Teoretyczne podstawy informatyki

Wykład 3b:
Kombinatoryka i prawdopodobieństwo

http://hibiscus.if.uj.edu.pl/~erichter/Dydaktyka2009/TPI-2009

Kombinatoryka i prawdopodobieństwo

- Często spotykamy się z problemem obliczenia wartości wyrażającej prawdopodobieństwo zajścia określonych zdarzeń.
- Dziedzina matematyki zajmująca się tą tematyką to **kombinatoryka**.
- Pojęcia związane z próbami szacowania prawdopodobieństwa występowania zdarzeń definiuje **teoria prawdopodobieństwa**.

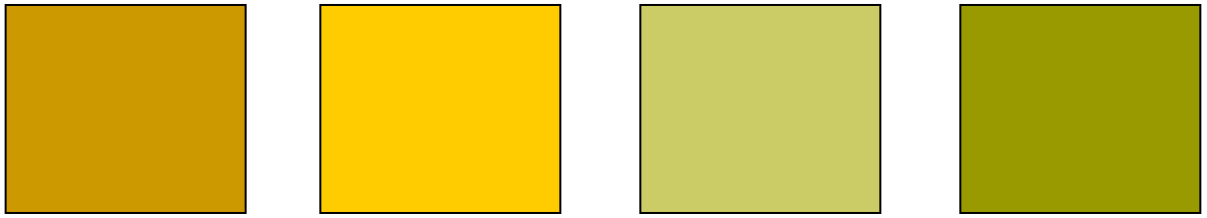
- Zacznijmy od kombinatoryki...

Wariacje z powtórzeniami

- Jednym z najprostszych, ale też najważniejszych problemów jest analiza listy elementów, z których każdemu należy przypisać jedną z wartości należących do stałego zbioru.
- Należy określić możliwą liczbę różnych przyporządkowań (*wariacji z powtórzeniami*) wartości do elementów.
- Przykład:

  4 kwadraty, każdy można pokolorować jednym z 3 kolorów.

  Ile możliwych pokolorowań? $3 \bullet 3 \bullet 3 \bullet 3 = 34 = 81$



Wariacje z powtórzeniami

- Mamy listę **n**-elementów. Istnieje zbiór **k**-wartości z których każda może być przyporządkowana do jakiegoś elementu. Przyporządkowanie jest listą **n** wartości (n_1, n_2,...n_n). Gdzie każda z n_1, n_2,...n_n jest jedną z wartości **k**.
- Istnieje k^n różnych przyporządkowań.
- **Twierdzenie:**
  S(n): liczba możliwych sposobów przyporządkowania dowolnej z k wartości do każdego z n elementów wynosi k^n.

Wariacje z powtórzeniami

- **Podstawa:**

  Przypadek podstawowy to **n=1**. Jeżeli mamy **1** element możemy wybrać dla niego dowolną spośród **k** wartości. Istnieje wiec **k** różnych przyporządkowań. Ponieważ $k^1=k$, podstawa indukcji jest prawdziwa.

- **Indukcja:**

  Załóżmy że **S(n)** jest prawdziwe i rozważmy **S(n+1)**,określające że istnieje k^{n+1} możliwych przyporządkowań jednej z **k** wartości do każdego z **n+1** elementów.

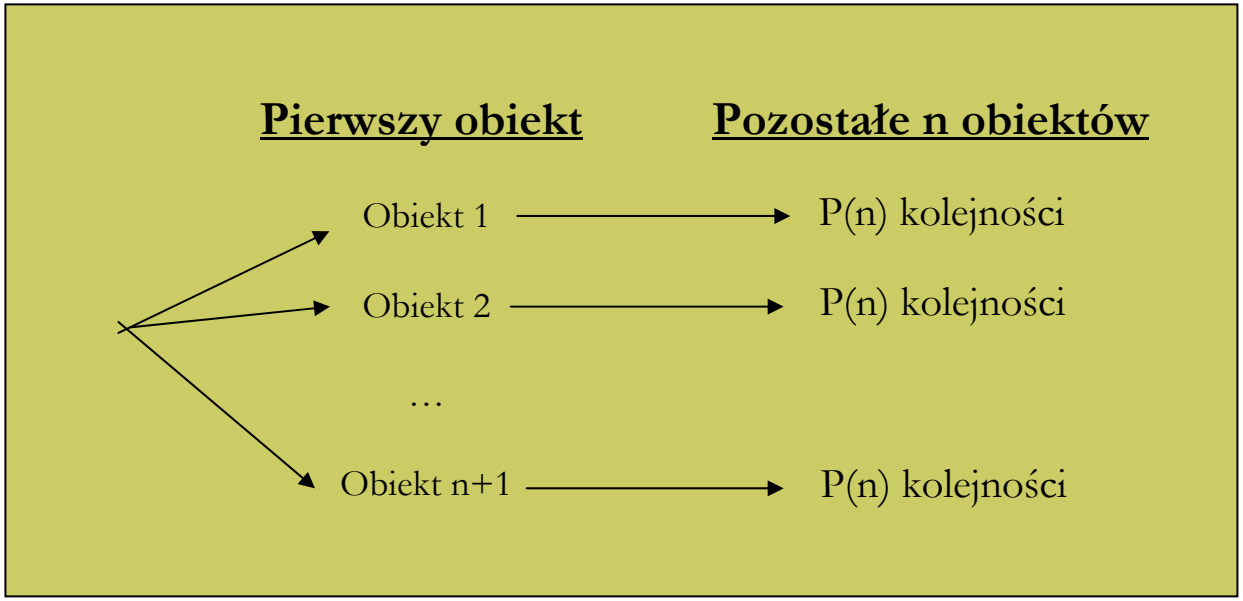
  Wiemy, że istnieje **k** możliwości doboru wartości dla pierwszego elementu. Zgodnie z hipoteza indukcyjną, istnieje k^n przyporządkowań wartości do pozostałych **n** elementów. Łączna liczba przyporządkowań wynosi
  $k \bullet k^n = k^{n+1}$. Cnd.

Permutacje

- Mając **n** różnych obiektów, na ile różnych sposobów można je uporządkować w jednej lini?
- Takie uporządkowanie nazywamy permutacją.
- Liczbę permutacji **n** obiektów zapisujemy jako **P(n)**.

Jak obliczyć P(n+1)?

- Problem: mamy **n+1** obiektów ($a_1, a_2,\ldots,a_n, a_{n+1}$) które mają zostać posortowane.
- Ilość możliwych wyników sortowania jest **P(n+1)**



Permutacje n+1 obiektów

Jak obliczyć P(n+1)?

- **Twierdzenie**:
  P(n) = n! dla wszystkich **n ≥ 1**

- **Podstawa**:
  Dla **n=1**, **P(1)=1** określa że istnieje jedna permutacja dla jednego obiektu.

- **Indukcja**:
  Załóżmy ze **P(n) = n!**
  Wówczas wg. naszego twierdzenia: **P(n+1)=(n+1)!**
  Rozpoczynamy od stwierdzenia że **P(n+1)=(n+1) • P(n)**
  Zgodnie z hipotezą indukcyjną **P(n)=n!**, zatem **P(n+1)=(n+1) • n!**
  Zatem **P(n+1)=(n+1) • n! =(n+1) • n • (n-1) • (n-2) • ... • 1 = (n+1)!**, czyli nasze twierdzenie jest poprawne. Cnd.

- Jednym z interesujących zastosowań wzoru na liczbę permutacji jest dowód na to że **algorytmy sortujące musza działać w czasie co najmniej proporcjonalnym do (n log n), dla n elementów do posortowania**, chyba że wykorzystują jakieś specjalne własności sortowanych elementów.

Wariacje bez powtórzeń

- Niekiedy chcemy wybrać tylko niektóre spośród elementów zbioru i nadać im określony porządek.
- Uogólniamy opisaną poprzednio funkcję P(n) reprezentującą liczbę permutacji, aby otrzymać **dwuargumentową funkcje P(n,m),** którą definiujemy jako ilość możliwych sposobów wybrania **m elementów z n-elementowego zbioru**, przy czym **istotną** rolę odgrywa **kolejność wybierania elementów**, natomiast nieważne jest uporządkowanie elementów nie wybranych.
- Zatem **P(n) = P(n,n).**

Wariacje bez powtórzeń

Przykład:

- Ile istnieje sposobów utworzenia sekwencji **m** liter ze zbioru **n** liter, jeżeli żadna litera nie może występować więcej niż raz?
- Na sam początek możemy zauważyć warunek, by zadanie miało sens: $n \geq m$
- Pierwszą literę możemy wybrać na n sposobów (wybieramy ze zbioru **n**-elementowego), drugą na **n-1** sposobów (gdyż nie możemy wybrać tej samej litery co poprzednio), trzecią na **n-2** sposoby…
- Ostatnią na **n-(m-1)** sposobów.

Twierdzenie: $P(n,m) = n \cdot (n-1) \cdot \ldots \cdot n-(m-1)$ **dla wszystkich** $m \leq n$

Twierdzenie: $P(n,m) = \frac{n!}{(n-m)!}$ **dla wszystkich** $m \leq n$

Kombinacje

- Kombinacja to każdy podzbiór zbioru skończonego. Kombinacją ***m***-elementową zbioru ***n***-elementowego **A** nazywa się każdy ***m***-elementowy podzbiór zbioru **A** (**$0 \leq m \leq n$**). Używa się też terminu "kombinacja z ***n*** elementów po ***m*** elementów" lub wręcz "kombinacja z ***n*** po ***m***".
- Taką funkcję zapisujemy jako: $\binom{n}{m}$

$$\binom{n}{m} = \frac{P(n,m)}{P(m)} = \frac{n!}{(n-m)! \bullet m!}$$

Wyznaczanie liczby kombinacji

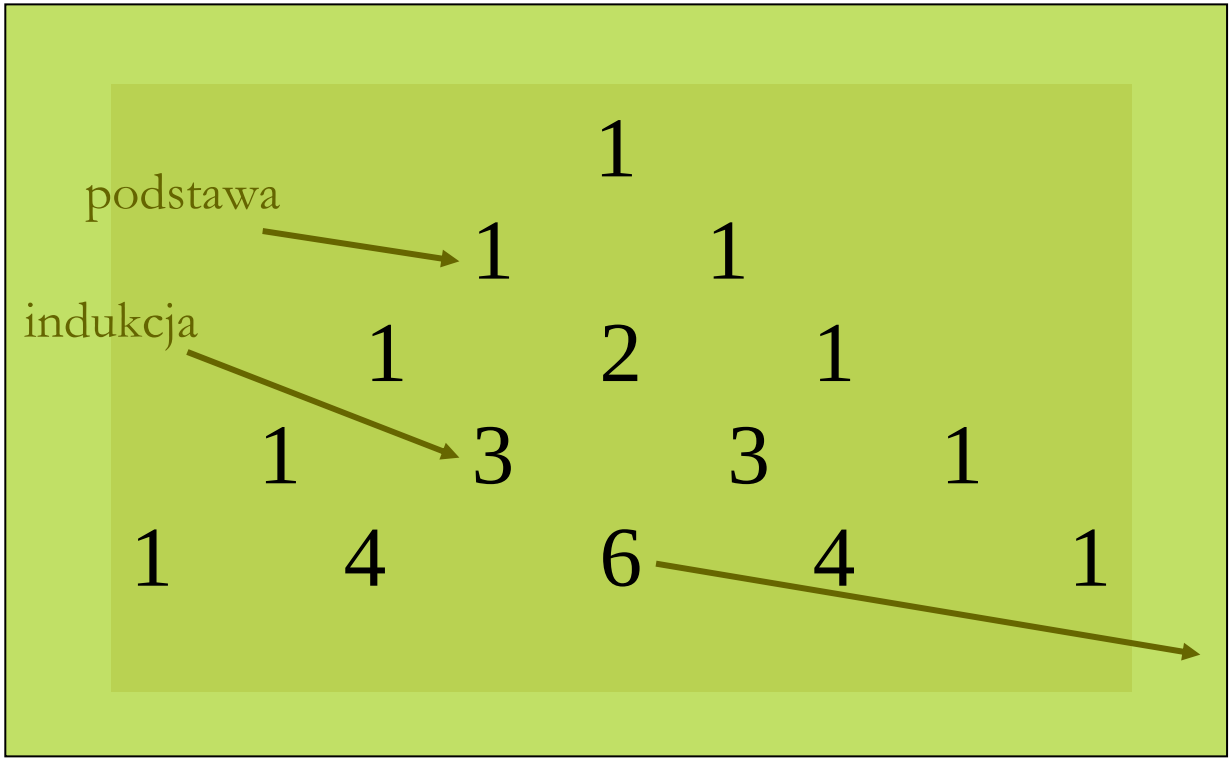
- **Rekurencyjny algorytm: (ilustruje tzw. trójkąt Pascala)**

- **Podstawa:** $\binom{n}{0}=1$ dla dowolnego n≥1.
  Oznacza to że istnieje tylko jeden sposób wybrania **zero** elementów ze zbioru **n**-elementowego – wybranie niczego.
  Także $\binom{n}{n}=1$, ponieważ jedynym sposobem wybrania **n**-elementów ze zbioru **n**-elementowego jest wybranie ich wszystkich.

- **Indukcja:** Jeśli O< m< n, to $\binom{n}{m}=\binom{n-1}{m}+\binom{n-1}{m-1}$.
  Oznacza to, że jeżeli chcemy wybrać **m** elementów ze zbioru **n**-elementowego, możemy albo:
  - nie wybrać pierwszego elementu, po czym wybrać m elementów z pozostałych n-1 elementów. Taką liczbę możliwości wyraża $\binom{n-1}{m}$.
  - wybrać pierwszy element, po czym wybrać m-1 elementów z pozostałych n-1 elementów. Taka liczbę możliwości wyraża $\binom{n-1}{m-1}$.

Trójkąt Pascala

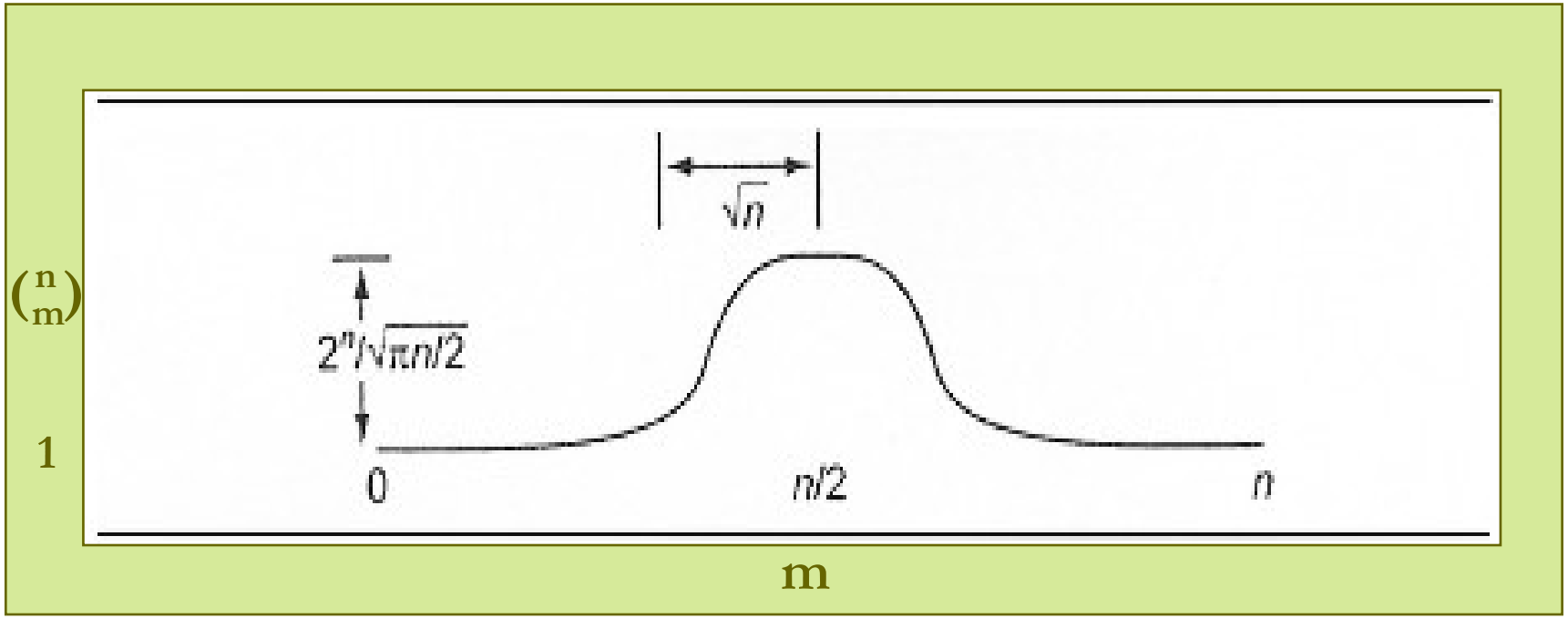
- Rekurencję przy obliczaniu liczby kombinacji często ilustruje się przy pomocy **trójkąta Pascala**.
- $\binom{n}{m}$ = **(m+1)** liczba w **(n+1)** wierszu



$\binom{4}{2} = 4!/(2! \times 2!) = 6$

Interesujące własności funkcji $\binom{n}{m}$

- To również są współczynniki rozkładu dwuwyrazowego wielomianu (dwumianu) $(x+y)^n$
- $\Sigma_{m=0}^{n} \binom{n}{m} = 2^n$
- Wykres funkcji $\binom{n}{m}$ dla stałej dużej wartości **n**:



Permutacje z powtórzeniami

- **Twierdzenie:**
  Jeżeli istnieje **n** elementów podzielonych na **k** grup o rozmiarach równych odpowiednio $i_1, i_2, i_3, \ldots, i_k$ gdzie elementy jednej grupy są identyczne, ale elementy różnych grup różnią się od siebie, liczba uporządkowań tych elementów wynosi
  $S(k) = n! / \prod_{j=1}^{k} i_j!$

- **Podstawa:**
  Dla **k=1**, istnieje tylko jedna grupa zawierająca identyczne elementy, które możemy uporządkować tylko w jeden sposób, niezależnie od liczności tego zbioru. Jeśli **k=1**, to $i_1=n$, zatem **S(1)=n!/n!=1** jest prawdziwe.

- **Indukcja:**
  Załóżmy że **S(k)** jest prawdziwe i rozważmy sytuacje, w której mamy **k+1** grup. Niech ostatnia grupa składa się z $m=i_{k+1}$ elementów, występujących na **m** pozycjach,
  z których możemy je wybierać na $\binom{n}{m}$ sposobów.
  Stosując hipotezę indukcyjną otrzymujemy że $S(k+1)= \binom{n}{m} \cdot (n-m)! / \prod_{j=1}^{k} i_j!$ co łatwo można przekształcić (pamiętając że $m=i_{k+1}$) do postaci:
  $S(k+1)= n! / \prod_{j=1}^{k+1} i_j!$ a więc cnd.

- Jest to typowy problem przy układania anagramów

Łączenie reguł kombinatorycznych

- Typowy problem kombinatoryczny wymaga łączenia przedstawionych reguł (cegiełek) w bardziej skomplikowane struktury.
- Techniki których używamy to:
  - prowadzenie obliczeń jako sekwencji wyborów;
  - prowadzenie obliczeń jako różnicy innych obliczeń (np. wszystkich wyborów – nieprawidłowych wyborów);
  - prowadzenie obliczeń jako sumy rozwiązań dla podprzypadków które są wzajemnie rozłączne.

Teoria prawdopodobieństwa

- Teoria prawdopodobieństwa, szeroko stosowana we współczesnej nauce, ma również wiele zastosowań w informatyce, np.:
  - szacowanie czasu działania programów dla przypadków ze średnimi, czyli typowymi danymi wejściowymi,
  - wykorzystanie do projektowania algorytmów „podejmujących decyzje" w niepewnych sytuacjach, np. najlepsza możliwa diagnoza medyczna na podstawie dostępnej informacji,
  - algorytmy typu Monte Carlo,
  - różnego rodzaju symulatory procesów,
  - **prawie zawsze „prawdziwe" rozwiązania.**

Teoria prawdopodobieństwa

- **Przestrzeń probabilistyczna Ω:**

  Skończony zbiór punktów, z których każdy reprezentuje jeden z możliwych wyników doświadczenia. Każdy punkt **x** jest związany z taką nieujemną liczbą rzeczywistą zwaną prawdopodobieństwem **x**, że suma prawdopodobieństw wszystkich punktów wynosi **1**.
  Istnieje także pojęcie nieskończonych przestrzeni probabilistycznych ale nie mają one większego zastosowania w informatyce.

- **Zdarzenie E:**

  Podzbiór punktów w przestrzeni probabilistycznej.
  Prawdopodobieństwo zdarzenia, **P(E)**, jest sumą prawdopodobieństw punktów należących do tego zdarzenia.

- **Dopełnienie zdarzenia E czyli Ē :**

  Zbiór punktów przestrzeni probabilistycznej które nie należą do zdarzenia **E**.
  P(E) + P(Ē) = 1.

Prawdopodobieństwo warunkowe

- Prawdopodobieństwem warunkowym zajścia zdarzenia **F** pod warunkiem zajścia zdarzenia **E**, gdzie **P(E) > 0** nazywamy liczbę: **P(F|E) = P(E ∩ F) / P (E).**
- Jest to iloraz prawdopodobieństwa części wspólnej zdarzeń **E, F** i prawdopodobieństwa zdarzenia **E**.
- Zdarzenia **E, F** nazywamy niezależnymi jeśli zachodzi: **P(E ∩ F) = P(E) • P(F)**
- W przeciwnym wypadku zdarzenia są zależne.
- Dla zdarzeń niezależnych **E, F** zachodzi: **P(F|E) = P(F)**

Przykłady

- **Zdarzenia niezależne:**
  Rzucamy dwoma kostkami, wyrzucenie liczby „1” na pierwszej kostce (zdarzenie E) nie wpływa na możliwość pojawienia się liczby „1” na drugiej kostce (zdarzenie F). **$P(F|E) = P(F)$**

- **Zdarzenia zależne:**
  Ciągniemy dwa razy kartę z talii kart. Wyciągniecie jako pierwszej karty asa (zdarzenie E), wpływa na możliwość wyciągnięcia jako drugiej karty asa (zdarzenie F). **$P(F|E) \neq P(F)$.**

- W niektórych sytuacjach liczenie prawdopodobieństw jest łatwiejsze jeżeli podzielimy przestrzeń probabilistyczna na rozdzielne obszary **$R_1, R_2, \ldots, R_k$**. Wówczas **$P(E) = \Sigma_{i=0}^{k} P(E|R_i) \bullet P(R_i)$.**

Przykład z kartami

- Ciągniemy dwie karty z talii **52** kart. Liczba możliwych wyników tego doświadczenia (czyli wariacji bez powtórzeń) wynosi **$|\Omega| = 52 * 51 = 2652$**.
- Oznaczmy poprzez **E** zdarzenie polegające na wyciągnieciu jako pierwszej karty As'a.
  $|E| = 4 * 51 = 204.$
  $P(E) = |E| / |\Omega| = 204/2652 = 1/13.$
- Prawdopodobieństwo wyciągnięcia As'a jako drugiej karty, (**zdarzenie F**), jeżeli pierwsza wyciągnięta karta to był As jest **$P(F|E) = P(E \cap F) / P(E) = 4 \bullet 3/204 = 1/17$**.
  $P(E) \bullet P(F|E) = 1/13 \bullet 1/17 = 1/221$.

- Podzielmy przestrzeń na dwa obszary:
  - **R_1** - pierwszą karta jest As, **$|R_1| = 4*51 = 204$**.
  - **R_2** - pierwszą karta nie jest As, **$|R_2| = 2652-204 = 2448$**.
  - **$P(E \cap F)=P((E \cap F)|R1) \bullet P(R1) + P((E \cap F)|R2) \bullet P(R2)$**
  - **$P(E \cap F| R_2) = 0$**
  - **$P(E \cap F| R_1) = 4 \bullet 3/ 4 \bullet 51 = 1/17$**
  - **$P(R_1) = 204/ 2652 = 1/13$**
  - **$P(E \cap F) = P((E \cap F)|R_1) \bullet P(R_1) + P((E \cap F)|R_2) \bullet P(R_2) = 1/17 \bullet 1/13 + 0 \bullet P(R_2) = 1/221.$**

- Gdybyśmy po wyciągnięciu pierwszej karty zwracali ją z powrotem do talii, to mielibyśmy **$P(F|E) = P(F) = 1/13$** (zdarzenia niezależne).
- Wówczas **$P(E) \bullet P(F|E)=1/13 \bullet 1/13=1/169$**.

Reguły związane z wieloma zdarzeniami

Oznaczmy:

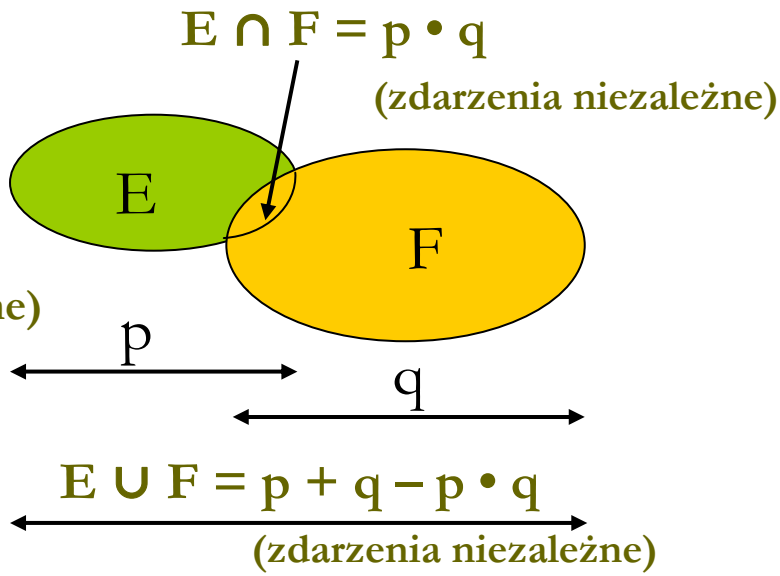
$P(E) = p, P(F) = q.$

Wówczas:

$\max(0, (p+q-1)) \leq P(E \cap F) \leq \min(p,q)$

$P(E \cup F) = p+q - P(E \cap F)$

$P(E \cup F) = p+q - p \bullet q$ (zdarzenia niezależne)



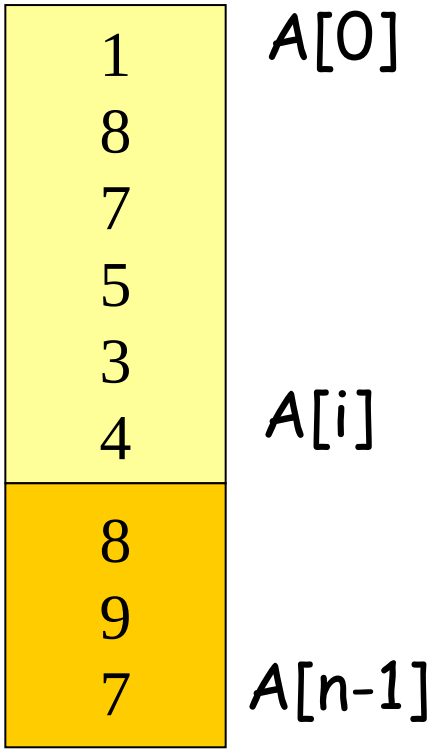
- W zastosowaniach czasem akceptujemy że nie możemy wyznaczyć dokładnie prawdopodobieństw oraz zależności między zdarzeniami. Potrafimy tylko wskazać sytuacje najmniej lub najbardziej prawdopodobne.
- **Zastosowanie: różnego typu diagnostyka**

Oczekiwane wartości obliczeń i analiza probabilistyczna

- Przypuśćmy, że mamy pewną funkcję określoną na przestrzeni probabilistycznej **f(x)**. Wartość oczekiwana tej funkcji po wszystkich punktach przestrzeni $E(f) = \Sigma\, f(x)\, P(x)$.

- Mamy tablicę **n** liczb całkowitych, sprawdzamy czy jakaś liczba całkowita „**x**" jest elementem tej tablicy. Algorytm przegląda całą tablice, po napotkaniu **A[i] = x** kończy działanie.
  - Jeżeli **A[0] = x** to algorytm ***O*(1)**
  - Jeżeli **A[n-1] = x** to algorytm ***O*(n)**

- $E(f) = \sum_{i=0}^{n-1} (c\, i + d) \bullet (1/n) = c \bullet (n-1) / 2 + d$
- $E(f) \sim c \bullet n/2$ dla dużego n



Algorytmy wykorzystujące prawdopodobieństwo

- Jest bardzo wiele różnych typów algorytmów wykorzystujących prawdopodobieństwo.
- Jeden z nich to tzw. **algorytmy Monte-Carlo** które wykorzystują liczby losowe do zwracania albo wyniku pożądanego („prawda"), albo żadnego („nie wiem").
  Wykonując algorytm **stałą** liczbę razy, możemy rozwiązać problem, dochodząc do wniosku, że jeśli żadne z tych powtórzeń nie doprowadziło nas do odpowiedzi „prawda", to odpowiedzią jest „fałsz".
  Odpowiednio dobierając liczbę powtórzeń, możemy dostosować prawdopodobieństwo niepoprawnego wniosku „fałsz" do tak niskiego poziomu, jak w danym przypadku uznamy za konieczne.

- **Nigdy** jednak nie osiągniemy prawdopodobieństwa popełnienia błędu na poziomie **zero**.

Co to są liczby losowe?

- Mówimy, że wyniki pewnych doświadczeń są **losowe,** co oznacza ze wszystkie możliwe wyniki są równie prawdopodobne.
- Przykładowo, jeżeli rzucamy normalną (prawidłową) kostką do gry to zakładamy że nie ma możliwości fizycznego kontrolowania wyniku tego rzutu w taki sposób aby jeden wynik był bardziej prawdopodobny od drugiego.
- Podobnie zakładamy że mając uczciwie potasowana talie kart, nie możemy wpłynąć na wynik - prawdopodobieństwo otrzymania w rozdaniu każdej karty jest identyczne.

Co to są liczby losowe?

- **Wszystkie** generowane przez komputer **losowe** sekwencje są wynikiem działania specjalnego rodzaju algorytmu zwanego **generatorem liczb losowych** (ang. random number generator). Zaprojektowanie takiego algorytmu wymaga specjalistycznej wiedzy matematycznej.
  Przykład prostego generatora który całkiem dobrze sprawdza się w praktyce to tzw. "**liniowy generator kongurencyjny**".
  Wyznaczamy **stałe $a \geq 2$, $b \geq 1$, $x_0 \geq 0$** oraz **współczynnik $m > \max(a, b, x_0)$**.
  Możemy teraz wygenerować sekwencje liczb x_1, x_2, … za pomocą wzoru:

$$x_{n+1} = (a\, x_n + b) \bmod (m)$$

- Dla właściwych wartości stałych a, b, m oraz x_0, sekwencja wynikowa będzie wyglądała na losową, mimo ze została ona wygenerowana przy użyciu konkretnego algorytmu i na podstawie "jądra" x_0.
- **Dla szeregu zastosowań istotna jest odtwarzalność sekwencji liczb losowych.**

Algorytmy wykorzystujące prawdopodobieństwo

- Mamy pudełko w którym jest **n**-procesorów, nie mamy pewności czy zostały przetestowane przez producenta. Zakładamy że prawdopodobieństwo że procesor jest wadliwy (w nieprzetestowanym pudelku) jest 0.10.

- **Co możemy zrobić aby potwierdzić czy pudełko dobre?**
  - przejrzeć wszystkie procesory -> algorytm O(n)
  - losowo wybrać k procesorów do sprawdzenia -> algorytm O(1)
  - błąd polegałby na uznaniu ze pudełko dobre (przetestowane) jeżeli nie było takie.

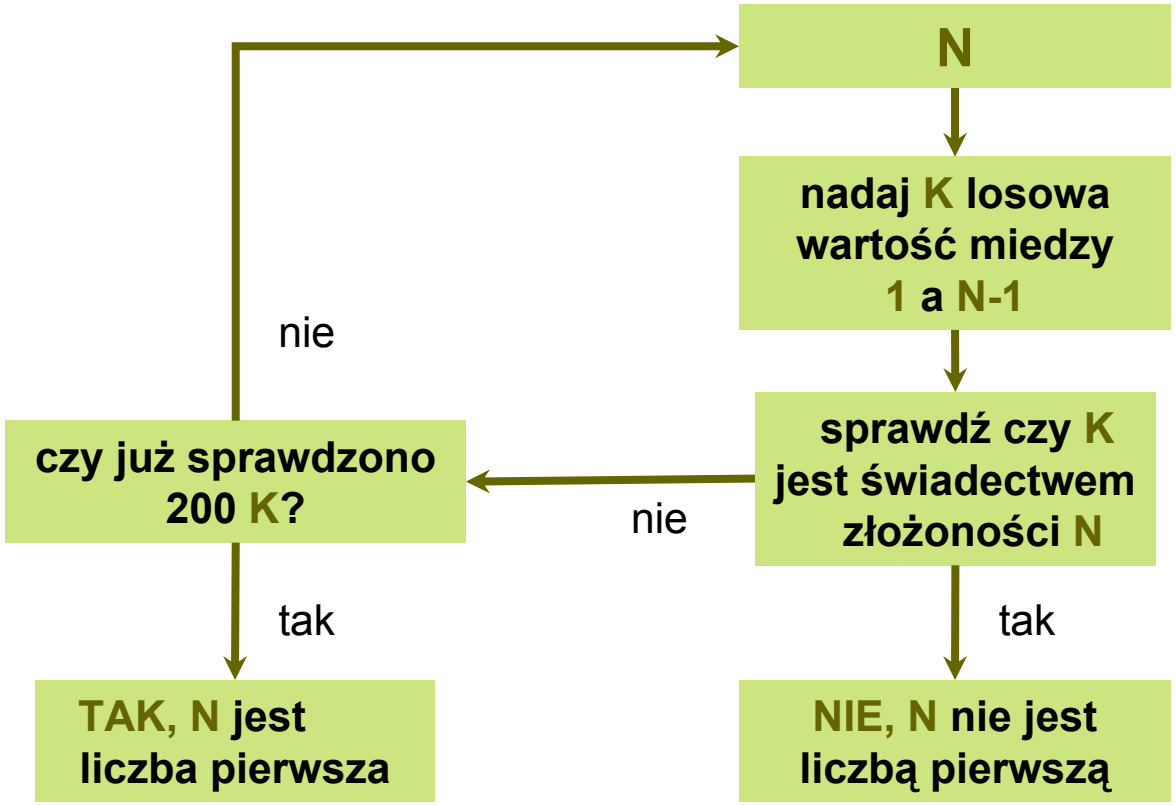
- Losujemy **k=131** procesorów.
  Jeżeli procesor jest dobry odpowiadamy „nie wiem". Prawdopodobieństwo ze „nie wiem" dla każdego z **k**-procesorów $(0.9)^k = (0.9)^{131} = 10^{-6}$.
  10^{-6} to jest prawdopodobieństwo że pudełko uznamy za dobre choć nie było testowane przez producenta.
  Za cenę błędu = 10^{-6}, zamieniliśmy algorytm z $O(n)$ na $O(1)$.
  Możemy regulować wielkość błędu/czas działania algorytmu zmieniając **k**.

Probabilistyczne algorytmy sprawdzania

„ Czy liczba N jest liczbą pierwszą ?”

- W połowie lat 70-tych odkryto **dwa bardzo eleganckie probabilistyczne algorytmy** sprawdzające, czy liczba jest pierwsza. Były one jednymi z pierwszych rozwiązań probabilistycznych dla trudnych problemów algorytmicznych. Wywołały fale badań które doprowadziły do probabilistycznych rozwiązań wielu innych problemów.
- Oba algorytmy wykonują się w czasie wielomianowym (niskiego stopnia), zależnym od liczby cyfr w danej liczbie **N** (czyli ***O* (log N)**).
- Oba algorytmy są oparte na losowym szukaniu pewnych rodzajów potwierdzeń lub **świadectw złożoności** liczby N.
- Po znalezieniu takiego świadectwa algorytm może się bezpiecznie zatrzymać z odpowiedzią **„nie, N nie jest liczbą pierwszą”,** ponieważ istnieje bezdyskusyjny dowód że **N** jest liczbą złożoną.
- Poszukiwanie musi być przeprowadzone w taki sposób aby w pewnym rozsądnym czasie algorytm mógł przerwać szukanie odpowiadając, że **N** jest liczbą pierwszą z bardzo małą szansą omyłki.
- Trzeba zatem znaleźć dająca się **szybko sprawdzać** definicje świadectwa złożoności.

„Czy liczba N jest liczbą pierwszą ?



jeśli to jest odpowiedź to jest to prawda z błędem $1/2^{200}$

jeśli to jest odpowiedź to jest to prawda

Świadectwa złożoności (zarys)

- Każda liczba parzysta poza 2 to jest złożona
- Jeżeli suma cyfr liczby jest podzielna przez 3 to liczba jest złożona (iteracyjny prosty algorytm liniowo zależny od liczby cyfr)
- **Test pierwszości Fermata:**
  - jeśli **n** jest liczbą pierwszą oraz **k** jest dowolna liczba całkowita **(1, n-1)**, to $k^{n-1} \equiv 1 \pmod n$.
  - natomiast jeśli **n** jest liczbą złożoną (z wyjątkiem kilku złych liczb złożonych – liczb Carmichael'a) oraz jeśli **k** wybierzemy losowo z przedziału **(1, n-1)** to prawdopodobieństwo tego że $k^{n-1} \not\equiv 1 \pmod n$ jest mniejsze niż ½.
  - Zatem liczby złożone (poza liczbami Carmichael'a) spełniają warunek testu dla danego **k** z prawdopodobieństwem nie mniejszym niż ½.
- **Test pierwszości Solovay-Strassena:**
  - jeśli **k** i **n** nie maja wspólnych dzielników (co by było świadectwem złożoności) policz:
    $X = k^{(n-1)/2} \pmod n$, **Y = Js(n,k) (symbol Jacobiego),**
    jeśli $X \neq Y$ to **k** jest świadectwem złożoności liczby **n**.
  - dla tego testu nie ma 'złych' liczb złożonych.

Podsumowanie

- Przestrzeń probabilistyczna składa się z punktów z których każdy reprezentuje wynik jakiegoś doświadczenia. Każdy punkt **x** związany jest z nieujemna liczbą zwaną prawdopodobieństwem punktu **x**. Suma prawdopodobieństw wszystkich punktów składających się na przestrzeń probabilistyczna wynosi **1**.
- Zdarzenie jest podzbiorem punktów z przestrzeni probabilistycznej. Prawdopodobieństwo zdarzenia jest sumą prawdopodobieństw należących do niego punktów. Prawdopodobieństwo każdego zdarzenia mieści się w przedziale od **0** do **1**.

Podsumowanie

- Reguła sum określa, że prawdopodobieństwo tego, że zajdzie jedno z dwóch zdarzeń **E** lub **F** jest większe lub równe większemu z prawdopodobieństw obu zdarzeń, ale nie większa niż suma tych prawdopodobieństw.
- Reguła iloczynów określa, ze prawdopodobieństwo tego, że wynikiem pewnego doświadczenia będą dwa zdarzenia **E** i **F**, jest nie większe niż mniejsze z prawdopodobieństw obu zdarzeń.
- Wykonując algorytm **Monte Carlo** stałą liczbę razy, możemy rozwiązać problem, dochodząc do wniosku, że jeśli żadne z tych powtórzeń nie doprowadziło nas do odpowiedzi „prawda”, to odpowiedzią jest „fałsz”.