Algorytm rozwiązywania układów równań liniowych nad pierścieniem wielomianów

28 marca 2018

J. Brojacz, U. Dobrowolska, K. Jastrzębski, P. Rudnicki, O. Słowik

1 Wstęp

W artykule tym przedstawiono algorytm rozwiązywania układów równań liniowych postaci

$$Ax = y$$

gdzie $A \in M_{n\times n}(R)$ jest (znaną) macierzą wymiaru $n \times n$ o elementach z pierścienia wielomianów v zmiennych całkowitych $R = \mathbb{Z}[X_1, X_2, ..., X_v]$, $y \in F^n$ jest (znanym) wektorem o elementach z tego pierścienia, natomiast $x \in F^n$ jest szukanym wektorem. Zakładamy dodatkowo, że wielomiany będące elementami x są *rzadkie* (por. Def 4.1).
Algorytm ten oparty jest o metodę Zippela [1] i w konsekwencji jest algorytmem probabilistycznym i, po niewielkiej modyfikacji, modularnym.

2 Lemat Zippela-Schwartza

Lemat 2.1 (Lemat Zippela-Schwartza) *Niech $p(x_1, \ldots, x_n)$ będzie niezerowym wielomianem n zmiennych stopnia d nad ciałem F. Niech S będzie skończonym podzbiorem F oraz niech $r_1, \ldots, r_n$ będą losowymi elementami z S, wtedy:*

$$\Pr[p(r_1, \ldots, r_n) = 0] \leq \frac{d}{|S|}$$

.

Dowód:
Dowód lematu przeprowadzimy przez indukcję względem n. Dla $n = 1$ wielomian p może mieć co najwyżej d zer, co daje podstawę indukcji. Załóżmy teraz, że lemat ten zachodzi dla wszystkich wielomianów nad $n - 1$ zmiennymi. Możemy wtedy zapisać p jako wielomian zmiennej x_1: $p(x_1, \ldots, x_n) = \sum_{i=0}^{d} x_i^d p_i(x_2, \ldots, x_n)$.
Ponieważ p jest niezerowy, to istnieje takie i dla którego p_i jest niezerowy. Weźmy największe takie i. Wtedy $\deg p_i \leq d - i$. Wybierzmy teraz losowo elementy $r_2, \ldots, r_n$ z S. Z hipotezy indukcyjnej mamy: $\Pr[p_i(r_2, \ldots, r_n) = 0] \leq \frac{d-i}{|S|}$.

Jeżeli $p_i(r_2, \ldots, r_n) \neq 0$, to wtedy $p(x_1, r_2, \ldots, r_n)$ ma stopień i, a więc

$$\Pr[p(r_1, r_2, \ldots, r_n) = 0 | P_i(r_2, \ldots, r_n) \neq 0] \leq \frac{i}{|S|}.$$

Mamy więc:

$$\Pr[p(r_1, r_2, \ldots, r_n) = 0] =$$

$$= \Pr[P_i(r_2, \ldots, r_n) = 0] \Pr[p(r_1, r_2, \ldots, r_n) = 0 | P_i(r_2, \ldots, r_n) = 0] +$$

$$+ \Pr[P_i(r_2, \ldots, r_n) \neq 0] \Pr[p(r_1, r_2, \ldots, r_n) = 0 | P_i(r_2, \ldots, r_n) \neq 0] \leq$$

$$\leq \Pr[P_i(r_2, \ldots, r_n) = 0] + \Pr[p(r_1, r_2, \ldots, r_n) = 0 | P_i(r_2, \ldots, r_n) \neq 0] \leq \frac{d-i}{|S|} + \frac{i}{|S|} = \frac{d}{|S|}.$$

□

Fakt 2.1 (Chińskie Twierdzenie o Resztach) *Układ kongruencji*

$$x \equiv c_1 \mod m_1,$$

$$x \equiv c_2 \mod m_2,$$

$$\ldots$$

$$x \equiv c_n \mod m_n,$$

gdzie $m_1, \ldots, m_n$ są parami względnie pierwsze, a $c_1, \ldots, c_n$ są dowolnymi liczbami całkowitymi, ma dokładnie jedno rozwiązanie w zbiorze $\{0, 1, \ldots, m_1 m_2 \ldots m_k - 1\}$.

i dołącz rozwiązanie do S otrzymując wielomian $p_j(X_1, \ldots, X_{i-1})$

3 Algorytm D

Algorytm D, służy do budowy wielomianu, który interpoluje zadaną funkcje.

Data: $\{p_1, p_2, \ldots, p_k\}$ - zbiór liczb wymiernych należących do dziedziny funkcji;
$\{m_1, m_2, \ldots, m_k\}$ - zbiór liczb całkowitych ;
Result: $f(x)$ - wielomian taki, że $f(p_i) = m_i dla 1 \leq i \leq k$
$f(x) \leftarrow m_1$;
$q(x) \leftarrow (x - p_1)$;
for $i = 2$ *do* k **do**
  $f(x) \leftarrow f(x) + q(p_i)^{-}1 q(x)(m_i - f(p_i))$;
  $q(x) \leftarrow (x - p_i) q(x)$;
end
zwróć $f(x)$;

4 Algorytm S

Poprawność działania poniższego algorytmu wymaga założenia *rzadkości* wielomianu p_0:

Definicja 4.1 *Wielomian $p(X_1, \ldots, X_v)$ złożony z t jednomianów, w którym stopień każdej ze zmiennych jest ograniczony przez d, jest rzadki, jeżeli*

$$t \ll (d+1)^v$$

Ponadto zakłada się, że punkt początkowy jest *dobry*, tzn:

Definicja 4.2 *Punkt $\boldsymbol{a} = (a_1, a_2, \ldots, a_v)$ jest dobry, jeżeli nie jest miejscem zerowym wielomianu $F = F_1 \ldots F_{v-1}$, gdzie F_i jest iloczynem niezerowych współczynników P traktowanego jako wielomian zmiennych $X_1, \ldots, X_i$*

Definicja 4.3 *Przyjmujemy, że wielomian szkieletowy S ma t symboli, które reprezentują współczynniki jednomianów otrzymanych z S. Zapisujemy je jako $s_1, \ldots, s_t$ gdzie indeks i jest związany z wektorem $(e_i 1, \ldots, e_i v)$. Wtedy definiujemy:*

$$S(a_1, ..., a_v) = s_1 a_1^{e_{11}} ... a_v^{e_{1v}} + s_2 a_2^{e_{21}} ... a_v^{e_{2v}} + ... + s_t a_t^{e_{t1}} ... a_v^{e_{tv}}$$

Zainicjowanie *Algorytmu S* w dobrym punkcie gwarantuje, że szukany wielomian zostanie odtworzony prawidłowo. Ponieważ każdy z wielomianów F_i jest iloczynem co najwyżej t wielomianów stopnia co najwyżej d w każdej ze zmiennych, dostajemy, że wielomian F jest stopnia co najwyżej $d \cdot v \cdot t$. Z *Lematu Schwartza-Zippla* można wykazać, że przy losowaniu punktów ze zbioru K elementowego prawdopodobieństwo wylosowania punktu, który nie jest dobry, jest ograniczone z góry przez $\frac{v^2 d(d+1)^v}{K}$.

Data: $\mathbf{X} = \{X_1, X_2, \ldots, X_v\}$ - zbiór zmiennych;
d - ograniczenie górne stopnia każdej ze zmiennych;
$F(X_1, X_2, \ldots, X_v)$ - funkcja;
$(a_1, a_2, \ldots, a_v)$ - dobry punkt początkowy
Result: $P(X_1, X_2, \ldots, X_v)$ - wielomian p_0 stopnia co najwyżej d w każdej ze zmiennych, spełniający $P(b_1, b_2, \ldots, b_v) = F(b_1, b_2, \ldots, b_v)$ dla wszystkich liczb całkowitych b_i

```
S ← {(0)};
p_0 ← a_0;
for i = 1 do v do
    for j = 1 do d do
        wylosuj r_j;
        L ← pusta lista;
        t ← długość S;
        for k = 1 do t do
            wylosuj (i − 1)-tkę Λ_k;
            dodaj równanie S(Λ_k) = F(Λ_k, r_j, a_{j+1}, ..., a_v) do układu L;
        end
        rozwiąż układ L i dołącz rozwiązanie do S otrzymując wielomian
         p_j(X_1, ..., X_{i−1})
    end
    przekaż do Algorytmu D współczynniki wielomianów p_0, ..., p_d
     odpowiadające jednomianom z S oraz a_i, r_1, ..., r_i;
    połącz otrzymane t wielomianów z S, przypisz tak otrzymany
     wielomian do p_0, a jego wielomian szkieletowy do S;
end
zwróć p_0;
```

Uwaga 4.4 *Przy przejściu przez zewnętrzną pętlę dla $i = 1$ otrzymane równania liniowe mogą być zależne. W takim wypadku należy powtarzać losowanie Λ_k i dodawanie równań do L aż do otrzymania wystarczającej liczby równań.*

5 Algorytm M

Data: $\mathbf{X} = \{X_1, X_2, \ldots, X_v\}$ - zbiór zmiennych;
A - macierz wymiaru $n \times n$ o elementach z R;
y - wektor długości n o elementach z R;
d - ograniczenie górne stopnia każdej ze zmiennych wielomianów z docelowego wektora;
$(a_1, a_2, \ldots, a_v)$ - dobry punkt początkowy;
Result: x - wektor długości n o elementach będących rzadkimi wielomianami z R spełniający równanie $Ax = y$

```
for i = 1 do n do
    przypisz do F(X) i−tą współrzędną z będącego rozwiązaniem układu
     równań A(X)z = y(X);
    przypisz do i−tej współrzędnej x wynik Algorytmu S zastosowanego
     do X, d, F(X) oraz (a_1, a_2, ..., a_v);
end
zwróć x;
```

6 Modularność

Możliwą modyfikacją powyższego algorytmu jest uczynienie z niego algorytmu modularnego, co może przyspieszyć działanie. W tym celu należy zastąpić *Algorytm S* w *Algorytmie M* poniższym *Algorytmem* S_{mod}.

Data: $\mathbf{X} = \{X_1, X_2, \ldots, X_v\}$ - zbiór zmiennych;
d - ograniczenie górne stopnia każdej ze zmiennych;
$F(X_1, X_2, \ldots, X_v)$ - funkcja;
$(a_1, a_2, \ldots, a_v)$ - dobry punkt początkowy;
Result: $P(X_1, X_2, \ldots, X_v)$ - wielomian stopnia co najwyżej d w każdej ze zmiennych, spełniający $P(b_1, b_2, \ldots, b_v) = F(b_1, b_2, \ldots, b_v)$ dla wszystkich liczb całkowitych b_i;
$M \leftarrow 1$;
$i \leftarrow 1$;
repeat
  weź nie wybraną wcześniej liczbę pierwszą m_i;
  $M \leftarrow M \cdot m_i$;
  przypisz do $p^{(i)}$ wynik *Algorytmu S* użytego do wejściowych danych, wykonując wszystkie obliczenia w $\mathbb{Z}_{m_i}$;
  odtwórz ze współczynników wielomianów $p^{(1)}, \ldots, p^{(i)}$ przy użyciu Chińskiego Twierdzenia o Resztach współczynniki wielomianu p_0;
until *otrzymany wielomian jest równy otrzymanemu w poprzednim kroku*;
zwróć p_0;

Literatura

[1] R. Zippel, Probabilistic Algorithms for Sparse Polynomials