4b. Teoria liczb: arytmetyka modularna

Grzegorz Kosiorowski

Uniwersytet Ekonomiczny w Krakowie

1. Arytmetyka modularna
2. Kongruencje liniowe i ich układy
3. Chińskie twierdzenie o resztach
4. Funkcja φ Eulera

Arytmetyka modularna - motywacja

Zauważmy, że do modelowania niektórych zjawisk (i opisywania ich algorytmami) zwykła arytmetyka liczb całkowitych nie jest wystarczająca. Dotyczy to szczególnie zjawisk cyklicznych jak np. dzienna rachuba czasu. Na przykład, jeśli zaśniemy o 21 na 9 godzin, to wiemy, że zbudzimy się o 6, a nie o 30 $(21 + 9)$. Analogicznie, Jeśli jest godzina 3 w nocy, to odpowiadając na pytanie, która godzina była 5 godzin temu nie mówimy, że „minus druga" $(3 - 5)$ tylko 22. W tym przykładzie nie interesuje nas właściwy wynik, lecz jego reszta z dzielenia przez 24.
Takie i inne cykliczne zjawiska (np. zmiany pór roku, względne przemieszczenia mechanizmu złożonego z kół zębatych itp.) modeluje arytmetyka modularna.

Przystawanie modulo

Przystawanie modulo

Mówimy, że dwie liczby a i b *przystają do siebie modulo* n, jeśli ich różnica $a - b$ jest wielokrotnością n (lub innymi słowy, jeśli liczby te dają tę samą resztę z dzielenia przez n). Zapisujemy to symbolem $a \equiv b(mod\ n)$ lub $a \equiv_n b$.

Nieprzypadkowo oznaczenie jest tak podobne do oznaczenia reszty z dzielenia. Jeśli $a = b \mod n$, to $a \equiv b(mod\ n)$ (niekoniecznie na odwrót, bo w pierwszym przypadku musi zachodzić $a < n$, a w drugim niekoniecznie - np. $8 \equiv 13(mod\ 5)$, choć resztą z dzielenia 13 przez 5 jest 3).

Przystawanie modulo - równoważność

Równoważność przystawania modulo

Dla dowolnych a, b, c oraz $n > 0$ mamy:
a) $a \equiv_n a$,
b) $a \equiv_n b \Leftrightarrow b \equiv_n a$,
c) jeśli $a \equiv_n b$ i $b \equiv_n c$, to $a \equiv_n c$.

Relacja przystawania modulo n dzieli zbiór liczb całkowitych na rozłączne podzbiory (tak zwane klasy abstrakcji) tak, że każde dwie liczby należące do tego samego podzbioru przystają do siebie modulo n, a dwie liczby z różnych podzbiorów nie przystają do siebie modulo n. Tymi podzbiorami są zbiory liczb dające tę samą resztę z dzielenia przez n - czyli w skrócie reszty z dzielenia przez n.

Przystawanie modulo - własności

Własności przystawania modulo

Dla dowolnych a, b, c, d oraz $n > 0$ mamy:
a) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a + c \equiv_n b + d$,
b) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a - c \equiv_n b - d$,
c) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $ac \equiv_n bd$.

Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17} 15$.

Arytmetyka modularna

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje* czyli działania na klasach abstrakcji relacji modulo (czyli na resztach z dzielenia) tak samo jak na liczbach. Jako, że przynajmniej 3 z działań algebraicznych zachowują się tak samo dla każdego elementu klasy abstrakcji, zamiast używać dowolnych elementów, możemy symbolicznie używać po jednym przedstawicielu każdej klasy (najwygodniej po prostu użyć reszty z dzielenia przez n). Dzięki temu, zamiast pisać, że obliczamy $a + b \equiv_6 ?$, gdzie $a \equiv_6 3$ i $b \equiv_6 5$, możemy w skrócie napisać:
$3 + 5 \equiv_6 2$ (bo dodanie dowolnych liczb z których jedna z dzielenia przez 6 daje resztę 3, a druga 5, zawsze da jako wynik liczbę, która z dzielenia przez 6 da resztę 2). Analogicznie $3 - 5 \equiv_6 4$, $3 \cdot 5 \equiv_6 3$.
Przez $\mathbb{Z}_n$ będziemy oznaczać zbiór reszt z dzielenia przez n z działaniami arytmetycznymi modulo n.

Arytmetyka modularna - przykład $\mathbb{Z}_4$

Oto „tabliczka dodawania” w zbiorze $\mathbb{Z}_4$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

I tabliczka mnożenia w tym samym zbiorze:

$\cdot$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Zauważmy, że o ile łatwo definiować odejmowanie za pomocą dodawania, to dzielenie modulo trzeba wykonywać ostrożnie, bo czasem nie ma sensu (np. $3 : 2$ w $\mathbb{Z}_4$), a czasem może dawać wiele wyników (np. $2 : 2$ w $\mathbb{Z}_4$).

Reguła skracania

Reguła skracania

Dla $n > 0$, jeśli $ad \equiv bd (mod\ n)$ i $d \perp n$, to $a \equiv b (mod\ n)$.

Reguła ta mówi, kiedy możemy legalnie wykonać „dzielenie stronami" w arytmetyce modulo n - otóż poprawność jest gwarantowana tylko wtedy, gdy n i liczba przez którą dzielimy są względnie pierwsze.
Przykładowo, z faktu, że $4 \cdot 4 \equiv_8 2 \cdot 4 (\equiv_8 0)$ nie wynika, że $4 \equiv_8 2$, bo 4 i 8 nie są względnie pierwsze.
Dlatego we wszelkich obliczeniach najlepiej unikać „dzielenia stronami", chyba, że jesteśmy pewni, że działa (tj. sprawdzimy, że liczba przez którą dzielimy i podstawa działania modulo są względnie pierwsze).

Kongruencje liniowe i ich układy

Kongruencją liniową nazywamy przystawanie postaci $ax \equiv_n b$, gdzie $a, b \in \mathbb{Z}$, zaś $x \in \mathbb{Z}_n$ jest niewiadomą. Jeśli kongruencja staje się prawdziwa jest prawdziwa po wstawieniu pewnego $y \in \mathbb{Z}_n$ za x, to y nazywamy *rozwiązaniem* tej kongruencji.
Układem k kongruencji liniowych będziemy nazywać zbiór k przystawań postaci $Ax \equiv_n b$, gdzie A jest macierzą $k \times k$ o współczynnikach z $\mathbb{Z}$, $b \in \mathbb{Z}^k$ jest wektorem złożonym z k liczb całkowitych, zaś $x \in \mathbb{Z}_n^k$ jest wektorem niewiadomych. Jeśli ten układ kongruencji staje się prawdziwy po wstawieniu pewnego $y \in \mathbb{Z}_n^k$ za x, to y nazywamy *rozwiązaniem* tego układu.

Rozwiązywanie kongruencji i ich układów

Kongruencje liniowe i ich układy generalnie rozwiązujemy podobnie jak równości i ich układy na liczbach rzeczywistych. Jest jednak kilka wyjątków:

- Dla kongruencji modulo n używamy, zarówno w wyniku jak i w trakcie obliczeń, tylko liczb naturalnych od 0 do $n-1$. Ostatecznie można dopuścić chwilowe pojawienie się w kongruencji innych liczb całkowitych, lecz powinno się jak najszybciej dodawać lub odejmować od tych liczb odpowiednie wielokrotności n, by wrócić do właściwego przedziału.
- Zgodnie z regułą skracania, kongruencje modulo n można dzielić stronami tylko przez liczby względnie pierwsze z n.

Rozwiązywanie kongruencji i ich układów

Kongruencje i ich układy generalnie rozwiązujemy tak jak równości i ich układy na liczbach rzeczywistych. Jest jednak kilka wyjątków:

- Z reguły skracania ponadto wynika, że podczas rozwiązywania kongruencji nie powinno się również mnożyć stronami przez liczby, które nie są względnie pierwsze z n. Co prawda, rozwiązania oryginalnej kongruencji są rozwiązaniami kongruencji przemnożonej stronami przez dowolną liczbę, ale nie jest na odwrót: pomnożona kongruencja może mieć rozwiązania, które nie były rozwiązaniami kongruencji startowej.

Przykład: $3x \equiv_4 3$. Oczywiście jedynym rozwiązaniem jest $x \equiv_4 1$ (zgodnie z regułą skracania, możemy kongruencję modulo 4 podzielić stronami przez 3). Gdybyśmy pomnożyli kongruencję stronami przez 2 (które nie jest względnie pierwsze z 4), otrzymalibyśmy $6x \equiv_4 6 \Leftrightarrow 2x \equiv_4 2$, co ma dwa rozwiązania $x \equiv_4 1$ i $x \equiv_4 3$. To drugie nie jest rozwiązaniem wyjściowej kongruencji.

Kongruencja liniowa - przykład

Zadanie

Rozwiązać kongruencję

$$7x \equiv_{10} 6$$

Jak widać, operujemy działaniami w zbiorze $\mathbb{Z}_{10}$ - pamiętajmy więc, że mamy do dyspozycji tylko liczby całkowite od 0 do 9 i nie możemy używać ułamków! Aby rozwiązać tę kongruencję, wystarczy podzielić obie strony przez 7. Czy możemy to zrobić? Zgodnie z regułą skracania tak, bo $7 \perp 10$!

Kongruencja liniowa - przykład

Zadanie

$$7x \equiv_{10} 6$$

By wykonać dzielenie 6 przez 7 modulo 10, szukam liczby równoważnej 6 w tej arytmetyce, którą potrafię podzielić przez 7 w arytmetyce liczb całkowitych. Sprawdzam kolejne liczby równoważne 6, otrzymywane przez dodawanie do niej podstawy kongruencji (10): nie może to być 16, 26, 36, 46, bo te liczby, choć równoważne 6, nie dzielą się przez 7. Będzie to np. 56, stąd $7 \cdot 8 \equiv_{10} 6$, więc $6 : 7 \equiv_{10} 8$. Stąd $x \equiv_{10} 8$ jest rozwiązaniem tej kongruencji.

Układ kongruencji liniowych - przykład

Zadanie z egzaminu (2014, zaoczne, I termin)

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 9x - 4y \equiv_{13} 10. \end{cases}$$

Jak widać, operujemy działaniami w zbiorze $\mathbb{Z}_{13}$ - pamiętajmy więc, że mamy do dyspozycji tylko liczby całkowite od 0 do 12 i nie możemy używać ułamków!
Jest wiele sposobów rozwiązania tego zadania. Zobaczymy niekoniecznie najprostszy, ale łatwy do uogólnienia i pokazujący wiele poznanych przed chwilą mechanizmów.

Układ kongruencji liniowych - przykład

Zadanie z egzaminu (2014, zaoczne, I termin)

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 9x - 4y \equiv_{13} 10. \end{cases}$$

Spróbuję w pewnym momencie pozbyć się zmiennej x odejmując kongruencje stronami (operacja dozwolona!). W tym celu chcę drugą kongruencję przekształcić tak, by zamiast $9x$ było w niej $3x$. Najprościej byłoby podzielić obie strony przez 3. Czy to jest dozwolona operacja? Tak, bo 3 jest względnie pierwsze z 13! Jednak jest problem: liczby $\frac{4}{3}$ i $\frac{10}{3}$ nie istnieją w $\mathbb{Z}_{13}$.

Układ kongruencji liniowych - przykład

Zadanie z egzaminu (2014, zaoczne, I termin)

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 9x - 4y \equiv_{13} 10. \end{cases}$$

By wykonać dzielenie 4 przez 3 modulo 13, szukam liczby równoważnej 4 w tej arytmetyce, którą potrafię podzielić przez 3. Będzie to np. 30 (bo $30 = 2 \cdot 13 + 4$), stąd $3 \cdot 10 \equiv_{13} 4$, więc $4 : 3 \equiv_{13} 10$ (ciągle pamiętamy, że $3 \perp 13$, więc dzielenie jest dopuszczalne).
By wykonać dzielenie 10 przez 3 modulo 13, szukam liczby równoważnej 10 w tej arytmetyce, którą potrafię podzielić przez 3. Będzie to np. 36 (bo $36 = 2 \cdot 13 + 10$), stąd $3 \cdot 12 \equiv_{13} 10$, więc $10 : 3 \equiv_{13} 12$ (ciągle pamiętamy, że $3 \perp 13$, więc dzielenie jest dopuszczalne).

Układ kongruencji liniowych - przykład

Zadanie z egzaminu (2014, zaoczne, I termin)

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 9x - 4y \equiv_{13} 10. \end{cases}$$

Zatem powyższy układ kongruencji mogę zapisać następująco:

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 3x - 10y \equiv_{13} 12. \end{cases}$$

Układ kongruencji liniowych - przykład

Zadanie

$$\begin{cases} 3x - 5y \equiv_{13} 1 \\ 3x - 10y \equiv_{13} 12. \end{cases}$$

Teraz odejmę stronami kongruencje (uwzględniając, że $1 - 12 \equiv_{13} 2$), otrzymując

$$5y \equiv_{13} 2.$$

Stąd $y \equiv_{13} 3$ (bo $5 \cdot 3 = 15 \equiv_{13} 2$), a $x \equiv_{13} 1$ (łatwe obliczenie po wstawieniu 3 za y do dowolnej kongruencji). Warto na koniec sprawdzić obliczenia podstawiając wynik do wyjściowych kongruencji.

Istnienie rozwiązań kongruencji liniowych

Tak jak przy rozwiązywaniu równań liniowych i ich układów, powstaje pytanie, czy kongruencje liniowe i ich układy mają zawsze rozwiązanie i czy to rozwiązanie, jeśli istnieje, jest jedyne. I również teraz odpowiedź jest negatywna: kongruencje liniowe i ich układy, tak jak równania i ich układy, mogą nie mieć rozwiązań, mogą mieć jedno rozwiązanie lub (i tu jest różnica w stosunku do układów równań liniowych) mieć ich kilka, ale nie nieskończenie wiele.
Przykładowo, kongruencja $2x \equiv_4 3$ nie posiada rozwiązań, zaś kongruencja $2x \equiv_4 2$ ma dwa rozwiązania.

Istnienie rozwiązań układów kongruencji

Podobnie układ kongruencji:

$$\begin{cases} 3x + y \equiv_{17} 1 \\ x + 6y \equiv_{17} 2 \end{cases}$$

nie ma rozwiązań, a układ:

$$\begin{cases} 3x + y \equiv_{17} 1 \\ x + 6y \equiv_{17} 6 \end{cases}$$

ma aż 17 rozwiązań.

Istnienie rozwiązań układów kongruencji

Dla układów równań liniowych, kwestię istnienia i jednoznaczności rozwiązań rozstrzyga twierdzenie Kroneckera-Capellego. W wypadku kongruencji, odpowiednik byłby nieco bardziej skomplikowany. Dlatego zadowolimy się poniższymi wynikami częściowymi:

Rozwiązalność kongruencji

Kongruencja liniowa $ax \equiv_n b$ ma co najmniej jedno rozwiązanie wtedy i tylko wtedy gdy $NWD(a, n)|b$.

Rozwiązalność układów kongruencji

Jeśli $\det A \perp n$ to układ kongruencji liniowych $Ax \equiv_n b$ ma dokładnie jedno rozwiązanie.

Arytmetyka modularna - zastosowania

Pierwsze zastosowanie dotyczy bardziej pojęcia podzielności, niż arytmetyki modularnej, ale arytmetyka modularna może znacząco pomóc w ulepszeniu tego zastosowania.
Zagadnienie dotyczy sum kontrolnych stosowanych w wielu zagadnieniach, w których łatwo o błędy w odtwarzaniu kodu (numer karty kredytowej, konta bankowego, zapis cyfrowy muzyki itp.).
Powiedzmy, że chcemy przydzielić numery kont kilkudziesięciu tysiącom ludzi. Teoretycznie wystarczy im przypisać numery pięciocyfrowe. Jednakże, próbując wysłać przelew na czyjeś konto, łatwo pomylić jakąś cyfrę i przesłać pieniądze komuś zupełnie innemu. Naprawianie takich pomyłek jest dość kosztowne i czasochłonne, dlatego lepiej je uniemożliwić.

Arytmetyka modularna - zastosowania

Stąd pomysł tzw. „sum kontrolnych", które obecnie pojawiają się w każdym kodzie tego typu. Niektóre cyfry nie są numerem danego obiektu lecz pełnią funkcję badania poprawności całego kodu. Najprostszym przykładem w naszej sytuacji byłoby dodanie do kodu szóstej cyfry, która powstaje w następujący sposób: dodajemy do siebie pięć pozostałych cyfr i reszta z dzielenia tej sumy przez 10 będzie szóstą cyfrą. Na przykład klient o numerze 39605 miałby sumę cyfr równą 23, więc po zbadaniu reszty z dzielenia przez 10 ostateczny numer jego konta to:

$$396053.$$

Takie przypisanie chroni przed błędnym wpisaniem jednej cyfry kodu. Np. jeśli ktoś wziąłby niewyraźnie napisane 0 za kolejną cyfrę 6 i próbował wysłać przelew na konto numer 396653, natychmiast otrzymałby odpowiedź, że taki numer konta jest niemożliwy, gdyż suma cyfr $3, 9, 6, 6, 5$ daje resztę 9, a nie 3.

Arytmetyka modularna - zastosowania

Przedstawiony mechanizm jest bardzo uproszczony w stosunku do stosowanych w praktyce (nie chroni np. przed „czeskim błędem", czyli zapisaniem dwu cyfr w zamienionej kolejności) i w nich arytmetyka modularna i jej prawa przydają się bardziej.

- System International Standard Book Number (ISBN) używa arytmetyki modulo 11 (dla numerów 10-cyfrowych) lub modulo 10 (dla 13-cyfrowych).
- System International Bank Account Number (IBAN) używa arytmetyki modulo 97.
- Chemical Abstract Service (CAS) - system kodowania związków chemicznych - używa arytmetyki modulo 10.
- Kryptograficzne funkcje skrótu (haszujące) - podstawa m.in. kryptowalut.

Arytmetyka modularna - integer overflow

Naturalny sposób przechowywania liczb w pamięci komputera oznacza, że zmienne pozornie całkowitoliczbowe pochodzą tak naprawdę ze zbiorów $\mathbb{Z}_n$. Jest to źródłem błędów programistycznych typu *integer overflow*.
Jeśli w takiej sytuacji zmienna takiego typu zwiększa swoją wartość (np. indeks tablicy), to w pewnym momencie może dojść do *przekręcenia licznika*, który z wartości maksymalnych przeskoczy nagle na minimalne.
Przykłady: pierwszy lot rakiety Ariane 5 (1996), legenda o „atomowym Gandhim".

Chińskie twierdzenie o resztach

Do rozwiązywania układów kongruencji, w których podstawy modulo są różne przyda się następujące twierdzenie:

Chińskie twierdzenie o resztach, Sun Zi, II w.

Niech $n_1, \ldots n_k \in \mathbb{N}$ będą takie, że dla $i \neq j$ zachodzi $n_i \perp n_j$. Wtedy dla dowolnych $a_1, \ldots, a_k$ istnieje dokładnie jedna liczba naturalna $x < n_1 \cdot \ldots \cdot n_k$ taka, że

$$\begin{cases} x \equiv_{n_1} a_1 \\ x \equiv_{n_2} a_2 \\ \quad \ldots \\ x \equiv_{n_k} a_k \end{cases}.$$

Tym razem, na szczęście, twierdzenie jest konstruktywne, czyli istnieje algorytm znajdujący poszukiwane x.

Algorytm chiński

Rozwiązujemy układ kongruencji:

$$\begin{cases} x \equiv_{n_1} a_1 \\ x \equiv_{n_2} a_2 \\ \quad \ldots \\ x \equiv_{n_k} a_k \end{cases}.$$

Algorytm chiński

Dane: Dodatnie liczby całkowite: $n_1, \ldots, n_k$ (parami względnie pierwsze), $a_1, \ldots, a_k$.
Zmienne: N - liczba całkowita, $(N_i)_{i=1}^k$ - skończony ciąg liczb całkowitych, x - wynik (całkowity).

Algorytm chiński

Algorytm chiński

Dane: Ciągi liczby całkowitych dodatnich: $n_1, \ldots, n_k$ (parami względnie pierwsze), $a_1, \ldots, a_k$.
Zmienne: N - liczba całkowita, $(N_i)_{i=1}^k$ - skończony ciąg liczb całkowitych, x - wynik (całkowity).

- I. $N := \prod_{i=1}^k n_i$.
- II. Dla każdego $i \in \{1, \ldots, k\}$ definiujemy $N_i := N/n_i$. Oczywiście $\mathrm{NWD}(n_i, N_i) = 1$.
- III. Dla każdego $i \in \{1, \ldots, k\}$ znajdujemy taką liczbę x_i, że $N_i x_i \equiv_{n_i} 1$ (dzięki temu, że $n_i \perp N_i$).
- IV. $x := \sum_{i=1}^k a_i x_i N_i \mod N$.
- **Rezultat**: x jest szukaną liczbą z chińskiego twierdzenia o resztach.

Komentarz do algorytmu chińskiego

Niejasny może być krok III:
III. Dla każdego i znajdujemy taką liczbę x_i, że $N_i x_i \equiv_{n_i} 1$.
Zazwyczaj w zadaniach odgadnięcie liczby x_i jest proste. W trudniejszych sytuacjach, gdyby było trudno wskazać tę liczbę na pierwszy rzut oka, można zastosować rozszerzony algorytm Euklidesa. Wiemy, że $\mathrm{NWD}(n_i, N_i) = 1$. Zatem istnieją (i potrafimy je znaleźć rozszerzonym algorytmem Euklidesa) x_i, y_i takie, że
$x_i N_i + y_i n_i = NWD(n_i, N_i) = 1$,czyli $N_i x_i = -y_i n_i + 1$, zatem
$N_i x_i \equiv_{n_i} 1$.

Algorytm chiński - przykład

Zadanie

Dowódca gwardii pałacowej chińskiego cesarza przygotowywał swoich żołnierzy do przeglądu. Próbował ich ustawić w trójki, ale zostało 2 nadmiarowych żołnierzy. Następnie ustawiał ich w siódemki, ale zostało 4 nadmiarowych. Na koniec ustawiał ich w ósemki, ale zostało pięciu nadmiarowych. Wiedząc, że żołnierzy gwardii było mniej niż 168, wskazać liczbę tych żołnierzy.

Oczywiście zadanie się sprowadza do rozwiązania układu kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

Algorytm chiński - przykład

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

Faktycznie $3, 7$ i 8 są parami względnie pierwsze. Pierwszy krok algorytmu - obliczamy $N = 3 \cdot 7 \cdot 8 = 168$. Zgodnie z chińskim twierdzeniem o resztach wiemy, że możliwe jest znalezienie $x < 168$ spełniającego warunki zadania.

Algorytm chiński - przykład

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

W drugim kroku obliczamy:

$$N_1 = \frac{N}{n_1} = \frac{168}{3} = 56.$$

$$N_2 = \frac{N}{n_2} = \frac{168}{7} = 24.$$

$$N_3 = \frac{N}{n_3} = \frac{168}{8} = 21.$$

Algorytm chiński - przykład

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

W trzecim kroku obliczamy:

$$56x_1 \equiv_3 1 \rightarrow x_1 = 2.$$

$$24x_2 \equiv_7 1 \rightarrow x_2 = 5.$$

$$21x_3 \equiv_8 1 \rightarrow x_3 = 5.$$

Algorytm chiński - przykład

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

W czwartym kroku obliczamy:

$$x = a_1 x_1 N_1 + a_2 x_2 N_2 + a_3 x_3 N_3 \mod 168 =$$

$$= 2 \cdot 2 \cdot 56 + 4 \cdot 5 \cdot 24 + 5 \cdot 5 \cdot 21 \mod 168 = 1229 \mod 168 = 53.$$

Zatem w gwardii musiało być 53 żołnierzy.

Kongruencje - drugi sposób

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

Można to rozwiązać też w drugi sposób, chyba wygodniejszy w obliczeniach ręcznych, choć mniej efektywny dla dużych układów równań i dużych liczb.
Zauważmy, że ogólne rozwiązanie pierwszej kongruencji to $x = 3i + 2$ dla pewnego i. Wstawiając do drugiej kongruencji dostajemy:

$$3i + 2 \equiv_7 4 \Rightarrow 3i \equiv_7 2 \Rightarrow i \equiv_7 3.$$

Kongruencje - drugi sposób

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

Skoro $i \equiv_7 3$, to $i = 7j + 3$ dla pewnego j. Wstawiając do $x = 3i + 2$ dostajemy $x = 3(7j + 3) + 2 = 21j + 11$ dla pewnego j. Wstawiamy ten rezultat do trzeciej kongruencji i mamy:

$$21j + 11 \equiv_8 5 \Rightarrow 5j + 3 \equiv_8 5 \Rightarrow 5j \equiv_8 2 \Rightarrow j \equiv_8 2.$$

Kongruencje - drugi sposób

Zadanie

Rozwiązać układ kongruencji:

$$\begin{cases} x \equiv_3 2 \\ x \equiv_7 4 \\ x \equiv_8 5 \end{cases}$$

$j \equiv_8 2$, czyli $j = 8k + 2$ dla pewnego k. Zatem $x = 21j + 11$ zmienia się w $x = 21(8k + 2) + 11 = 168k + 53$ dla pewnego k. Ze względu na warunki zadania, $x = 53$.

Algorytm chiński - zastosowania

- Protokoły kryptograficzne typu „dzielenia sekretu” (algorytmy Mignotte i Asmutha-Blooma): pewna „sekretna wiadomość” jest szyfrowana i dzielona na fragmenty, rozdawane jego posiadaczom. Każdemu fragmentowi odpowiada znajomość jednej z kongruencji układu. Pełen „sekret” może być odtworzony tylko gdy wszyscy (lub odpowiednio wielu) uczestników dostarczy swoją informację.
- Protokoły Szybkiej Transformaty Fouriera (analiza sygnałów).
- Zagadnienia *range ambiguity resolution*, czyli wyznaczania poprawnej odległości obiektów wykrywanych przez radary.

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$. Liczby naturalne dodatnie nie większe od niej i względnie pierwsze z nią to 1 i 5, więc $\varphi(6) = 2$.
Oczywiście, nie jest to zbyt wygodny sposób obliczania wartości funkcji φ.

Funkcja Eulera - twierdzenia

Funkcja φ Eulera dla liczb pierwszych

Dla dowolnej liczby pierwszej p zachodzą związki:
a) $\varphi(p) = p - 1$
b) $\varphi(p^k) = p^k(1 - \frac{1}{p})$.

Funkcja φ Eulera dla iloczynów liczb względnie pierwszych

Dla dowolnych dwóch dodatnich liczb względnie pierwszych m i n zachodzi:
$\varphi(mn) = \varphi(m)\varphi(n)$.

Wniosek - potrafimy policzyć wartość funkcji Eulera dla każdej liczby, której rozkład na czynniki pierwsze znamy.

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$. Dodatkowo, oczywiście 2^3, 3 i 5^2 są parami względnie pierwsze, więc możemy skorzystać z obu twierdzeń z poprzedniego slajdu, otrzymując:

$$\varphi(600) = \varphi(2^3 \cdot 3 \cdot 5^2) = \varphi(2^3) \cdot \varphi(3) \cdot \varphi(5^2) = 2^3(1-\frac{1}{2}) \cdot 2 \cdot 5^2(1-\frac{1}{5}) = 160.$$

Obliczanie funkcji Eulera - ważne uwagi

Wydaje się, że z łatwością potrafimy w takim razie obliczyć funkcję Eulera dla każdej liczby. Jednak, trzeba tu podkreślić, że to jest faktycznie łatwe tylko dla liczb, które potrafimy rozłożyć na czynniki pierwsze. A to, jak już wiemy, jest niezwykle trudne w ogólnym przypadku.
Niestety (a może na szczęście, dla bezpieczeństwa różnych systemów) nie jest znany prostszy (i mniej złożony obliczeniowo) sposób obliczania funkcji Eulera niż rozkład liczby na czynniki pierwsze i skorzystanie z twierdzeń. Dlatego można powiedzieć, że obliczenie wartości funkcji Eulera jest równie trudne (i złożone obliczeniowo) jak rozkład liczb na czynniki pierwsze.

Twierdzenie Eulera i Małe Twierdzenie Fermata

Twierdzenie Eulera

Jeśli $a \perp n$, to $a^{\varphi(n)} \equiv_n 1$.

Prostym wnioskiem z twierdzenia Eulera jest:

Małe Twierdzenie Fermata

Dla dowolnej liczby pierwszej p i dodatniego n zachodzi $n^p \equiv_p n$.

Obydwa te twierdzenia są szalenie użyteczne w kryptologii, gdyż ułatwiają arytmetykę modularną na dużych liczbach.

Twierdzenie Eulera - przykład

Zadanie

Obliczyć $19^{74} \mod 28$

19^{74} jest liczbą gigantyczną i liczenie tego „na rympał” bez pomocy sztuczek (nawet przy pomocy komputera) jest koszmarne.

Twierdzenie Eulera

Jeśli $a \perp n$ to $a^{\varphi(n)} \equiv_n 1$.

Oczywiście $19 \perp 28$ (bo 19 jest liczbą pierwszą), więc możemy skorzystać z twierdzenia Eulera. $\varphi(28) = \varphi(2^2 \cdot 7) = 2 \cdot 6 = 12$. Stąd wiemy, że $19^{12} \equiv_{28} 1$.
$19^{74} = (19^{12})^6 \cdot 19^2$, czyli
$19^{74} \equiv_{28} (19^{12})^6 \cdot 19^2 \equiv_{28} (1)^6 \cdot 361 \equiv_{28} 361 \equiv_{28} 25$.