ROZPOZNANIE W ASOCJACJACH PONADREGULARNYCH

Waldemar Wietrzykowski

Instytut Inteligencji Stosowanej, 5 listopada 2021

Streszczenie *niniejsza praca przedstawia zagadnienie rozpoznawania ciągów produkcji przy pomocy asocjacji ponadregularnych. Pokazano w niej, że wśród asocjacji regularnych, wchodzących w skład asocjacji ponadregularnej, stuprocentowe rozpoznanie wykazuje ta asocjacja, którą można uzyskać przez konsolidację rozpoznawanego ciągu produkcji. Z kolei przy rozpoznawaniu kilku dowolnych ciągów produkcji przy pomocy asocjacji ponadregularnej uzyskanej z dowolnego innego ciągu produkcji, wśród asocjacji regularnych wchodzących w skład asosjacji ponadregularnej znajduje się asocjacja, która wykazuje najlepsze rozpoznanie danego ciągu.*

Wstęp

Podstawą rozpoznania w asocjacjach regularnych jest pierwsze prawo asocjacji, opisane w pracy „Asocjacje Regularne" stanowiące, że para produkcji *p* i *n*, powiązana ze sobą relacją następstwa $p \rightarrow n$, reprezentowana jest przez asocjację elementarną (jednostkową) *p n*, gdzie *p* jest poprzednikiem asocjacji a *n* jest następnikiem asocjacji.

Oznacza to, że jeżeli para produkcji jest skonsolidowana w pewnej asocjacji do pierwszą produkcję należy poszukiwać w poprzedniku asocjacji, a drugą produkcję należy poszukiwać w następniku tej asocjacji.

Rozpoznanie dwóch następujących po sobie produkcji

O tym, czy dwie produkcje p1 i p2, powiązane relacją następstwa, są skonsolidowane z asocjacją regularną wyrażoną w postaci listy asocjacji list_asoc informuje funkcja:

```
def recognit_unit(list_asoc, p1, p2):
    for asoc in list_asoc:
        if p1 in asoc[0] and p2 in asoc[1]:
            return True
    return False
```

Funkcja ta zwraca True jeżeli produkcje związane są z asocjacją regularną i False jeżeli nie są związane.

Rozpoznanie ciągu produkcji

W przypadku porównywania ciągu produkcji p z asocjacją regularną wyrażoną w postaci listy asocjacji list_asoc porównujemy kolejne pary: (p_0, p_1), (p_1, p_2), ... ,(p_i, p_{i+1}), (p_{i+1}, p_{i+2}), ... ,(p_{k-1}, p_k) z asocjacją regularną. Ilość zgodności zlicza funkcja:

```
def recognit_units(list_asoc, p):
    n = 0
    for i in range(len(p)-1):
        if recognit_unit(list_asoc,p[i],p[i+1]):
            n += 1    # zliczanie uzgodnień
    i += 1
    return n, i, round((n/i)*100)
```

Funkcja ta zwraca ilość zgodności n, ilość wszystkich porównań par produkcji „i" oraz procentowy udział zgodności.

Wyszukanie asocjacji regularnej najlepiej rozpoznającej ciąg produkcji

Poszczególne rozpoznania ciągu produkcji jakie dają asocjacje regularne zawarte w asocjacji ponadregularnej lists_asoc odbywają się w pętli:

```
for list_asoc in lists_asoc:
    n,i,c = recognit_units(list_asoc,p)
    print(f'{n}/{i} ({c} %)')
```

Powyższa pętla dla każdej asocjacji regularnej list_asoc zawartej w asocjacjach ponadregularnych lists_asoc wypisuje ilość zgodnych porównań n, ilość wszystkich porównań „i" oraz procentowy udział zgodności c dla każdej asocjacji regularnej.

Rozpoznanie ciągu produkcji przez asocjację ponadregularną

Do badania rozpoznania użyto następującą asocjację ponadregularną:

```
lists_asoc = [[[{1,2,3,7},{1,2,4,7}],[{0,4,5,6},{0,3,5,6}]],[[{0,1,2,7},{0,1,2,7}]],
              [[{0,5,6,7},{0,5,6,7}]], [[{0,3,5,6},{0,3,5,6}]],[[{1,2,3,4},{1,2,3,4}]]]
```

Zawiera ona asocjacje regularne, jak w tabeli:

Nr	Ciąg produkcji	Operacja na dwóch ciągach binarnych	Asocjacje (obl. consolid Python)
0	3,1,4,6,0,6,5,0,3,2,4,3,1,4,6,6,5,0,6,5,0,6,3,2,1,1,7,7,2,1,4	Suma arytmetyczna (+)	**(1,2,3,7)** (1,2,4,7) **(0,4,5,6)** (0,3,5,6)
1	7,1,0,2,0,2,1,0,7,2,0,7,1,0,2,2,1,0,2,1,0,2,7,2,	Iloczyn logiczny AND	**(0,1,2,7)** (0,1,2,7)

	1,1,7,7,2,1,0		
2	7,5,0,6,0,6,5,0,7,6,0,7,5,0,6,6,5,0,6,5,0,6,7,6,5,5,7,7,6,5,0	Suma logiczna OR	**(0,5,6,7)** (0,5,6,7)
3	3,5,0,6,0,6,5,0,3,6,0,3,5,0,6,6,5,0,6,5,0,6,3,6,5,5,3,3,6,5,0	Alternatywa rozłączna XOR	**(0,3,5,6)** (0,3,5,6)
4a	1,1,4,4,4,4,1,4,1,4,4,1,1,4,4,4,1,4,4,1,4,4,1,4,1,1,1,1,4,1,4	Jednoargumentowa negacja NOT	**(1,4)** (1,4)
4b	3,1,4,2,4,2,1,4,3,2,4,3,1,4,2,2,1,4,2,1,4,2,3,2,1,1,3,3,2,1,4	Dwuargumentowa negacja NOT	**(1,2,3,4)** (1,2,3,4)

Wyniki rozpoznania przez asocjację regularną

Nr	Ciąg produkcji do rozpoznania	Wyniki rozpoznania przez asocjację ponadregularną [0,1,2,3,4b]		
0	3,1,4,6,0,6,5,0,3,2,4,3,1,4,6,6,5,0,6,5,0,6,3,2,1,1,7,7,2,1,4	0	30/30	100 %
		1	6/30	20 %
		2	12/30	40 %
		3	13/30	43 %
		4b	12/30	40 %
1	7,1,0,2,0,2,1,0,7,2,0,7,1,0,2,2,1,0,2,1,0,2,7,2,1,1,7,7,2,1,0	0	15/30	50 %
		1	30/30	100 %
		2	3/30	10 %
		3	0/30	0 %
		4b	7/30	23 %
2	7,5,0,6,0,6,5,0,7,6,0,7,5,0,6,6,5,0,6,5,0,6,7,6,5,5,7,7,6,5,0	0	21/30	70 %
		1	3/30	10 %
		2	30/30	100 %
		3	20/30	67 %
		4b	0/30	0 %
3	3,5,0,6,0,6,5,0,3,6,0,3,5,0,6,6,5,0,6,5,0,6,3,6,5,5,3,3,6,5,0	0	24/30	80 %
		1	0/30	0 %
		2	20/30	67 %
		3	30/30	100 %
		4b	1/30	3 %
4a	1,1,4,4,4,4,1,4,1,4,4,1,1,4,4,4,1,4,4,1,4,4,1,4,1,1,1,1,4,1,4	0	14/30	47 %
		1	5/30	17 %
		2	0/30	0 %
		3	0/30	0 %
		4a	30/30	100 %

4b	3,1,4,2,4,2,1,4,3,2,4,3,1,4,2,2,1,4,2,1,4,2,3,2,1,1,3,3,2,1,4	0	22/30	73 %
		1	7/30	23 %
		2	0/30	0 %
		3	1/30	3 %
		4b	30/30	100 %

Rozpoznanie dowolnego ciągu produkcji przez dowolną asocjację ponadregularną

Utwórzmy dowolny ciąg produkcji złożony z liczb od 0 do 7.

p = (3,5,2,7,3,4,1,0,2,5,1,0,7,3,4,1,5,3,6,4,7,2,1,0,4,5,3,3,6,4,7,6,7,3,4,2,1,1,7,4,5,3,6,5,3,2,4,
6,7,6,6,7,4,2,7,4,3,6,4,1,7,6,7,4,5,7,2,4,1,7,3,2,5,3,6,3,2,7,5,6,4,5,2,1,3,4,2,7,6,3,5,4,6,5,3,2,
3,2,4,6,4,4,6,5,7,7,7,2,2,4,3,5,2,2,4,1,1,6,5,5,7,6,5,5,7,4,4,3,2,4,5,4,2,1,1,4,0,0,3,0,3,1,5,0,3,
4,2,5,4,0,1,1,2,2,3,3,4,4,5,5,2,2,7,7,4,4,2,1,5,5,1,0,0,0,2,3,4,5,6,7,3,8,2,4,5,6,2,3,5,2,4,6,7,5,
6,6,7,2,3,1,0,3,3,5,3,7,1,7,2,0,2,0,3,2,5,3,7,6,7,3,2,5,2,3,7,1,1,2,0,3,0,4,6,2,7,1,4,5,5,7,4,2,8,
3,3,7,6,4,4,0)

Konsolidując szereg p otrzymujemy zestaw asocjacji regularnych:

```
lists_asoc =
[
[[{2, 3}, {4, 5, 7}], [{7}, {3}], [{0, 4, 5}, {1, 2, 7}], [{1}, {0}]],
[[{5, 7}, {2, 3}], [{0, 3, 6}, {3, 4, 6}], [{1, 2, 4}, {0, 1, 5, 7}]],
[[{3, 4, 6, 7}, {3, 4, 6, 7}]],
[[{1, 2, 7}, {1, 4, 7}], [{5}, {3}], [{3, 4, 6}, {2, 5, 6, 7}]],
[[{2, 3, 6, 7}, {4, 6, 7}], [{4}, {1, 2, 3}], [{1}, {7}]],
[[{2, 3, 7}, {2, 3, 4, 6}], [{1, 5, 6}, {7}], [{4}, {1, 5}]],
[[{1, 3, 5, 6}, {2, 3, 4, 6}], [{2}, {1, 7}], [{4, 7}, {2, 5}]],
[[{2}, {3, 7}], [{4, 7}, {6}], [{3, 5, 6}, {2, 3, 4, 5}]],
[[{3, 5, 7}, {2, 7}], [{2, 4, 6}, {2, 4, 5, 6}]],
[[{1, 4, 7}, {1, 3, 4, 6}], [{2, 3, 5, 6}, {2, 4, 5, 7}]],
[[{2, 3, 4, 5}, {2, 3, 4, 5}]],
[[{0, 1, 3, 4}, {0, 1, 3, 4}]],
[[{0, 1}, {1, 2, 3}], [{3, 4, 5}, {0, 2, 4}], [{2}, {5}]],
[[{2, 3, 4, 5}, {2, 3, 4, 5}]],
[[{4, 7}, {2, 4, 7}], [{0, 1, 2, 5}, {0, 1, 2, 5}]],
[[{2, 3, 4, 7}, {3, 4, 5, 8}], [{8, 5, 6}, {2, 6, 7}]],
[[{0, 2, 3, 7}, {1, 2, 3, 5}], [{5, 6}, {3, 6, 7}], [{1}, {0}]],
[[{0, 1, 3, 7}, {1, 2, 3, 7}], [{2}, {0, 5}], [{5}, {3}]],
[[{3, 5, 6}, {2, 7}], [{1, 2, 7}, {1, 3, 5, 6}]],
[[{2, 3, 4, 5}, {0, 5, 6, 7}], [{0, 1, 7}, {1, 3, 4}], [{6}, {2}]],
[[{2}, {8}], [{8, 3}, {3, 7}], [{7}, {6}], [{4, 6}, {0, 4}]]
]
```

Następnie porównujemy otrzymaną asocjację ponadregularną z następującymi ciągami:

p1 = 5,3,6,2,2,7,0,3,1,2
p2 = 6,3,7,7,0,1,0,0,3,4,2,7
p3 = 2,2,4,4,0,4,5,5,7,3,7,7,7,0
p4 = 5,0,3,4,0

Otrzymujemy wyniki porównań jak w tabeli:

Lp	Asocjacje regularne zawarte w asocjacji ponadregularnej [0, ..., 20]	p1	p2	p3	p4
0	(2,3)(4,5,7),(7)(3),(0,4,5)(1,2,7),(1)(0) ;2	1, 9, 11 %	6, 11, 55 %	4, 13, 31 %	1, 4, 25 %
1	(5,7)(2,3),(0,3,6)(3,4,6),(1,2,4)(0,1,5,7) ;2	4, 9, 44 %	5, 11, 45 %	4, 13, 31 %	3, 4, 75 %
2	(3,4,6,7)(3,4,6,7) ;1	1, 9, 11 %	4, 11, 36 %	5, 13, 38 %	1, 4, 25 %
3	(1,2,7)(1,4,7),(5)(3),(3,4,6)(2,5,6,7) ;2	4, 9, 44 %	4, 11, 36 %	5, 13, 38 %	0, 4, 0 %
4	(2,3,6,7)(4,6,7),(4)(1,2,3),(1)(7) ;2	2, 9, 22 %	5, 11, 45 %	4, 13, 31 %	1, 4, 25 %
5	(2,3,7)(2,3,4,6),(1,5,6)(7),(4)(1,5) ;1	2, 9, 22 %	1, 11, 9 %	5, 13, 38 %	1, 4, 25 %
6	(1,3,5,6)(2,3,4,6),(2)(1,7),(4,7)(2,5) ;2	5, 9, 56 %	4, 11, 36 %	1, 13, 8 %	1, 4, 25 %
7	(2)(3,7),(4,7)(6),(3,5,6)(2,3,4,5) ;2	3, 9, 33 %	3, 11, 27 %	1, 13, 8 %	1, 4, 25 %
8	(3,5,7)(2,7),(2,4,6)(2,4,5,6) ;2	2, 9, 22 %	3, 11, 27 %	8, 13, 62 %	0, 4, 0 %
9	(1,4,7)(1,3,4,6),(2,3,5,6)(2,4,5,7) ;2	3, 9, 33 %	3, 11, 27 %	7, 13, 54 %	1, 4, 25 %
10	(2,3,4,5)(2,3,4,5) ;1	2, 9, 22 %	2, 11, 18 %	5, 13, 38 %	1, 4, 25 %
11	(0,1,3,4)(0,1,3,4) ;1	2, 9, 22 %	5, 11, 45 %	3, 13, 23 %	3, 4, 75 %
12	(0,1)(1,2,3),(3,4,5)(0,2,4),(2)(5) ;2	2, 9, 22 %	4, 11, 36 %	2, 13, 15 %	4, 4, 100 %
13	(2,3,4,5)(2,3,4,5) ;1	2, 9, 22 %	2, 11, 18 %	5, 13, 38 %	1, 4, 25 %
14	(4,7)(2,4,7),(0,1,2,5)(0,1,2,5) ;2	2, 9, 22 %	5, 11, 45 %	5, 13, 38 %	1, 4, 25 %
15	(2,3,4,7)(3,4,5,8),(8,5,6)(2,6,7) ;1	1, 9, 11 %	1, 11, 9 %	5, 13, 38 %	1, 4, 25 %
16	(0,2,3,7)(1,2,3,5),(5,6)(3,6,7),(1)(0) ;2	4, 9, 44 %	4, 11, 36 %	3, 13, 23 %	1, 4, 25 %
17	(0,1,3,7)(1,2,3,7),(2)(0,5),(5)(3) ;2	4, 9, 44 %	4, 11, 36 %	4, 13, 31 %	1, 4, 25 %
18	(3,5,6)(2,7),(1,2,7)(1,3,5,6) ;1	1, 9, 11 %	1, 11, 9 %	3, 13, 23 %	0, 4, 0 %
19	(2,3,4,5)(0,5,6,7),(0,1,7)(1,3,4),(6)(2) ;1	4, 9, 44 %	4, 11, 36 %	7, 13, 54 %	3, 4, 75 %
20	(2)(8),(8,3)(3,7),(7)(6),(4,6)(0,4) ;0	0, 9, 0 %	1, 11, 9 %	3, 13, 23 %	1, 4, 25 %

Ostatnia cyfra po średniku oznacza sposób zakończenia generacji danej asocjacji regularnej.

Pełny kod źródłowy ponadregularnej asocjacyjnej maszyny Turinga

```
# Rozpoznanie w asocjacjach ponadregularnych (Pyton 3.9.7)
# Waldemar Wietrzykowski, 2.12.2021

#p = 3,1,4,6,0,6,5,0,3,2,4,3,1,4,6,6,5,0,6,5,0,6,3,2,1,1,7,7,2,1,4
#p = 7,1,0,2,0,2,1,0,7,2,0,7,1,0,2,2,1,0,2,1,0,2,7,2,1,1,7,7,2,1,0
p = 7,5,0,6,0,6,5,0,7,6,0,7,5,0,6,6,5,0,6,5,0,6,7,6,5,5,7,7,6,5,0
#p = 3,5,0,6,0,6,5,0,3,6,0,3,5,0,6,6,5,0,6,5,0,6,3,6,5,5,3,3,6,5,0
#p = 1,1,4,4,4,4,1,4,1,4,4,1,1,4,4,4,1,4,4,1,4,4,1,4,1,1,1,1,4,1,4
#p = 3,1,4,2,4,2,1,4,3,2,4,3,1,4,2,2,1,4,2,1,4,2,3,2,1,1,3,3,2,1,4
```

```
lists_asoc = [[[{1,2,3,7},{1,2,4,7}],[{0,4,5,6},{0,3,5,6}]],[[{0,1,2,7},{0,1,2,7}]],[[{0,5,6,7},{0,5,6,7}]],
[[{0,3,5,6},{0,3,5,6}]],[[{1,2,3,4},{1,2,3,4}]]]

names_asoc = ('suma_+','iloczyn_and','suma_or','alternatywa_xor','negacja_not')

def recognit_asoc(lists_asoc, p):

    def recognit_unit(list_asoc, p1, p2):    # porównaj asocjację regularną z jednostkowym
                                              # następstwem produkcji
        for asoc in list_asoc:
            if p1 in asoc[0] and p2 in asoc[1]:
                return True # zgodne dla poprzednika i następnika
        return False    # niezgodne dla poprzednika i następnika


    def recognit_units_1(list_asoc, p): # porównaj asocjację regularną z ciągiem kolejnych
                                        # następstw produkcji
        for i in range(len(p)-1):         # ciąg następstw produkcji
            if recognit_unit(list_asoc,p[i],p[i+1]):
                pass        # ilość pierwszych kolejnych uzgodnień
            else: # jeżeli w powyższej instrukcji for nie została wykonana żadna
                # instrukcja break
                return False, i
        return True, i

    def recognit_units_2(list_asoc, p): # porównaj asocjacją regularną z ciągiem
                                        # kolejnych następstw produkcji
        n = 0
        for i in range(len(p)-1):             # ciąg następstw produkcji
            if recognit_unit(list_asoc,p[i],p[i+1]):
                n += 1    # ilość wystąpień uzgodnień
        #return n>0, n
        i += 1
        return n,i,round((n/i)*100)

    if type(p) is str:
        p = p.split(',')    # zamiana string na list: '1,2,3' -> ['1','2','3']
    else:
        if type(p) is tuple:
            pass

    for list_asoc in lists_asoc:
        n,i,c = recognit_units_2(list_asoc,p)
        # print(f'{n}/{i} ({c} %)')
        print(f'{n}, {i}, {c} %')
```

################## inne dane ####################

lists_asoc = [[[{2, 3}, {4, 5, 7}], [{7}, {3}], [{0, 4, 5}, {1, 2, 7}], [{1}, {0}]], [[{5, 7}, {2, 3}], [{0, 3, 6}, {3, 4, 6}], [{1, 2, 4}, {0, 1, 5, 7}]], [[{3, 4, 6, 7}, {3, 4, 6, 7}]], [[{1, 2, 7}, {1, 4, 7}], [{5}, {3}], [{3, 4, 6}, {2, 5, 6, 7}]], [[{2, 3, 6, 7}, {4, 6, 7}], [{4}, {1, 2, 3}], [{1}, {7}]], [[{2, 3, 7}, {2, 3, 4, 6}], [{1, 5, 6}, {7}], [{4}, {1, 5}]], [[{1, 3, 5, 6}, {2, 3, 4, 6}], [{2}, {1, 7}], [{4, 7}, {2, 5}]], [[{2}, {3, 7}], [{4, 7}, {6}], [{3, 5, 6}, {2, 3, 4, 5}]], [[{3, 5, 7}, {2, 7}], [{2, 4, 6}, {2, 4, 5, 6}]], [[{1, 4, 7}, {1, 3, 4, 6}], [{2, 3, 5, 6}, {2, 4, 5, 7}]], [[{2, 3, 4, 5}, {2, 3, 4, 5}]], [[{0, 1, 3, 4}, {0, 1, 3, 4}]], [[{0, 1}, {1, 2, 3}], [{3, 4, 5}, {0, 2, 4}], [{2}, {5}]], [[{2, 3, 4, 5}, {2, 3, 4, 5}]], [[{4, 7}, {2, 4, 7}], [{0, 1, 2, 5}, {0, 1, 2, 5}]], [[{2, 3, 4, 7}, {3, 4, 5, 8}], [{8, 5, 6}, {2, 6, 7}]], [[{0, 2, 3, 7}, {1, 2, 3, 5}], [{5, 6}, {3, 6, 7}], [{1}, {0}]], [[{0, 1, 3, 7}, {1, 2, 3, 7}], [{2}, {0, 5}], [{5}, {3}]], [[{3, 5, 6}, {2, 7}], [{1, 2, 7}, {1, 3, 5, 6}]], [[{2, 3, 4, 5}, {0, 5, 6, 7}], [{0, 1, 7}, {1, 3, 4}], [{6}, {2}]], [[{2}, {8}], [{8, 3}, {3, 7}], [{7}, {6}], [{4, 6}, {0, 4}]]]

p = 5,3,6,2,2,7,0,3,1,2

recognit_asoc(lists_asoc, p)

Bibliografia

1. Waldemar Wietrzykowski, *Asocjacje ponadregularne*, IIS 2021
2. Waldemar Wietrzykowski, *Implementacja asocjacyjnej maszyny Turinga*, IIS 2021
3. Waldemar Wietrzykowski, *Maszyna asocjacyjna*, IIS 2021
4. Waldemar Wietrzykowski, *Asocjacje regularne*, IIS 2021
5. Waldemar Wietrzykowski, *Subiektywne badanie pamięci*, IIS 2020
6. Waldemar Wietrzykowski, *Konsolidacja elementarna*, IIS 2019
7. Waldemar Wietrzykowski, *Naturalny System Operacyjny*, IIS 2019
8. Waldemar Wietrzykowski, *Dwie maszyny*, IIS 2018
9. Waldemar Wietrzykowski, *Uczenie języka naturalnego*, IIS 2018
10. Waldemar Wietrzykowski, *Implementacja rachunku sekwentowego*, DIL 2018
11. Waldemar Wietrzykowski, *Wystarczalność rachunku sekwentowego. Rachunek tautologiczny*, DIL 2018
12. Waldemar Wietrzykowski, *Teoria wiedzy*, DIL 2018
13. Waldemar Wietrzykowski, *Teoria wiedzy teoretycznej*, DIL 2018
14. Waldemar Wietrzykowski, *Teoria wiedzy praktycznej*, DIL 2018
15. Waldemar Wietrzykowski, *Teoria świadomej maszyny*, DIL 2018
16. Waldemar Wietrzykowski, *Założenia wstępne do projektu świadomej maszyny*, DIL 2017
17. Waldemar Wietrzykowski, *Świadoma maszyna*, DIL 2017
18. Waldemar Wietrzykowski, *Świadoma reakcja maszyn*, DIL 2017
19. Waldemar Wietrzykowski, *Samoucząca się maszyna motoryczna*, DIL 2017
20. Waldemar Wietrzykowski, *Strumień świadomości*, DIL 2017
21. Waldemar Wietrzykowski, S*calanie interpretacji maszyn*, DIL 2017
22. Waldemar Wietrzykowski, *Obraz rzeczywistości z poziomu maszyny*, DIL 2017
23. Waldemar Wietrzykowski, *Świadomość wielu maszyn*, DIL 2017
24. Waldemar Wietrzykowski, *Wiele maszyn*, DIL 2017
25. Waldemar Wietrzykowski, *Znaczenie interpretacji maszyn*, DIL 2017
26. Waldemar Wietrzykowski, *Interpretacje maszyn*, DIL 2016

27. Waldemar Wietrzykowski, *Jednofunkcyjne maszyny*, DIL 2016
28. Waldemar Wietrzykowski, *Samoucząca się maszyna*, DIL 2016
29. Waldemar Wietrzykowski, *Procesor samouczący się*, DIL 2016
30. Waldemar Wietrzykowski, *Processor self-learning*, DIL 2016
