Algorytmy optymalizacji dyskretnej 2026 / 27

LISTA 3

Zadania na tej liście zaczerpnięte są z podręczników [AMO93, BHM77] oraz ze strony [Bea]. Warto też zajrzeć na stronę [Sou12] (notatki do wykładu – część 4. *Knapsack* oraz część 8. *Bin Packing*).

Zadanie 1. Kierownictwo firmy rozważa cztery projekty możliwe do realizacji w trzyletnim okresie, których charakterystykę przedstawia poniższa tabela (wszystkie kwoty podane są w milionach \$; $-$ oznacza, że projekt nie jest realizowany w danym roku).

Projekt	Zysk	Wymagany kapitał		
		Rok 1	2	3
A	0,2	0,5	0,3	0,2
B	0,3	$-$	0,8	0,2
C	0,5	1,5	1,5	0,3
D	0,1	0,1	0,4	$-$
Dostępny kapitał		3,1	2,5	0,4

Dodatkowo mamy następujące ograniczenia:

- projekty A i B nie mogą być realizowane jednocześnie,
- jeśli wybrany zostanie projekt D, to cały zysk z jego realizacji musi być dodany do kapitału dostępnego w roku 3 (i nie wlicza się do końcowego zysku).

Celem jest wybór projektów do realizacji, który maksymalizuje końcowy zysk, nie przekracza dostępnego kapitału w kolejnych latach i uwzględnia wszystkie dodatkowe ograniczenia.

(a) Sformułuj powyższy problem jako zagadnienie ILP. Wyznacz optymalny wybór projektów (np. korzystając z wybranego solvera).

(b) Załóżmy, że kierownictwo może zdecydować, jaka część zysku z realizacji projektu D zostanie dodana do kapitału dostępnego w roku 3, a jaka wliczona do końcowego zysku. Zmodyfikuj model z podpunktu (a) tak, aby uwzględniał tą możliwość.

Zadanie 2. W pewnej grze mamy płytki z małymi literami alfabetu. Dostępne jest N_α płytek z literą $\alpha \in \{a, b, \ldots, z\}$. Możemy z nich ułożyć dowolne ze słów $w_1, \ldots, w_n$ (to mogą być np. wszystkie poprawne słowa w danym języku lub słowa z pewnego słownika). Każda płytka może być użyta co najwyżej raz i każde słowo możemy ułożyć co najwyżej raz. Za ułożenie słowa w_j otrzymujemy $v_j \geqslant 0$ punktów oraz dodatkowy bonus $b_{ij} \geqslant 0$ za ułożenie obu słów w_i oraz w_j, $i, j \in \{1, \ldots, n\}$, $i \neq j$. Naszym celem jest uzyskanie jak największej liczby punktów przy użyciu dostępnych płytek.

(a) Sformułuj powyższy problem optymalnego wyboru układanych słów jako zagadnienie ILP.

(b) Jak zmieni się model z podpunktu (a), jeśli na początku musimy wybrać 100 spośród wszystkich dostępnych płytek (w dowolny sposób), których możemy użyć do układania słów?

Zadanie 3. Trzy różne elementy obrabiane są kolejno na trzech maszynach. Każdy element musi być najpierw przetworzony na maszynie 1, następnie na maszynie 2, a na końcu na maszynie 3. Kolejność, w której poszczególne elementy są przetwarzane, może być różna dla każdej z maszyn. Załóżmy, że znane są czasy t_{ij} wymagane do przetworzenia elementu i na maszynie j dla $i, j \in \{1, 2, 3\}$. Celem jest zminimalizowanie całkowitego czasu potrzebnego do przetworzenia wszystkich elementów.

(a) Sformułuj powyższy problem jako zagadnienie ILP. Zaproponowany model musi uwzględniać następujące ograniczenia: (1) dwa elementy nie mogą być przetwarzane na jednej maszynie w tym samym czasie oraz (2) nie można rozpocząć przetwarzania elementu $i \in \{1, 2, 3\}$ na maszynie $j + 1$, dopóki nie zakończyło się jego przetwarzanie na maszynie j, $j \in \{1, 2\}$. HINT: Możesz zdefiniować zmienne decyzyjne x_{ij} jako czas rozpoczęcia przetwarzania elementu i na maszynie j.

(b) Załóżmy, że poszczególne elementy muszą być przetwarzane w tej samej kolejności na każdej z maszyn. Zmodyfikuj model ILP z podpunktu (a) tak, aby uwzględniał to dodatkowe ograniczenie.

Zadanie 4. (*Dyskretny problem plecakowy*)

(a) Przedstaw działanie podanego na wykładzie algorytmu opartego o programowanie dynamiczne dla następującej instancji problemu: $n = 4$, $W = 5$, $\overline{w} = (2, 3, 1, 4)$, $\overline{v} = (4, 5, 3, 7)$.

(b) Pokaż, że strategia zachłanna dla dyskretnego problemu plecakowego nie gwarantuje uzyskania rozwiązania o wartości $\geqslant \varepsilon \cdot V_{OPT}$ dla żadnej stałej $0 < \varepsilon < 1$, gdzie V_{OPT} to wartość rozwiązania optymalnego. HINT: W konstrukcji kontrprzykładu wystarczy użyć dwóch przedmiotów.

(c) Pokaż, jak zmodyfikować ten algorytm, żeby wartość zwróconego rozwiązania była nie mniejsza niż połowa wartości rozwiązania optymalnego.

Zadanie 5. Rozważmy następującą prostą heurystykę NEXT FIT dla problemu BIN PACKING. Początkowo wszystkie pudełka są puste. Rozpoczynamy od pudełka $j = 1$ i elementu $i = 1$. Jeśli w pudełku j zmieści się przedmiot i, to wkładamy go tam i rozważamy kolejny przedmiot $(i \leftarrow i + 1)$, a jeśli nie, to zamykamy pudełko j (nigdy do niego już nie wracamy) i otwieramy kolejne pudełko $(j \leftarrow j + 1)$. Powtarzamy procedurę, aż spakujemy wszystkie przedmioty.

(a) Jaka jest złożoność czasowa algorytmu NEXT FIT?

(b) Pokaż, że NEXT FIT użyje co najwyżej 2 razy tyle pudełek co w rozwiązaniu optymalnym.

(c) Dla dowolnego $n \in \mathbb{N}$ skonstruuj instancję o $\geqslant n$ pudełkach / przedmiotach, dla której algorytm NEXT FIT użyje dokładnie $2 \cdot k^* - 1$ pudełek, gdzie k^* jest optymalną liczbą pudełek.

Uwaga. Naturalnym ulepszeniem strategii NEXT FIT jest algorytm FIRST FIT, który dla każdego przedmiotu szuka pierwszego z dotychczas otwartych pudełek, do którego ten się zmieści. Oznaczając przez k^* optymalną liczbę pudełek można pokazać, że FIRST FIT użyje $k \leqslant \lceil 17/10 \cdot k^* \rceil$ pudełek. Jeśli na początku dodatkowo posortujemy przedmioty nierosnąco według rozmiarów, to otrzymamy strategię FIRST FIT DECREASING, która działa w czasie $O(n^2)$ i używa $k \leqslant 3/2 \cdot k^*$ pudełek (znane jest też inne ograniczenie postaci $k \leqslant 11/9 \cdot k^* + 4$). Jeśli $P \neq NP$, to „lepiej się nie da", tzn. nie istnieje algorytm wielomianowy, który dla dowolnej instancji użyje $k \leqslant \rho \cdot k^*$ pudełek dla pewnej stałej $\rho < 3/2$. Te fakty pokażemy na ćwiczeniach do kursu *Teoria obliczeń i złożoność obliczeniowa* (II stopień INA).

Zadanie 6. Sformułuj problem komiwojażera (TRAVELLING SALESMAN PROBLEM) jako zagadnienie ILP. HINT: Możesz poszukać wskazówek np. w rozdziale 9.1 w [BHM77] lub w przykładzie Application 16.2 w [AMO93] (patrz także zadanie 16.20 w [AMO93]).

Zadanie 7. (*Redukcja problemu* 3SAT *do* INTEGER LINEAR PROGRAMMING)
W NP-zupełnym problemie 3SAT dana jest formuła logiczna $\varphi(x_1, \ldots, x_n)$ w postaci koniunkcji klauzul zawierających alternatywę co najwyżej 3 literałów (tj. zmiennych lub ich negacji) i pytamy się, czy jest ona spełnialna, tj. czy istnieje wartościowanie $\pi = (\pi_1, \ldots, \pi_n) \in \{0, 1\}^n$, dla którego $\varphi(\pi) = 1$ (rozważamy wersję decyzyjną problemu 3SAT).

Przykładowo, w wersji z dokładnie trzema literałami w każdej klauzuli, formułę 3SAT możemy zapisać następująco:

$$\varphi(x_1, \ldots, x_n) = \bigwedge_{i=1}^{m} (l_{i_1} \vee l_{i_2} \vee l_{i_3}), \quad \text{gdzie } l_{i_j} \in \{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}.$$

Zredukuj problem 3SAT do decyzyjnej wersji problemu INTEGER LINEAR PROGRAMMING, tj. pokaż, że dla każdej formuły logicznej $\varphi(x_1, \ldots, x_n)$ w postaci 3SAT możemy w czasie wielomianowym względem jej rozmiaru skonstruować egzemplarz problemu programowania liniowego całkowitoliczbowego, dla którego istnieje rozwiązanie dopuszczalne wtedy i tylko wtedy, gdy formuła φ jest spełnialna.

Podaj przykład takiego zagadnienia ILP dla poniższej formuły.

$$\varphi(x_1, x_2, x_3) = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_3)$$

Literatura

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., USA, 1993.

[Bea] John E. Beasley. OR-Notes. `https://people.brunel.ac.uk/~mastjjb/jeb/or/contents.html`. Operations research notes by prof. J. E. Beasley.

[BHM77] S.P. Bradley, A.C. Hax, and T.L. Magnanti. *Applied Mathematical Programming*. Addison-Wesley Publishing Company, 1977.

[Sou12] Alexander Souza. Combinatorial Optimization. `https://ac.informatik.uni-freiburg.de/lak_teaching/ws11_12/combinatorial_optimization.php`, 2012. Universität Freiburg, Graduate Course – Winter Term 2011/12.