i=1}^{\infty} i e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}$$

$$= \lambda \left( \sum_{i=1}^{\infty} (i-1) e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!} + \sum_{i=1}^{\infty} e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!} \right) = \lambda (\lambda + 1)$$

în final se obtine $Var(X) = E(X^2) - (E(X))^2 = \lambda$. Asadar, pentru o variabilă aleatoare distribuită conform legii lui Poisson, media si dispersia sunt egale.

4. **Numărul de puncte fixe**. Fie $X$ numărul de puncte fixe ale unei permutări de $n$ obiecte aleasă la întâmplare (vezi problema lucrărilor restituite aleator studentilor dintr-o grupă de 20). Am văzut că $E(X) = 1$ indiferent care este $n$. Pentru a calcula $E(X^2)$ se scrie

$$X = X_1 + X_2 + \ldots + X_n \text{ cu } X_i = \begin{cases} 1 & \textit{pentru i punct fix} \\ 0 & \textit{în celelalte cazuri} \end{cases}$$

Ca de obicei avem

$$E(X^2) = \sum_{i=1}^{n} E(X_i^2) + \sum_{i \neq j} E(X_i X_j)$$

Deoarece $X_i$ este o variabilă aleatoare indicator, avem $E(X_i^2) = \Pr[X_i = 1] = \frac{1}{n}$.

În cazul acesta mediile $E(X_iX_j)$ trebuie tratate cu grijă: nu se mai poate spune că variabilele aleatoare $X_i$ si $X_j$ sunt independente (de ce?). Dar deoarece $X_i$ si $X_j$ sunt indicatori, media $E(X_iX_j)$ se poate calcula direct:

$$E(X_iX_j) = \Pr[(X_i = 1) \wedge (X_j = 1)] = \Pr[i \text{ si } j \text{ să fie puncte fixe}] = \frac{1}{n(n-1)}$$

(Verificati întelegerea acestui calcul). Introducând în relatia de mai devreme se obtine

$$E(X^2) = n(1/n) + n(n-1)[1/n(n-1)] = 1 + 1 = 2$$

Astfel că $Var(X) = E(X^2) - (E(X))^2 = 2 - 1 = 1$. Dispersia si media sunt ambele egale cu 1. Ca si media, dispersia este independentă de $n$. Intuitiv, asta înseamnă că este improbabil să fie mai putine puncte fixe când numărul de obiecte $n$ este foarte mare.

**Inegalitatea lui Cebîşev**

Am văzut că, intuitiv, dispersia (sau deviatia standard) este o măsură a împrăstierii sau a devierii de la medie a unei variabile aleatoare. Scopul pasului următor este a face această măsură intuitivă mai precisă cantitativ. Ce se poate demonstra este următoarea

**Teorema 18.2** (**inegalitatea lui Cebîşev**): Pentru o variabilă aleatoare $X$ cu media $E(X) = \mu$ si pentru orice $\alpha > 0$

$$\Pr[|X - \mu| \geq \alpha] \leq \frac{Var(X)}{\alpha^2}$$

Înainte de a demonstra inegalitatea lui Cebîşev, să întelegem ce vrea ea să spună. Inegalitatea spune că probabilitatea unei deviatii date, $\alpha$, fată de medie în sus sau în jos, este cel mult $Var(X)/\alpha^2$. Cum era de asteptat, această probabilitate a deviatiei este mică dacă dispersia este mică. De aici un corolar imediat al inegalitătii lui Cebîsev:

**Corolarul 18.3**: Pentru o variabilă aletoare $X$ cu media $E(X) = \mu$ si cu deviatia standard $\sigma = \sqrt{Var(X)}$

$$\Pr[|X - \mu| \geq \beta\sigma] \leq \frac{1}{\beta^2}$$

**Demonstratie**: Se introduce $\alpha = \beta\sigma$ în inegalitatea lui Cebîşev.

□

Astfel, de exemplu, probabilitatea unei deviatii mai mari (să spunem) de două deviatii standard de orice parte a mediei este de cel mut 1/4. În acest sens, deviatia standard este o definitie bună a lărgimii, a împrăştierii unei distributii.

Mergem înapoi la inegalitatea lui Cebîsev. Pentru demonstrarea ei vom face următoarea limitare simplă care se aplică numai variabilelor aleatoare nenegative, adică cele care iau valori mai mari sau egale cu zero.

**Teorema 18.4** (**Inegalitatea lui Markov**): Pentru o variabilă aleatoare $X$ nenegativă cu media $E(X) = \mu$ si pentru $\alpha > 0$ oarecare

$$\Pr[X \geq \alpha] \leq \frac{E(X)}{\alpha}$$

**Demonstratie**: Din definitia mediei avem

$$E(X) = \sum_a a \Pr[X = a] \geq \sum_{a \geq \alpha} a \Pr[X = a] \geq \alpha \sum_{a \geq \alpha} \Pr[X = a] = \alpha \Pr[X \geq \alpha]$$

Pasul crucial este cel în care se foloseste faptul că $X$ ia numai valori nenegative (de ce n-ar fi valid pasul în cazul general?)

□

Acum se poate demonstra inegalitatea lui Cebîşev.

**Demonstratia teoremei 18.2**: Se defineste variabila aleatoare $Y = (X - \mu)^2$. De observat că $E(Y) = Var(X)$. De asemenea, probabilitatea de care suntem interesati, $\Pr[|X - \mu| \geq \alpha]$ este exact aceeasi cu $\Pr[Y \geq \alpha^2]$ (de ce?). Mai mult, $Y$ este evident nenegativă astfel că putem aplica inegalitate lui Markov pentru a obtine

$$\Pr[Y \geq \alpha^2] \leq \frac{E(Y)}{\alpha^2} = \frac{Var(X)}{\alpha^2}$$

ceea ce încheie demonstratia.

□

Să aplicăm inegalitatea lui Cebîsev pentru a avea un răspuns la întrebarea de la începutul acestei sectiuni, relativă la mersul cu pasi aleatori. Se cunoaste media si dispersia variabilei aleatoare $X$, distanta la pătrat fată de origine, după $n$ pasi: 0 respectiv $n$. Corolarul 18.3 spune că pentru orice $\beta > 0$, $\Pr[|X| \geq \beta\sqrt{n}] \leq 1/\beta^2$. Astfel, după $n = 10^6$ pasi, probabilitatea de a ne afla la mai mult de 10.000 de pasi distantă de origine este de cel mult 1/100.

Iată acum încă vreo câteva aplicatii ale inegalității lui Cebîşev (calculul algebric rămâne în seama cititorului).

1. **Aruncarea monedei**. Fie $X$ numărul de steme observate în $n$ aruncări cu o monedă corectă. Probabilitatea ca $X$ să devieze de la media $n/2$ cu mai mult de $\sqrt{n}$ este cel mult 1/4. Probabilitatea ca deviatia să fie dincolo de $5\sqrt{n}$ este cel mult 1/100.
2. **Distributia Poisson**. Fie $X$ o variabilă aleatoare poissoniană cu parametrul $\lambda$. Probabilitatea ca $X$ să devieze da la $\lambda$ cu mai mult de $2\sqrt{\lambda}$ este cel mult 1/4.
3. **Puncte fixe**. Fie $X$ numărul de puncte fixe într-o permutare aleatoare a $n$ obiecte. Se stie că $E(X) = Var(X) = 1$. Probabilitatea ca mai mult de (să spunem) 10 studenti să-si primească propria lucrare după amestecarea lucrărilor este de cel mult 1/100, oricât de mare ar fi $n$.

# Lecția 19

## Variabile aleatoare independente identic distribuite

### Estimarea incorectitudinii unei monede

**Întrebare**: Dorim să estimăm proportia de partizani ai unui partid politic în populatia României pe baza unui esantion aleator. Cât de mare trebuie să fie esantionul pentru a garanta estimarea la un nivel (să spunem) de 10% (în tremeni relativi) a valorii adevărate, cu probabilitatea de 0,95?
Asta este o problemă fundamentală de estimare statistică, întâlnită frecvent în împrejurări variate. Va fi dezvoltată o solutie simplă care utilizează numai inegalitatea lui Cebîsev. Pentru rezultate mai precise pot fi utilizate metode mai rafinate.
Se notează cu $n$ dimensiunea esantionului (a fi determinată) si cu $S_n$ numărul (aleator) de adepti ai partidului în acel esantion. (Indicele $n$ aminteste de faptul că variabila aleatoare depinde de dimensiunea esantionului). Estimarea va avea valoarea $A_n = S_n/n$.
Asa cum a fost frecvent cazul, se consideră util a scrie $S_n = X_1 + X_2 + \ldots + X_n$ cu $X_i = 1$ sau 0 după cum persoana $i$ este atasată sau nu partidului în discutie.
De notat că fiecare $X_i$ poate fi văzut ca o aruncare de monedă cu probabilitatea $p$ pentru stemă, necunoscută pe care încercăm a o estima. Aruncările monedei sunt independente (se consideră aici o esantionare cu înlocuire, adică se selectează o persoană din întreaga populatie care include si persoane deja intervievate; probabilitatea de a chestiona o aceeasi persoană de două ori este practic exclusă).
Care este media estimatiei?

$$E(A_n) = E\left(\frac{1}{n}S_n\right) = \frac{1}{n}E(X_1 + X_2 + ... + X_n) = \frac{1}{n}(np) = p$$

Astfel, pentru orice valoare a lui *n*, estimarea obtinută va avea media corectă *p* (O asemenea variabilă aleatoare este denumită adesea un *estimator absolut corect* al lui *p*). Anticipând, pe măsură ce *n* creste estimarea devine din ce în ce mai precisă. Asta se va confirma prin descresterea dispersiei pe măsură ce *n* creste, cu alte cuvinte cu *n* mai mare probabilitatea ca estimarea să fie departe de media *p* descreste.

Pentru a vedea aceasta trebuie calculată $Var(A_n)$. Si deoarece $A_n = \frac{1}{n}\sum_{i=1}^{n} X_i$, este necesar a sti cum se calculează dispersia unei sume de variabile aleatoare.

**Teorema 19.1**: Pentru orice variabilă aleatoare *X* si constantă *c*, are loc

$$Var(cX) = c^2 Var(X)$$

Pentru două variabile aleatoare *X*, *Y* independente are loc

$$Var(X + Y) = Var(X) + Var(Y)$$

**Demonstratie**: Din definitia dispersiei avem

$$Var(cX) = E((cX - E(cX))^2) = E((cX - cE(X))^2) = E(c^2(X - E(X))^2) = c^2 Var(X)$$

Demonstratia celei de a doua afirmatii este lăsată ca exercitiu. De observat că afirmatia a doua nu este valabilă decât dacă variabilele aleatoare *X* si *Y* sunt independente.

□

Utilizând teorema 19.1, se poate calcula $Var(A_n)$:

$$Var(A_n) = Var\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}Var\left(\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}\sum_{i=1}^{n} Var(X_i) = \frac{\sigma^2}{n}$$

în care s-a scris $\sigma^2$ pentru dispersia fiecăreia din variabilele $X_i$. Cum se vede, dispersia pentru $A_n$ variază invers proportional cu *n*. Acest fapt ne asigură că luând selectii cu *n* din ce în ce mai mare, probabilitatea unor deviatii mari de la media *p* devine din ce în ce mai mică.

Să folosim acum inegalitatea lui Cebîsev pentru a aprecia cât de mare trebuie să fie *n* pentru a asigura o acuratețe specificată asupra proportiei reale de adepti ai partidului, *p*. O cale naturală de a măsura aceasta este specificarea a doi parametri, $\varepsilon$ si $\delta$, ambii în intervalul (0, 1). Parametrul $\varepsilon$ controlează *eroarea* de estimare pe care suntem pregătiti să o acceptăm, iar $\delta$ exprimă *încrederea* pe care trebuie s-o avem în estimarea făcută. O versiune mai precisă a problemei formulate în introducerea acestei secțiuni este următoarea:

**Întrebare**: Pentru problema de estimarea a sustinerii electorale formulată mai sus, care trebuie să fie dimensiunea *n* a sondajului pentru a fi siguri că

$$\Pr[|A_n - p| \geq \varepsilon p] \leq \delta?$$

În forma initială, se punea $\varepsilon = 0{,}1$ si $\delta = 0{,}05$. De observat că $\varepsilon$ exprimă o eroare relativă adică o proportie din valoarea tintă *p*. Asta pare o conventie mai rezonabilă decât cea asupra erorii absolute exprimată ca $\Pr[|A_n - p| \geq \varepsilon]$, deoarece o eroare absolută, cum a fi ±0,1, ar putea fi nepotrivită în contextul

măsurării unei valori cum este $p = 0,5$ si sigur mai nepotrivită pentru $p = 0,05$. În contrast, eroarea relativă se potriveste în ambele situatii si în multe altele.
Prin aplicarea inegalitătii lui Cebîsev, se obtine un răspuns mai precis la întrebarea de mai sus. Deoarece se cunoaste $Var(A_n)$

$$\Pr[|A_n - p| \geq \varepsilon p] \leq \frac{Var(A_n)}{(\varepsilon\ p)^2} = \frac{\sigma^2}{np^2\varepsilon^2}$$

Pentru a avea ultima cantitate sub valoarea dorită $\delta$, punem

$$n \geq \frac{\sigma^2}{p^2}\frac{1}{\varepsilon^2\delta}$$

Acum, ne reamintim că $\sigma^2 = Var(X_i)$ este dispersia unei singure probe $X_i$. Astfel, deoarece $X_i$ este o variabilă aleatoare care ia una din două valori, 0 si 1, avem $\sigma^2 = p(1 - p)$ si inegalitatea de mai sus devine

$$n \geq \frac{1 - p}{p}\frac{1}{\varepsilon^2\delta}$$

Introducând $\varepsilon = 0,1$ si $\delta = 0,05$ se obtine că esantionul $n = 2000(1 - p)/p$ este suficient.
La acest moment ar putea apărea o îngrijorare: formula contine pe $p$, adică exact ceea ce trebuie estimat, evaluat. Această problemă poate fi ocolită astfel: se ia o valoare $p'$ despre care stim cu siguranță că este inferioară lui $p$ (de exemplu, în cazul în discutie, $p' = 1/3$). Se utilizează apoi în ecuatia de mai sus $p'$ în loc de $p$, ceea ce duce la un număr de observatii mai mult decât suficient (de ce?). În exemplul sondajului electoral, cu $p' = 1/3$ se obtine o dimensiune a selectiei $n = 2000x2 = 4000$.

**Estimarea mediei generale**

Dar dacă se doreste estimarea a ceva mai complex decât sustinerea electorală a unui partid, cum ar fi averea medie a cetătenilor tării? Se folseste aceeasi schemă ca mai sus, cu deosebirea că acum variabila aleatoare $X_i$ este averea persoanei $i$ din selectia prin care se face sondajul. Este clar că $E(X_i) = \mu$, averea medie (pe care încercăm să o estimăm). Estimatia este din nou $A_n$, media aritmetică a observatiilor $X_i$ pentru o listă de observatii de o dimensiune $n$ potrivită. Din nou, variabilele $X_i$ sunt independente si au toate aceeasi distributie: astfel de colectii de variabile aleatoare sunt numite uzual *independente si identic distribuite*, pe scurt *i.i.d*. Ca si mai devreme $E(X_i) = \mu$ si $Var(A_n) = \sigma^2/n$ cu $\sigma^2 = Var(X_i)$ dispersia lui $X_i$ (reamintim că singurele fapte relativ la variabilele aleatoare $X_i$ pe care le utilizăm sunt independenta si distribuirea lor identică).
Dintr-o ecuatie scrisă mai devreme, este suficient ca dimensiunea $n$ a sondajului să satisfacă relatia

$$n \geq \frac{\sigma^2}{\mu^2}\frac{1}{\varepsilon^2\delta}$$

Aici $\varepsilon$ si $\delta$ sunt ca si altădată eroarea dorită si respectiv nivelul de încredere. Dar de data aceasta nu se cunosc cele două cantităti $\mu$ si $\sigma^2$. Practic, se utilizează o valoare inferioară pentru $\mu$ si o valoare superioară pentru $\sigma^2$ (asa cum s-a utilizat o valoare mai scăzută pentru $p$ în problema sondajului electoral). Introducând aceste valori în ecuatie, ne asigurăm că dimensiunea sondajului este suficient de mare.

De exemplu, în problema averii medii se poate lua în sigurantă un posibil $\mu$ de 2.000 de dolari, poate chiar ceva mai mult. Totusi, existenta unor oameni ca Ion Țiriac impune o dispersie $\sigma^2$ mare. Desigur, dacă există cel putin o persoană cu o avere de 1 milard de dolari, admitând o medie $\mu$ relativ mică, se impune o dispersie de cel putin $(10^9)^2/(20\text{x}10^6) = 5\text{x}10^{10}$, cu populatia României de cca. 20 de milioane (a se verifica). Totusi, această persoană contribuie la medie cu numai $10^9/(20\text{x}10^6) = 50$. Nu există o cale simplă de a cuprinde această problemă prin sondajul uniform: inegalitatea distribuirii averilor înseamnă că dispersia este inevitabil foarte mare si va fi necesar un număr urias de persoane chestionate înainte de a include în sondaj pe cineva imens de bogat. Dacă nu sunt incluse astfel de persoane estimarea noastră poate fi foarte joasă fată de media reală.

Ca un exemplu suplimentar, să admitem că încercăm o investigatie estimativă asupra ratei medii de emisie a unei surse radioactive si să-i asociem acesteia o distributie Poisson cu un parametru necunoscut $\lambda$. Evident, $\lambda$ este media pe care încercăm s-o estimăm. În acest caz media este $\mu = \lambda$ si $\sigma^2 = \lambda$ (vezi lectia anterioară). Astfel, $\sigma^2/\mu^2 = 1/\lambda$. Aici o serie de observatii de dimensiunea $n = 1/(\lambda\varepsilon^2\delta)$ este suficientă. Un exemplu similar este cel al numărului mediu de cioburi de ciocolată într-o prăjitură, cu eroarea relativă 0,1 si cu nivelul de încredere de 95% si presupunând $\lambda \geq 4$ (media este cel putin 4). Dimensiunea suficientă a sondajului este de 500.

**Legea numerelor mari**

Metoda de estimare utilizată în sectiunile anterioare este bazată pe un principiu acceptat în viata de zi cu zi, *legea numerelor mari*. Aceasta afirmă că, dacă observăm o variabilă aleatoare în mai multe rânduri si calculăm media observatiilor atunci media va fi convergentă spre o *valoare unică*, valoarea medie a variabilei aleatoare. Cu alte cuvinte, medierea are tendinta de a netezi fluctuatiile mari si cu cât mai multe valori sunt utilizate în mediere cu atât mai bună este această netezire.

**Teorema 19.2** (**Legea numerelor mari**): Fie $X_1, X_2, \ldots, X_n$ variabile aleatoare i.i.d. cu media comună $\mu = E(X_i)$. Se defineste $A_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. Atunci, pentru orice $\alpha > 0$ avem

$$\Pr[|A_n - \mu| \geq \alpha] \to 0 \text{ pe măsură ce } n \to \infty$$

Această teoremă nu va fi demonstrată aici. Este de notat mai curând ceea ce spune ea: probabilitatea oricărei deviatii $\alpha$ de la medie, oricât de mică, tinde către zero dacă numărul $n$ de observatii mediate tinde la infinit. Astfel, luând un $n$ suficient de mare, se poate face probabilitatea unei deviatii date oricât de mică dorim (legea numerelor mari nu spune cât de mare trebuie să fie $n$ pentru a atinge o anumită precizie; pentru asta este necesar un alt instrument cantitativ cum este de pildă inegalitatea lui Cebîsev).
Se poate spune ceva încă mai tare decât legea numerelor mari si anume: distributia mediei de selectie $A_n$, pentru $n$ suficient de mare arată grafic ca o curbă sub formă de clopot, centrată pe media $\mu$. Lărgimea acestei curbe descreste cu $n$ astfel că se apropie de o formă foarte ascutită în apropierea mediei $\mu$. Acest fapt este cunoscut ca *Teorema limită centrală*.
Pentru a fi mai rigurosi, trebuie să definim "curba sub formă de clopot". Aceasta este asa-numita *distributie normală* si este prima (dar si unica) distributie non-discretă discutată în acest curs. Pentru variabilele aleatoare care iau valori reale pe un interval compact nu mai are sens a vorbi de $\Pr[X = a]$. Ca exemplu, se consideră o variabilă aleatoare $X$ care are o distributie uniformă pe intervalul [0, 1]. Pentru orice punct $0 \le a \le 1$, $\Pr[X = a] = 0$. Dar, foarte clar, $\Pr[0{,}25 \le X \le 0{,}75] = 0{,}5$. Prin urmare, în locul probabilităților punctuale $\Pr[X = a]$ apare necesitatea unui alt mod de a preciza distributia unei variabile aleatoare de tip continuu.
**Definitia 19.1** (**functia densitate de probabilitate**): Pentru o variabilă aleatoare $X$ cu valori reale într-un interval compact, o functie reală $f(x)$ este densitatea de probabilitate a lui $X$ dacă

$$\Pr[X \le a] = \int_{-\infty}^{a} f(x)dx$$

Astfel, imaginăm $f(x)$ a o curbă definitorie cu particularitatea că aria dintre curbă, verticalele $x = a$ si $x = b$ si axa absciselor este tocmai $\Pr[a \le X \le b]$. Totodată integrala functiei $f(x)$ pe întreaga axă reală este egală cu unitatea (de ce?). Ca exemplu, densitatea uniformă pe intervalul [0, 1] arată astfel:

$$f(x) = \begin{cases} 0 & x < 0 \\ 1 & 0 \le x \le 1 \\ 0 & x > 1 \end{cases}$$

(Verificati că această functie este conformă cu definitia unei functii densitate de probabilitate. Cum s-ar scrie functia care descrie o distributie uniformă pe intervalul [–1, 1]?)
Media unei variabile aleatoare continue se calculează analog celei pentru variabilele aleatoare discrete cu schimbarea sumelor în integrale. Astfel

$$E(X) = \int_{-\infty}^{\infty} xf(x)dx$$

si tot asa si pentru dispersie avem

$$Var(X) = E(X^2) - (E(X))^2 \text{ cu } E(X^2) = \int_{-\infty}^{\infty} x^2 f(x)dx$$

(Verificati că pentru distributia uniformă pe [0, 1] media este 1/2 si dispersia este 1/12.)

Suntem acum pregătiti să definim distributia normală.

**Definitia 19.2** (**distributia normală**): Distributia normală cu media $\mu$ si dispersia $\sigma^2$ este distributia cu functia densitate de probabilitate

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

Ratiunea pentru care apare factorul $\frac{1}{\sigma\sqrt{2\pi}}$ este una singură: integrala functiei de la $-\infty$ la $+\infty$ trebuie să dea unitatea.

Reprezentarea grafică a functiei $f(x)$ arată o formă de clopot simetrică fată de verticala $x = \mu$. Lărgimea, deschiderea acestui clopot este determinată de dispersia $\sigma^2$: 50% probabilitate pentru valorile care sunt continute într-un interval de $\pm 0{,}67\sigma$ si 99,7% probabilitate pentru valorile din intervalul $\pm 3\sigma$ în jurul mediei.

Acum se poate formula teorema limită centrală. Deoarece tratarea de până acum a distributiilor continue a fost mai curând schematică, ne vom folosi de un enunt destul de imprecis. Acesta poate fi făcut deplin riguros fără efort suplimentar exagerat în vreun fel.

**Teorema 19.3** (**Teorema limită centrală**): Fie $X_1, X_2, \ldots, X_n$ variabile aleatoare independente si identic distribuite cu media comună $\mu = E(X_i)$ si dispersia comună $\sigma^2 = Var(X_i)$, ambele mărginite. Se defineste o medie a $n$ observatii $A_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. Cu $n \to \infty$, distributia lui $A_n$ se apropie de distributia normală cu media $\mu$ si dispersia $\sigma^2/n$.

De notat că dispersia se micsorează odată cu cresterea lui $n$, asadar lărgimea unei curbe gen clopot scade cu un factor de $\sqrt{n}$.

Teorema limită centrală este realmente surprinzătoare. Dacă se ia media a $n$ observatii asupra oricărei variabile aleatoare $X$, distributia acelei medii este o curbă de forma unui clopot centrat pe $\mu = E(X)$. Astfel orice urmă a distributiei lui $X$ dispare când $n$ este suficient de mare: orice distributii, oricât de complexe dar cu media si dispersia finite arată ca o distributie normală dacă sunt mediate. Singurul efect al distributiei initiale este $\sigma^2$ care determină lărgimea clopotului pentru un anumit $n$ si prin asta rata cu care curba se transformă într-un vârf ascutit.

# Lecția 20

## Jocul Minesweeper si probabilitătile

Aplicatia ultimă relativ la probabilități dată aici se referă la jocul Minesweeper. Începem prin a discuta cum se joacă acest joc la modul optim; problema este probabil infezabilă, dar o apropiere bună este a încerca pătratul cel mai sigur. Aceasta motivează calculul probabilității ca fiecare pătrat să conțină o mină, ceea ce este o aplicatie interesantă a ceea ce am învătat atât la logica matematică cât si la probabilități.

## Jocul optim la Minesweeper

Am întâlnit multe cazuri în Minesweeper când analiza logică pură nu este sufucientă deoarece sunt situatii în care nici o miscare nu este garantat sigură. Asadar, nici un program Minesweeper nu poate câstiga 100% din jocuri. Am măsurat performanta prin proportia de victorii ca functie de densitatea initială a minelor. Poate fi găsit un algoritm care joacă mai bine decât oricare altul, adică are o probabilitate mai înaltă de câstig?
Primul pas în argumentatie este deplasarea de la *algoritmi*, care sunt infinit de multi, la *strategii*, care sunt multe dar în număr finit. Fundamental, o strategie spune ce trebuie făcut la fiecare punct al jocului. De observat că o strategie este specifică pentru un număr particular de mine $M$ si un număr total de pătrate $N$ (ca si de forma grilei de joc):
**Definitia 20.1** (**strategie**): O strategie pentru un joc Minesweeper este un arbore; fiecare nod este etichetat cu un pătrat de verificat si fiecare arc/ramură cu un număr de mine descoperite a fi adiacente acelui nod. Fiecare nod are 9 descendenti cu ramuri etichetate 0, …, 8; nici o etichetă de nod nu repetă eticheta unui antecesor; arborele este complet cu $N - M$ niveluri.

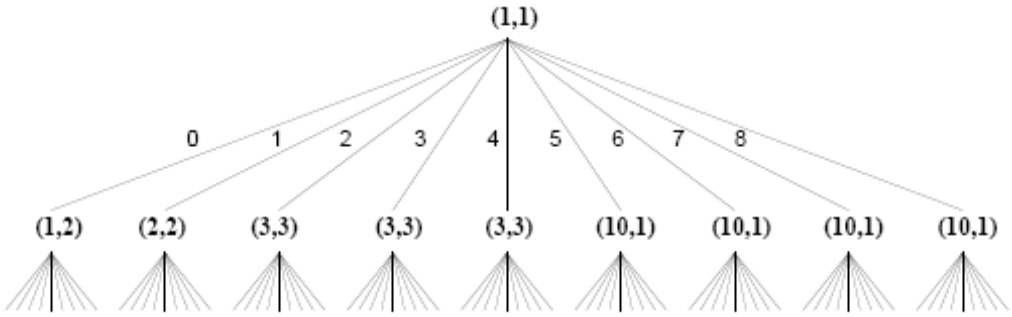
Figura de mai jos arată un exemplu. Chiar si pentru *M* si *N* fixate, există foarte multe strategii posibile:

$$\prod_{i=0}^{N-M-1} (N-i)^{9^i}$$

Pentru N = 6, M = 3, acestea sunt în număr de

685078892498860742907977265335757665463718376520000000000

Numărul este încă finit. Este usor de văzut că fiecare algoritm (cu final) pentru Minesweeper (cu *N*, *M* fixate si formă a grilei precizată) corespunde exact unei strategii. Mai mult, cu fiecare strategie există o probabilitate de câstig asociată. Desigur, este posibil a calcula exact acea probabilitate dar este probabil mai usor a o măsura prin încercări repetate.




Primele două niveluri ale unei strategii la Minesweeper

**Definita 20.2** (**optimalitate**): O strategie pentru Minesweeper este *optimă* dacă probabilitatea ei de câstig este cel putin la fel de mare ca aceea a oricărei alte strategii.

Clar, există o strategie optimă pentru orice *N*, *M* si formă a tablei date. Mai mult, se poate construi o metodă optimă de joc în general, pentru *N*, *M* si formă ale tablei de joc arbitrare: se enumeră fiecare strategie pentru acea configuratie si se joacă cea care este optimă. Profilul de performante ale acestui algoritm va domina pe al oricărui alt algoritm.

(De observat că existenta unui algoritm optimal se bazează pe numărul finit de strategii posibile. Sunt infinit de multi algoritmi si este usor de imaginat problemele pentru care orice algoritm dat poate fi totdeauna îmbunătătit, de exemplu prin consumarea unui timp de calcul ceva mai îndelungat, astfel încât nu există un algoritm optimal).

Din nefericire, aceasta este probabil situatia: *practic* nu există un algoritm optimal. În loc vom încerca o tratare mai simplă: se alege un pătrat sigur atunci când unul este cunoscut, altminteri se alege cel mai sigur pătrat judecat prin probabilitatea ca el să conțină o mină.

**Spatiul probabilităților**

Primul pas este cel al identificării setului de variabile aleatoare necesare:

- Ca si în cazul logicii propozitionale, dorim o variabilă booleană $X_{ij}$ care este adevărată dacă si numai dacă pătratul $(i, j)$ contine în realitate o mină. Utilizând $N$ pentru numărul total de pătrate, acestea se pot nota mai comod si cu $X_1, \ldots, X_N$.
- Vom avea totodată variabilele $D_{ij}$ care corespund continutului afisat pentru *acele k pătrate care au fost probate sau marcate ca mine*. Si de data aceasta, pentru simplitate, putem eticheta aceste variabile cu $D_1, \ldots, D_k$. Domeniul pentru fiecare din aceste variabile este ($m$, 0, 1, …, 8). Vom denumi aceste variabile *numerele* si variabilele corespunzătoare $X_{ij}$ vor fi numite *cunoscute*. (De notat că *numerele* includ mine marcate; admitem că numai minele garantate logic sunt marcate.)

În cele din urmă, vom folosi si $M$ pentru a exprima numărul total de mine (atât $N$ cât si $M$ sunt constante).

Ca un exemplu, se consideră afisajul următor:

| Mine rămase: 3 | | | | |
|---|---|---|---|---|
| 3 | | | | |
| 2 | *m* | | | |
| 1 | 2 | | 2 | |
| | 1 | 2 | 3 | 4 |

Aici $N$ este 12, $M$ este 4, (1, 1), (3, 1) si (1, 2) sunt *cunoscute*.

Acum trebuie scris spatiul probabilităților, adică distributia conjugată a tuturor variabilelor. Se dovedeste că cel mai usor este a face aceasta în două părti folosind regula lantului:

$$P(X_1, \ldots, X_N, D_1, \ldots, D_k) = P(D_1, \ldots, D_k | X_1, \ldots, X_N) P(X_1, \ldots, X_N)$$

De ce în acest mod? Deoarece (1) distributia initială a minelor, $P(X_1, \ldots, X_N)$, este usor de calculat deoarece minele sunt împrăstiate uniform si aleator si (2) variabilele afisajului sunt determinate de minele ascunse, astfel distributia conditionată $P(D_1, \ldots, D_k | X_1, \ldots, X_N)$ este si ea relativ usor de descris.

Mai întâi, distributia initială $P(X_1, \ldots, X_N)$. Să ne reamintim că primul pătrat ales este totdeauna sigur. Fără pierdere de generalitate, să denumim acest pătrat $X_1$: stim că $X_1 = \mathit{false}$. Pentru cele $N - 1$ pătrate rămase, $M$ mine sunt împrăstiate aleator. Sunt $C(N - 1, M)$ modalităti de a face această distribuire si fiecare combinatie este egal probabilă, astfel că avem

$$P(x_1, \ldots, x_N) = \begin{cases} 1 / C_{N-1}^{M} & \text{daca } \#(x_2, \ldots, x_N) = M \\ 0 & \text{altminteri} \end{cases}$$

unde $\#(x_2, \ldots, x_N)$ notează numărul de "*true*" în punctele de probă $x_2, \ldots, x_N$. Pentru verificare, se poate calcula $P(X_i)$, probabilitatea initială anticipată ca un pătrat $X_i$ ($i \neq 1$) să contină o mină: aceasta este dată de

$$\frac{C_{N-2}^{M-1}}{C_{N-1}^{M}} = \frac{M}{N-1}$$

care este în concordantă cu asteptările noastre.
Din presupunerea că distribuirea este uniformă si faptul că $P(X_i)$ sunt date pentru toti $i \neq 1$, există tentatia a scrie distributia combinată sub forma

$$P(X_2, \ldots, X_N) = P(X_2).P(X_2). \ldots .P(X_N)$$

ceea ce este gresit.
Independenta nu are loc deoarece numărul total de mine este fixat. De pildă, dacă primele M pătrate sunt toate minate, pătratul următor are probabilitatea de a fi minat egală cu zero.
Revenind la variabilele de afisaj, se cunosate că ele sunt determinate precis potrivit regulilor jocului Minesweeper, fiind dat continutul real al tuturor pătratelor, adică pentru toate combinatiile de valori $d_1, \ldots, d_k, x_1, \ldots, x_n$

$$P(d_1, \ldots, d_k | x_1, \ldots, x_N) = \begin{cases} 1 & \text{daca } d_1,...,d_k \text{ afiseaza corect } x_1,...,x_N \\ 0 & \text{altminteri} \end{cases}$$

**Găsirea de pătrate mai sigure**

Trebuie acum calculată pentru fiecare pătrat necunoscut $(i, j)$, cantitatea $P(X_{ij}|$ *cunoscute*, *numere*). Este dezvoltată mai departe o expresie simplă si calculabilă pentru această probabilitate într-o secventă de pasi.
Mai întâi, trebuie lucrat asupra expresiei spre o formă care contine termeni despre care avem cunostintă – anterior asupra tuturor variabilelor $X$ si probabilitatea conditionată pentru variabilele de afisaj fiind date aceste variabile. Expresia pentru $P(X_{ij}|$*cunoscute*, *numere*) nu contine alte variabile înafară de $X_{ij}$ pe care le numim *necunoscute*. De exemplu, în afisajul 4x3 de mai sus, $X_{ij}$ poate fi $X_{2,1}$ si sunt 8 pătrate *necunoscute*. Le introducem prin însumare (suma standard pe punctele de probă constitutive ale unui eveniment):

$$P(X_{ij}|e) = P(X_{ij}|cunoscute, numere) =$$
$$= \sum_{necunoscute} P(X_{ij}, \ necunoscute|cunoscute, numere)$$

De exemplu, cu 8 pătrate necunoscute, suma aceasta are $2^8 = 256$ de termeni. Ar fi de preferat o expresie cu numerele conditionate de variabilele $X$, ceea ce se poate obtine recurgând la regula lui Bayes:

$$P(X_{ij} \mid e) = \alpha \sum_{necunoscute} P(numere \mid cunoscute, X_{ij}, necunoscute)$$
$$P(X_{ij}, necunoscute \mid cunoscute) =$$
$$= \alpha \sum_{necunoscute} P(numere \mid cunoscute, X_{ij}, necunoscute)$$
$$P(X_{ij}, necunoscute, cunoscute) / P(cunoscute) =$$
$$= \alpha' \sum_{necunoscute} P(numere \mid cunoscute, X_{ij}, necunoscute)$$
$$P(X_{ij}, necunoscute, cunoscute)$$

Deocamdată e bine; variabilele *cunoscute*, $X_{ij}$, *necunoscute* constituie toate variabilele $X$, astfel că avem aici termeni pe care i-am definit deja în spatiul

probabilistic dat mai devreme. Singura problemă este aceea că avem prea multi astfel de termeni. Numărul de variabile necunoscute poate fi la fel de mare ca N astfel că însumarea să parcurgă peste $O(2^N)$ cazuri.
Solutia acestui aspect este a identifica un subset al acestor variabile care afectează afisajul si de a simplifica expresiile utilizând independenta conditională astfel încât însumarea să cuprindă numai un subset.
Fie *rama* (*Fringe*) numele acelor variabile necunoscute (care nu includ $X_{ij}$) care sunt adiacente pătratelor numerotate. De pildă, dacă $X_{ij}$ se referă la pătratul (2, 1), atunci *rama* contine (2, 2), (3, 2), 4, 2), (4, 1). Ideea este că *numerele* sunt complet determinate numai de *cunoscute*, de *ramă* si de $X_{ij}$ (dacă este adiacent unui număr). Fiind date acestea, *numerele* sunt independente conditional de variabilele necunoscute rămase, pe care le denumim *fundal* (*Background*). În exemplul dat, *fundalul* constă în linia de sus: (1, 3), (2, 3), (3, 3), (4, 3).
Sunt de tratat în fapt două cazuri usor diferite, care depind de adiacenta lui $X_{ij}$ la un număr.
**Cazul 1**. $X_{ij}$ este adiacent unui număr.

$$P(X_{ij} \mid e) = \alpha' \sum_{rama, fundal} P(numere \mid cunoscute, rama, fundal, X_{ij})$$

$$P(cunoscute, rama, fundal, X_{ij}) =$$

$$= \alpha' \sum_{rama, fundal} P(numere \mid cunoscute, rama, X_{ij})$$

$$P(cunoscute, rama, fundal, X_{ij}) =$$

$$= \alpha' \sum_{rama} P(numere \mid cunoscute, rama, X_{ij})$$

$$\sum_{fundal} P(cunoscute, rama, fundal, X_{ij})$$

în prima etapă s-au înlocuit *necunoscutele* cu *rama*, *fundal*, în a doua etapă s-a tinut seamă de independenta conditională si în etapa finală s-a folosit faptul că primul termen nu depinde de *fundal*.
Acum, în expresia ultimă, termenul $P(numere|cunoscute,rama,X_{ij})$ este nul cu exceptia cazului când asignarea notată prin $rama,X_{ij}$ este, în terminologia logicii noastre, un *model* al expresiei CNF implicată prin evidentă. Dacă acea asignare este un model, atunci probabilitatea este 1. Astfel, însumarea pe *rama* se reduce la o însumare mai simplă pe modelele evidentei:

$$P(X_{ij} \mid e) = \alpha' \sum_{\{rama:(rama, X_{ij}) \in \mathrm{mod}els(e)\}} \sum_{fundal} P(cunoscute, rama, fundal, X_{ij})$$

Suma pe variabilele de fundal, care pot avea încă un număr imens de cazuri, poate fi simplificată deoarece termenul probabilistic premergător (prior) $P(cunoscute,rama,fundal,X_{ij})$ este $1/C_{N-1}^{M}$ sau 0, depinzând de faptul dacă $\#(cunoscute,rama,fundal,X_{ij}) = M$. Asadar, avem de numărat numai cazurile în care fundalul are numărul de mine corect. Acesta este dat de $C_B^{M-L}$ cu $B$ dimensiunea fundalului si $L$ numărul $\#(cunoscute,rama,X_{ij})$, adică numărul de mine care nu sunt în fundal. În final se obtine

$$P(X_{ij} \mid e) = \alpha' \sum_{\{rama:(rama, X_{ij}) \in models(e)\}} C_B^{M-L} / C_{N-1}^M = \alpha'' \sum_{\{rama:(rama, X_{ij}) \in models(e)\}} C_B^{M-L} \quad (1)$$

ceea ce este simplu de calculat datorită faptului că se pot enumera modelele pentru variabilele de pe ramă (contur). Costul operatiei este $O(2^{|rama|})$, cam acelasi ca pentru algoritmul logic. De observat că termenul $1/C_{N-1}^M$ dispare în constanta de normalizare deoarece el nu depinde de *ramă*.

Acum aplicăm formula aceasta la calculul probabilitătii $P(X_{2,1}|e)$ din exemplul de mai sus. Mai întâi să enumerăm modelele.

Când $X_{2,1}$ = *true* modelele pe ramă sunt următoarele:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | ? | ? | ? | ? | | 3 | ? | ? | ? | ? | | 3 | ? | ? | ? | ? |
| 2 | m | | m | | | 2 | m | | | m | | 2 | m | | | |
| 1 | 2 | m | 2 | | | 1 | 2 | m | 2 | | | 1 | 2 | m | 2 | m |
| | 1 | 2 | 3 | 4 | | | 1 | 2 | 3 | 4 | | | 1 | 2 | 3 | 4 |

Când $X_{2,1}$ = *false*, modelele pe ramă sunt următoarele:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | ? | ? | ? | ? | | 3 | ? | ? | ? | ? | | 3 | ? | ? | ? | ? |
| 2 | m | m | m | | | 2 | m | m | | m | | 2 | m | m | | |
| 1 | 2 | | 2 | | | 1 | 2 | | 2 | | | 1 | 2 | | 2 | m |
| | 1 | 2 | 3 | 4 | | | 1 | 2 | 3 | 4 | | | 1 | 2 | 3 | 4 |

Pentru fiecare din aceste modele, $C_B^{M-L} = C_4^1 = 4$, astfel că

$$P(X_{2,1}|e) = \alpha''(3\text{x}4, 3\text{x}4) = (1/2, 1/2)$$

Aceste valori sunt în concordantă cu argumentul intuitiv care spune că există exact o mină în (2, 1) sau (2, 2) si că este egal probabil să fie într-una sau în alta din cele două pozitii. Se poate arăta similar că $P(X_{3,2}|e) = (1/3, 2/3)$.

**Cazul 2**: $X_{ij}$ nu este adiacent unui număr.

Mersul calculelor este similar dar $X_{ij}$ apare ca o variabilă de fundal si nu ca o variabilă de ramă:

$$P(X_{ij} \mid e) = \alpha \sum_{rama, fundal} P(numere \mid cunoscute, rama, fundal, X_{ij})$$

$$P(cunoscute, rama, fundal, X_{ij}) =$$

$$= \alpha \sum_{rama, fundal} P(numere \mid cunoscute, rama) P(cunoscute, rama, fundal, X_{ij}) =$$

$$= \alpha \sum_{rama} P(numere \mid cunoscute, rama) \sum_{fundal} P(cunoscute, rama, fundal, X_{ij})$$

Acum variabilele de ramă constituie întregul model:

$$P(X_{ij} \mid e) = \alpha \sum_{rama \in models(e)} \sum_{fundal} P(cunoscute, rama, fundal, X_{ij})$$

si expresia finală este foarte asemănătoare.

$$P(X_{ij} \mid e) = \alpha' \sum_{rama \in models(e)} C_B^{M-L} \quad (2)$$

Este limpede că expresia aceasta este aceeasi pentru toti $X_{ij}$ care nu sunt adiacenti unui număr, astfel că trebuie făcută această operatie o singură dată pentru a obtine probabilitatea sigurantei pentru pătratele de fundal. Fiecare pătrat de ramă trebuie evaluat separat (De notat: semnificatiile pentru *rama* si pentru *fundal* sunt diferite pentru cazul 1 si cazul 2. În primul caz, *rama* include toate variabilele adicente numerelor cu exceptia lui $X_{ij}$, *fundal*ul cuprinde toate celelalte variabile. În cazul al doilea, rama include toate variabilele adiacente numerelor, fundalul include pe toate celelalte cu exceptia variailelor $X_{ij}$. Aceasta face expresiile matematice întrucâtva mai simple.).

| Toatal mine: 6; de localizat: 6 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| – | – | – | – | | 0,298 | 0,298 | 0,298 | 0,737 |
| – | 2 | – | – | | 0,298 | –2– | 0,298 | 0,737 |
| – | – | – | – | | 0,298 | 0,105 | 0,105 | 0,895 |
| – | – | 2 | 1 | | 0,737 | 0,895 | –2– | –1– |

Tabelul de mai sus exemplifică o situatie în care inferenta probabilistică distinge între pătrate foarte sigure – (2, 2) si (3, 2) – si pătrate foarte nesigure – (2, 1) si (4, 2). În esentă, împărtirea unei mine între două numere "2" ar însemna fortarea a 3 mine în cele trei pătrate ale fundalului (1, 1), (4, 3) si (4, 4), ceea ce se poate întâmpla într-un singur mod si de aceea este improbabil.
Să calculăm de pildă $P(X_{1,4}|e)$ în exemplul de mai sus. Avem $B = 3$ si $M = 4$.
Dacă $X_{1,4} = true$, $L = 4$, astfel încât $C_B^{M-L} = C_3^0 = 1$
Dacă $X_{1,4} = false$, $L = 3$, astfel încât $C_B^{M-L} = C_3^1 = 3$
Sunt 6 modele ale ramei, asa încât

$$P(X_{1,4}|e) = \alpha'(6\times1, 6\times3) = (1/4, 3/4)$$

Aceasta arată că fiecare pătrat de fundal are o probabilitatea de 1/4 de a contine o mină. Astfel, miscarea cea mai sigură este într-un pătrat din fundal.
Putem verifica teorema demonstrată în sectiunea referitoare la liniaritatea mediei: suma probabilităților este egală cu numărul de mine rămas. Suma probabilităților este

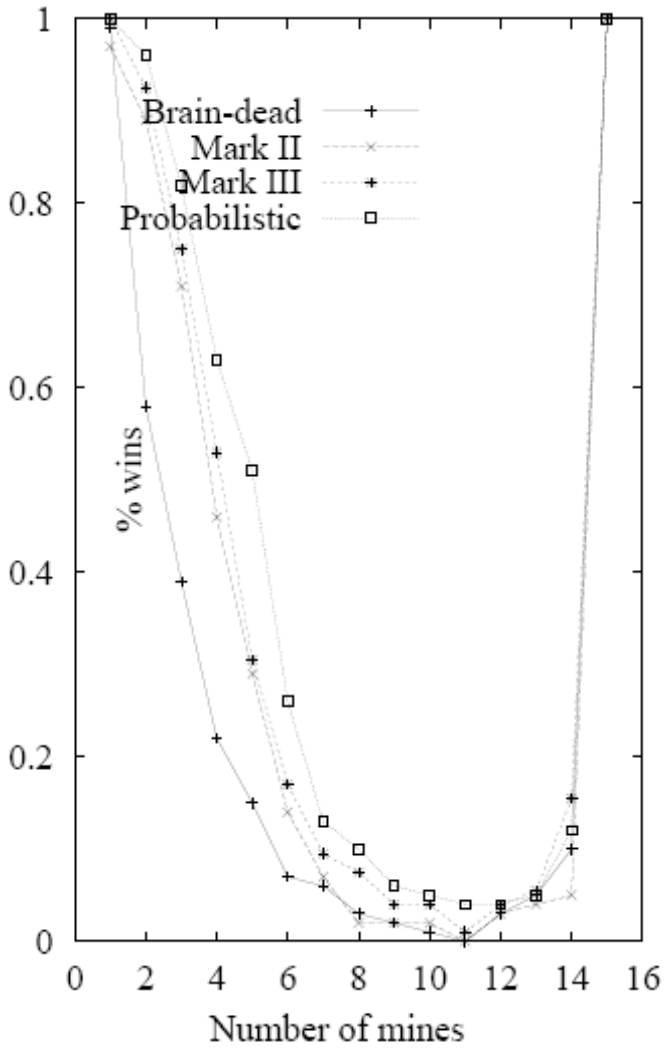
$$1/2 + 1/2 + 1/3 + 1/3 + 1/3 + 1/4 + 1/4 + 1/4 + 1/4 = 3$$

Putem verifica acea lucrare prin calcularea probabilitătii unei mine după prima mutare (înainte de a afla numărul din pătrat). În acel caz, toate pătratele care au rămas dincolo de $X_{ij}$ si pătratul initial sunt pătrate din fundal ($B = N - 2$) si $L = 1$ dacă $X_{ij} = true$ si 0 când $X_{ij} = false$. Nu există variabile în ramă astfel că există exact un model (vid). Asadar

$$P(X_{ij} \mid e) = \alpha \left\langle C_{N-2}^{M-1}, C_{N-2}^{M} \right\rangle = \frac{C_{N-2}^{M-1}}{C_{N-2}^{M-1} + C_{N-2}^{M}} = \frac{C_{N-2}^{M-1}}{C_{N-1}^{M}} =$$

$$= \frac{(N-2)!}{(M-1)!(N-M-1)!} \frac{M!(N-M-1)!}{(N-1)!} = \frac{M}{N-1}$$

uzând la un moment dat de identitatea lui Pascal: $C_{n+1}^{k} = C_{n}^{k-1} + C_{n}^{k}$, ceea ce este în cord cu calculele de mai devreme.
Tabelul de mai devreme arată un caz în care probabilitătile ajută realmente o decizie potrivită. Figura care urmează arată performanta tuturor celor patru metode pentru Minesweeper. Orice îmbunătătire a algoritmului aduce o îmbunătătire în joc la orice densitatea a minelor, dar îmbunătătirea este mai evidentă în cazurile mai dificile.




Performantele medii pe circa 100 de încercări pentru algoritmii brain-dead, Mark II, Mark III si probabilistic pe o tablă 4x4