MODUŁ OPTYMALIZACJI ASORTYMENTU PRODUKCJI UWZGLĘDNIAJĄCY POPYT RYNKOWY

Andrzej BOŻEK

Streszczenie: W pracy omówiono koncepcję i sposób implementacji modułu optymalizacji asortymentu produkcji. Celem optymalizacji jest maksymalizacja zysku, określonego jako różnica pomiędzy przychodem ze sprzedaży i kosztem produkcji. Przychody obliczane są z uwzględnieniem funkcji popytu, które uzależniają ceny jednostkowe wyrobów od rozmiarów ich sprzedaży. Problem został sprowadzony do zadania programowania kwadratowego. W rozwiązaniu wykorzystano funkcję optymalizacji liniowo-kwadratowej z bezpłatnego pakietu inżynierskiego Scilab.

Słowa kluczowe: optymalizacja asortymentu, programowanie kwadratowe, Scilab, ERP.

1. Wprowadzenie

Problem optymalizacji asortymentu produkcji jest w swojej podstawowej postaci dobrze znanym zadaniem programowania liniowego [1, 2]. Zmienne decyzyjne określają wielkość produkcji poszczególnych wyrobów. Kryterium optymalizacji stanowi całkowity zysk. Ograniczenia wynikają z limitu zasobów potrzebnych do produkcji oraz z maksymalnych wielkości produkcji, warunkowanych popytem. Czasami dodaje się ograniczenia narzucające proporcje pomiędzy wielkościami produkcji niektórych wyrobów lub stanowiące, aby dla pewnych wyrobów wielkości produkcji nie były mniejsze od zadanych wartości minimalnych.

W praktyce zarządzania przedsiębiorstwami wytwórczymi dokonywanie numerycznych wyliczeń optymalnego asortymentu produkcyjnego nie jest spotykane często. Można próbować wskazać kilka przyczyn takiego stanu rzeczy:

1. Niewiele przedsiębiorstw, zwłaszcza małych i średnich, posiada rozbudowane systemy klasy MRP/ERP, pozwalające łatwo realizować zaawansowane analizy, oparte na automatycznie archiwizowanych parametrach produkcji.
2. Podstawowe wyliczenia można zrealizować w arkuszach kalkulacyjnych, np. z wykorzystaniem dodatku Solver arkusza MS Excel [3]. Ręczna edycja znacznej liczby danych przy dużym koszyku produktów jest jednak kłopotliwa, a właściwa reprezentacja zależności i ograniczeń wymaga odpowiedniej znajomości arkusza kalkulacyjnego i podstaw teorii optymalizacji.
3. Poza utrudnieniami związanymi z wygodą obsługi, funkcjonalne ograniczenia wykluczają arkusze kalkulacyjne jako narzędzia do realizacji zadań optymalizacyjnych o większych rozmiarach. Na przykład liczba komórek decyzyjnych w Excel-Solver ograniczona jest do 200.

W związku z wymienionymi niedogodnościami oraz uwzględniając wymagania pewnego przedsiębiorstwa, realizującego szeroko-asortymentową produkcję wyrobów papierniczych, opracowano programowy moduł optymalizacji asortymentu produkcji. Moduł spełnia następujące założenia:

1. Maksymalizowany zysk obliczany jest jako różnica między przychodem ze sprzedaży i kosztami produkcji. Wszystkie trzy wartości: zysk, przychód i koszt, uzyskane dla optymalnego asortymentu, powinny być zwracane użytkownikowi.
2. Przy obliczaniu zysku, cena jednostkowa wyrobów powinna być uzależniona od wielkości sprzedaży za pomocą funkcji popytu. Można przyjąć liniową aproksymację tej funkcji.
3. Moduł powinien obsłużyć znaczną liczbę zmiennych decyzyjnych, rzędu tysiąca, a dane określające problem optymalizacyjny mają być przekazywane automatycznie pomiędzy modułem i systemem informatycznym przedsiębiorstwa za pomocą ściśle zdefiniowanego interfejsu.

Cena jednostkowa, uzależniona liniowo od wielkości produkcji, w iloczynie z rozmiarem produkcji wprowadza składniki funkcji kryterialnej kwadratowo zależne od wartości zmiennych decyzyjnych. Ponieważ wszystkie ograniczenia można uznać za liniowe, problem jest zadaniem programowania kwadratowego, dla którego opracowano wiele wydajnych algorytmów rozwiązania. W prezentowanej pracy wykorzystano funkcję *qld* z bezpłatnego pakietu Scilab [4, 5]. Wszystkie operacje dodatkowe, do których należy wymiana danych z funkcją optymalizującą, wymiana danych z systemem informatycznym przedsiębiorstwa i realizacja pomocniczych obliczeń, zostały zaimplementowane z użyciem języka Java.

2. Sformułowanie problemu

Zadanie optymalizacji dotyczy zbioru n produktów. W przedsiębiorstwie realizowanych jest m rodzajów procesów wytwórczych. Każdy produkt wymaga przetworzenia w z góry zadanym ciągu procesów, reprezentując dla każdego procesu określoną czasochłonność.

We wszystkich dalszych zapisach zmienna i będzie indeksem odpowiadającym jednemu z produktów, tzn. $i \in \{1, 2, \ldots, n\}$, natomiast zmienna j będzie indeksem odpowiadającym jednemu z procesów, tzn. $j \in \{1, 2, \ldots, m\}$.

Wektor kolumnowy

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T,\ x_i \geq 0 \tag{1}$$

reprezentuje zbiór zmiennych decyzyjnych, wartość x_i określa wielkość produkcji i-tego wyrobu.

Macierz

$$\mathbf{tc} = \begin{bmatrix} tc_{11} & tc_{12} & \cdots & tc_{1m} \\ tc_{21} & tc_{22} & \cdots & tc_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ tc_{n1} & tc_{n2} & \cdots & tc_{nm} \end{bmatrix},\ tc_{ij} \geq 0,\ (\forall i \in n)(\exists j \in m)(tc_{ij} > 0) \tag{2}$$

definiuje czasochłonności, element tc_{ij} oznacza czas potrzebny na przetworzenie jednostki produktu i w procesie j.

Koszt realizacji procesu j przez czas jednostkowy określa element k_j wektora kolumnowego

$$\mathbf{k} = \left[k_1 \quad k_2 \quad \cdots \quad k_m\right]^T,\ k_j \geq 0\,. \tag{3}$$

Całkowity koszt produkcji jest wartością wyrażenia

$$koszt = \mathbf{x}^T \cdot \mathbf{tc} \cdot \mathbf{k}\,. \tag{4}$$

Cenę jednostkową i-tego produktu określa liniowa aproksymacja funkcji popytu $cena\_jednostkowa = a_i x_i + b_i$, gdzie a_i oraz b_i są elementami macierzy odpowiednio

$$\mathbf{ad} = diag\left(a_1, a_2, \ldots, a_n\right),\ a_i < 0 \text{ oraz } \mathbf{bd} = \left[b_1 \quad b_2 \quad \cdots \quad b_n\right]^T,\ b_i > 0\,, \tag{5}$$

stąd całkowity przychód ze sprzedaży

$$przychód = \mathbf{x}^T \cdot (\mathbf{ad} \cdot \mathbf{x} + \mathbf{bd})\,. \tag{6}$$

Ostatecznie, zysk przedsiębiorstwa określa wyrażenie

$$zysk = przychód - koszt = \mathbf{x}^T \cdot \mathbf{ad} \cdot \mathbf{x} + \mathbf{x}^T \cdot (\mathbf{bd} - \mathbf{tc} \cdot \mathbf{k})\,, \tag{7}$$

którego wartość należy zmaksymalizować.

Maksymalna dostępność czasowa av_j procesu j jest elementem wektora kolumnowego

$$\mathbf{av} = \left[av_1 \quad av_2 \quad \cdots \quad av_m\right]^T,\ av_j \geq 0\,, \tag{9}$$

zatem ograniczenie wynikające z dostępności procesów przyjmuje postać nierówności

$$\mathbf{tc}^T \cdot \mathbf{x} \leq \mathbf{av}\,. \tag{10}$$

Wektory kolumnowe

$$\begin{aligned} \mathbf{dn} &= \left[dn_1 \quad dn_2 \quad \cdots \quad dn_n\right]^T,\ dn_i \geq 0, \\ \mathbf{up} &= \left[up_1 \quad up_2 \quad \cdots \quad up_n\right]^T,\ up_i \geq dn, \end{aligned} \tag{11}$$

reprezentują odpowiednio dolne i górne granice wielkości produkcji kolejnych wyrobów, istnieje więc ograniczenie nierównościowe

$$\mathbf{dn} \leq \mathbf{x} \leq \mathbf{up}\,. \tag{12}$$

Dodatkowo uwzględnione zostają macierze

$$\mathbf{eq} = \begin{bmatrix} eq_{11} & eq_{12} & \cdots & eq_{1n} \\ eq_{21} & eq_{22} & \cdots & eq_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ eq_{k1} & eq_{k2} & \cdots & eq_{kn} \end{bmatrix} \in \Re^{k\,x\,n}, \qquad (13)$$

$$\mathbf{er} = \begin{bmatrix} er_1 & er_2 & \cdots & er_k \end{bmatrix}^T \in \Re^{k\,x\,1},\ k \in \mathbf{N}_+ \cup \{0\}$$

które pozwalają uzupełnić ograniczenia o zbiór *k* dowolnych liniowych ograniczeń równościowych postaci

$$\mathbf{eq} \cdot \mathbf{x} = \mathbf{er}\,, \qquad (14)$$

a także analogicznej postaci macierze

$$\mathbf{nq} = \begin{bmatrix} nq_{11} & nq_{12} & \cdots & nq_{1n} \\ nq_{21} & nq_{22} & \cdots & nq_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ nq_{p1} & nq_{p2} & \cdots & nq_{pn} \end{bmatrix} \in \Re^{p\,x\,n}, \qquad (15)$$

$$\mathbf{nr} = \begin{bmatrix} nr_1 & nr_2 & \cdots & nr_p \end{bmatrix}^T \in \Re^{p\,x\,1},\ \ p \in \mathbf{N}_+ \cup \{0\}$$

umożliwiające uwzględnienie *p* dodatkowych liniowych ograniczeń nierównościowych

$$\mathbf{nq} \cdot \mathbf{x} \le \mathbf{nr}\,. \qquad (16)$$

Opcjonalność dodatkowych ograniczeń wyraża się możliwością wystąpienia zerowej liczby wierszy w macierzach (13) i (15).

3. Moduł optymalizacyjny

Reprezentacja zadania optymalizacji, jakiej wymaga funkcja *qld* Scilaba ma postać

$$\begin{aligned} &\text{minimalizuj wartość:} && J = \frac{1}{2}\mathbf{x}^T \cdot \mathbf{Q} \cdot \mathbf{x} + \mathbf{p}^T \cdot \mathbf{x} \\ &\text{przy ograniczeniach:} && \mathbf{C}(j,:) \cdot \mathbf{x} = \mathbf{b}(j), \text{ dla } j = 1, \ldots, me \\ & && \mathbf{C}(j,:) \cdot \mathbf{x} \le \mathbf{b}(j), \text{ dla } j = me+1, \ldots, me+md \\ & && \mathbf{ci} \le \mathbf{x} \le \mathbf{cs} \end{aligned} \qquad (17)$$

gdzie: **x** – wektor kolumnowy zmiennych decyzyjnych,

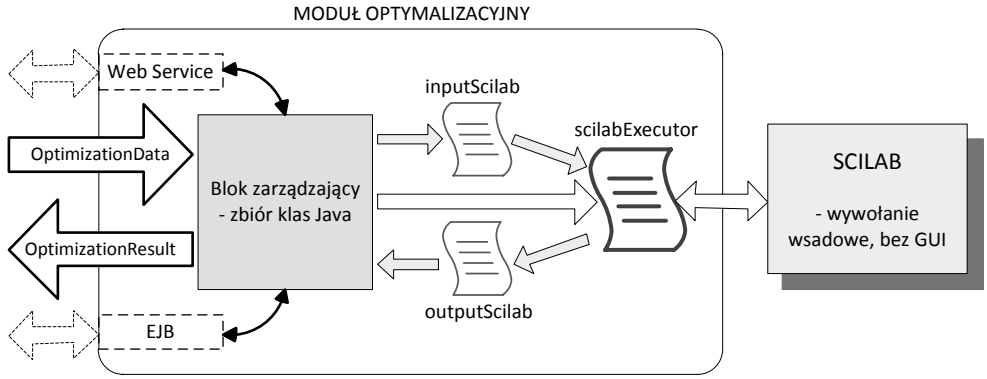
Q, **p**, **C**, **b**, **ci**, **cs** – macierze parametrów ,
me, *md* – liczba ograniczeń, odpowiednio równościowych i nierównościowych,
C(*j*,:) – wydzielenie z macierzy **C** wiersza *j*.

Ponieważ definicja zadania optymalizacji (17) zakłada minimalizację funkcji kryterialnej, przyjęte zostało podstawienie *J* = *-zysk*. Łatwo sprawdzić, że problem zdefiniowany danymi i relacjami (2)-(16) sprowadza się do problemu (17) poprzez podstawienia

$$\mathbf{Q} = -2\mathbf{ad},\ \mathbf{p} = \mathbf{w}\cdot\mathbf{k} - \mathbf{bd},\ \mathbf{C} = \begin{bmatrix} \mathbf{eq} \\ \mathbf{nq} \\ \mathbf{tc}^T \end{bmatrix},\ \mathbf{b} = \begin{bmatrix} \mathbf{er} \\ \mathbf{nr} \\ \mathbf{av} \end{bmatrix},$$
$$\mathbf{ci} = \mathbf{dn},\ \mathbf{cs} = \mathbf{up},\ me = k,\ md = p + m \tag{18}$$

Funkcja *qld* wykorzystuje algorytm opracowany przez K. Schittkowski'ego [5, 6]. Algorytm ten wymaga, aby macierz **Q** była symetryczna i dodatnio określona. Taki warunek jest równoważny ograniczeniu postawionemu dla macierzy **ad** w definicji (5), które oznacza, że wszystkie funkcje popytu muszą być malejące.

Poglądowy schemat struktury modułu optymalizacyjnego ukazano na rysunku 1.



Rys. 1. Poglądowy schemat struktury modułu optymalizacyjnego

Blok zarządzający dostosowuje formaty danych tak, aby były właściwe dla poszczególnych interfejsów oraz steruje sekwencją czynności wykonywanych w procedurze optymalizacyjnej. Kolejność tych czynności jest następująca:

1. Moduł odbiera od klienta reprezentację zadania optymalizacyjnego, zdefiniowaną przez obiekt implementujący interfejs *OptimizationData*. Reprezentacja ta jest programistycznym odzwierciedleniem opisu zadania optymalizacji ujętego w danych i relacjach (2)-(16).
2. Generowany jest plik *inputScilab*, który zawiera zadanie optymalizacyjne dla Scilaba, w postaci zbioru macierzy odpowiadającego reprezentacji (17).
3. Blok zarządzający uruchamia wsadowo skrypt Scilaba *scilabExecutor*, który wczytuje plik z danymi (*inputScilab*), wykonuje właściwy algorytm

optymalizacyjny i zapisuje wyniki w pliku wyjściowym *outputScilab*.

4. Blok zarządzający odczytuje dane wynikowe z pliku *outputScilab* i zwraca je po przetworzeniu formatu do klienta, w postaci obiektu implementującego interfejs *OptimizationResult*.

Na dane wynikowe składają się wyznaczone wartości zmiennych decyzyjnych, tj. optymalne wielkości produkcji poszczególnych wyrobów, oraz wartości całkowitego kosztu, przychodu i zysku. Dodatkowo zwracane są wartości obciążenia czasowego procesów. Dla każdego z wyrobów i procesów udostępniane są także zmienne logiczne, określające czy osiągnięto górne lub dolne ograniczenie w dopuszczalnym zakresie zmian danej wielkości.

Wywołanie usługi optymalizacyjnej aplikacja kliencka realizuje za pomocą pojedynczej metody *OptimizationResult invokeOptimizer(OptimizationData optimizationData)*. Dostępna jest także przeciążona wersja tej metody *OptimizationResult invokeOptimizer(OptimizationData optimizationData, double[] x)*, pozwalająca narzucić wartości zmiennych decyzyjnych. W drugim przypadku optymalizacja nie jest przeprowadzana, jedynie obliczane są wartości określone relacjami (4), (6), (7) dla zadanego wektora **x**. Funkcjonalność taka jest przydatna, gdy po zasadniczej optymalizacji zachodzi potrzeba zbadania wpływu pewnych parametrów na jej rezultat.

Z uwagi na zastosowanie technologii Java, istnieje możliwość łatwego rozszerzenia modułu do postaci komponentu EJB (*Enterprise Java Bean*) lub usługi sieciowej (*web service*).

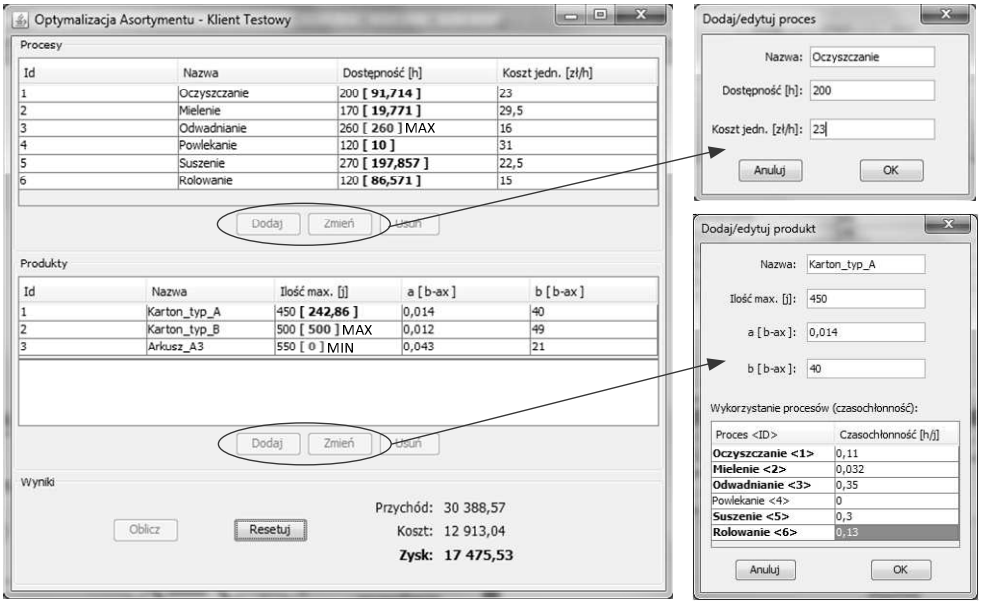
4. Wykorzystanie modułu

Omówione zostaną trzy przykłady wykorzystania modułu. Pierwszy dotyczy jego połączenia z testowym oprogramowaniem klienckim. Drugi przykład przedstawia uzupełnienie modułu o dodatkowy blok, pozwalający na optymalizację uwzględniającą koszty stałe procesów. W trzecim przykładzie zaprezentowana zostanie koncepcja połączenia modułu z systemem klasy MRP/ERP.

4.1. Testowe oprogramowanie klienckie

Do współpracy z modułem utworzony został testowy program kliencki, pozwalający na wprowadzenie zadania optymalizacyjnego przy pomocy prostego graficznego interfejsu użytkownika (rys. 2). Z uwagi na konieczność ręcznej edycji danych, rozwiązanie to nie jest przeznaczone dla zadań o dużym rozmiarze, zostało natomiast przewidziane do następujących zastosowań:

1. Weryfikacja poprawności działania modułu poprzez porównanie wyników uzyskanych dla kilku zadań próbnych w kliencie testowym z wynikami otrzymanymi przy wykorzystaniu niezależnego pakietu optymalizacyjnego. Taka weryfikacja została wykonana z pomocą arkusza kalkulacyjnego Excel i zakończyła się pozytywnie.
2. Prezentacja możliwości modułu przed jego wdrożeniem jako składnika zintegrowanego systemu zarządzania produkcją.
3. Rzeczywiste planowanie asortymentu produkcji w przypadku zadań o niewielkich rozmiarach, dla których ręczna edycja danych nie będzie uciążliwa.



Rys. 2. Klient testowy modułu optymalizacyjnego – graficzny interfejs użytkownika

Program kliencki uwzględnia wszystkie parametry zadania optymalizacyjnego, przedstawione w poprzednim rozdziale, poza liniowymi ograniczeniami równościowymi i nierównościowymi o dowolnej postaci, zdefiniowanymi relacjami (14) i (16). Do tabel wprowadzane są parametry procesów i produktów. Każdemu z produktów można przydzielić dowolny podzbiór dostępnych procesów oraz dla każdej pary (proces, produkt) można przypisać dowolną czasochłonność. Po wykonaniu obliczeń, wyznaczone wartości zmiennych decyzyjnych wyświetlane są w tabeli produktów, w nawiasach kwadratowych obok maksymalnych rozmiarów produkcji (rys. 2). Także wyliczone wartości obciążenia procesów wyświetlane są w jednej kolumnie wraz z limitami dostępności tychże procesów. Takie zestawienie ma na celu umożliwić łatwą poglądową ocenę wielkości produkcji wyrobów i obciążeń procesów dla najbardziej opłacalnego wariantu. Osiągnięcie przez wielkość produkcji lub obciążenie procesu wartości granicznej sygnalizowane jest dodatkowo oznaczeniami MIN/MAX.

4.2. Optymalizacja z uwzględnieniem kosztów stałych procesów

W bardziej szczegółowym ujęciu zadania optymalizacji asortymentu produkcji, w stosunku do sformułowania podanego w rozdziale 2, można uwzględnić koszty stałe procesów wytwórczych. Koszty te są związane z utrzymaniem maszyn oraz budynków potrzebnych do realizacji procesów i z innymi czynnikami, które generują koszty niezależne od rozmiaru produkcji, a zależne jedynie od przedziału czasowego, dla którego są naliczane. Przy założeniu niezmiennej puli procesów, koszty stałe nie zależą od wartości zmiennych decyzyjnych i przez to nie mają wpływu na procedurę optymalizacyjną w jej dotychczas rozważanej postaci. Zakres analizy ekonomicznej można jednak pogłębić, rozważając możliwość eliminacji wybranych procesów. Usunięcie procesu likwiduje związane z nim koszty stałe, a jednocześnie uniemożliwia wytwarzanie wszystkich

produktów, które tego procesu wymagają, likwidując zyski ze sprzedaży tych produktów. W sumarycznym bilansie optymalny zysk osiągalny po eliminacji procesu lub kilku procesów może okazać się większy niż przed eliminacją.

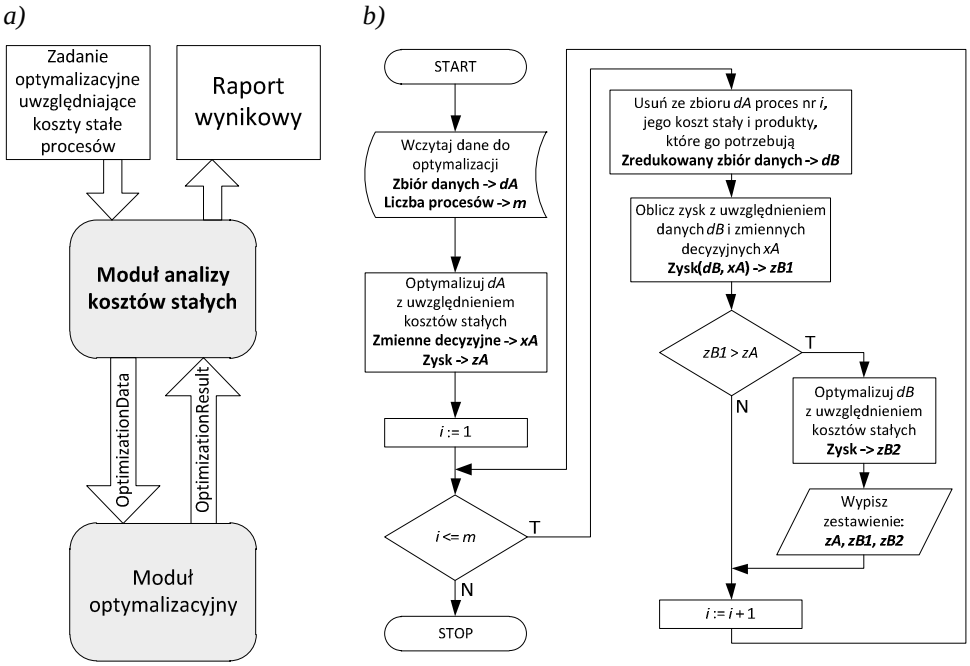
Moduł optymalizacyjny, omówiony w rozdziale 3, połączono z dodatkowym modułem analizy kosztów stałych (rys. 3a). Uzyskana struktura pozwala na optymalizację uwzględniającą możliwość eliminacji pojedynczych procesów w celu zwiększenia zysku. Zadanie optymalizacyjne poza danymi i relacjami (2)-(16) zostaje uzupełnione wektorem

$$\mathbf{cc} = \begin{bmatrix} cc_1 & cc_2 & \cdots & cc_m \end{bmatrix}^T, \; cc_j \geq 0, \tag{19}$$

w którym cc_j jest wartością kosztów stałych procesu j. Wartości *koszt* i *zysk* z relacji (4) i (7) zostają zastąpione nowymi wartościami, odpowiednio *koszt_2* i *zysk_2*, określonymi zależnościami

$$koszt\_2 = koszt + ccs, \;\; zysk\_2 = \; zysk - ccs, \;\; ccs = \sum_{j=1}^{m} \mathbf{cc}(j). \tag{20}$$

Na rysunku 3b ukazano schemat blokowy algorytmu realizowanego przez moduł analizy kosztów stałych. Po wczytaniu danych, dokonywana jest optymalizacja z uwzględnieniem wszystkich procesów i ich kosztów stałych. Uzyskany wektor wartości zmiennych decyzyjnych zapisywany jest do zmiennej *xA*, a wartość zysku do zmiennej *zA*.



Rys. 3. Optymalizacja z uwzględnieniem kosztów stałych: a) schemat połączenia modułów, b) schemat blokowy algorytmu realizowanego przez moduł analizy kosztów stałych

Następnie tworzone są zredukowane zestawy danych, z których każdy powstaje przez usunięcie jednego procesu z zestawu danych bazowych. Dla każdego zestawu zredukowanego *dB*, w oparciu o zmienne decyzyjne *xA* (z pominięciem zmiennych dotyczących usuniętych produktów), obliczana jest wartość zysku (*zB1*) i porównywana z zyskiem optymalnym uzyskanym dla danych bazowych (z*A*). Jeżeli zostanie stwierdzony wzrost zysku spowodowany usunięciem procesu, tzn. *zB1* > *zA*, zredukowany zestaw danych *dB* poddawany jest optymalizacji, dla której wyliczony zysk zapisywany jest w zmiennej *zB2*, a następnie porównawcze zestawienie wartości *zA*, *zB1*, *zB2* dodawane jest do raportu wynikowego. Efektem działania modułu jest wygenerowanie raportu, zestawiającego te procesy, których usunięcie skutkuje zwiększeniem zysku.

Dla przetestowania działania modułu analizy kosztów stałych, a równocześnie także w celu przetestowania podstawowego modułu optymalizacyjnego w pracy za znaczną liczbą zmiennych decyzyjnych, utworzono algorytm losowej generacji zadania optymalizacyjnego o zadanej liczbie produktów. Parametry dla poszczególnych procesów i produktów określane są przez wartości zmiennych losowych o rozkładzie jednostajnym i zakresach podanych w tabeli 1. Przyjęto liczbę procesów równą 50 oraz liczbę produktów (zmiennych decyzyjnych) równą 1000.

Tab. 1. Charakterystyka rozkładu losowego parametrów testowego zadania optymalizacji

	Dostępność czasowa procesu	**Kosz jedno-stkowy procesu**	**Koszt stały procesu**	**Maksy-malna wielkość produkcji**	**Wsp. *a* funkcji liniowej popytu**	**Wsp. *b* funkcji liniowej popytu**	**Czasochłon-ność dla pary (produkt, proces)**
Min	2500	0,5	20000	400	– 0,040	10	0,01
Max	5000	2,0	50000	800	– 0,002	50	0,20

W pierwszym wykonanym teście każdy z procesów przypisywany był do kolejnych produktów z jednakowym prawdopodobieństwem, wynoszącym 0,3. Przy liczbie procesów równym 50, uzyskano przypisanie średnio 15 procesów do każdego produktu. W tym przypadku eliminacja żadnego z procesów nie spowodowała zwiększenia zysku. Wynika to z obecności każdego procesu w zestawie technologicznym blisko co trzeciego produktu. Usunięcie kosztów stałych jednego z 50 procesów nie zwiększyło zysku ponad wartość strat związanych z eliminacją około 30% produktów.

W drugim teście zmodyfikowano sposób przypisywania procesów do produktów wykorzystując, zamiast jednostajnego, normalny rozkład prawdopodobieństwa. Dołączenie *j*-tego spośród *m* procesów do technologii wytwarzania każdego z produktów odbywało się z prawdopodobieństwem

$$p_{\mu,\sigma,pn}(j) = \frac{pn}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(j-\mu)^2}{\sigma}\right), \ \mu = \frac{m-1}{2}, \ \sigma = \frac{\mu}{3}, \qquad (21)$$

osiągającym minima dla skrajnych wartości indeksu *j*, oddalonych o przedział 3σ od centrum rozkładu. Współczynnik *pn* w zależności (21) zwielokrotnia prawdopodobieństwo określone funkcją Gaussa, dzięki czemu średnia liczba procesów dla każdego z produktów wynosi *pn*. Rezultaty uzyskane po wykonaniu optymalizacji uwzględniającej koszty stałe

procesów zostały zamieszczone w tabeli 2. Przyjęto *pn* = 15, tj. każdy produkt wymagał przetwarzania w średnio 15 procesach.

Tab. 2. Bilans zysków, uzyskany w optymalizacji uwzględniającej koszty stałe procesów

Numer procesu	**Zysk bazowy**	**Zysk po usunięciu procesu (poprawa procentowa)**	**Zysk po optymalizacji (poprawa procentowa)**
1	455466	475149 (4,3%)	489024 (7,4%)
3	455466	486632 (6,8%)	499350 (9,6%)
44	455466	465405 (2,2%)	494589 (8,6%)
49	455466	485928 (6,7%)	485928 (6,7%)
50	455466	468026 (2,8%)	495329 (8,8%)

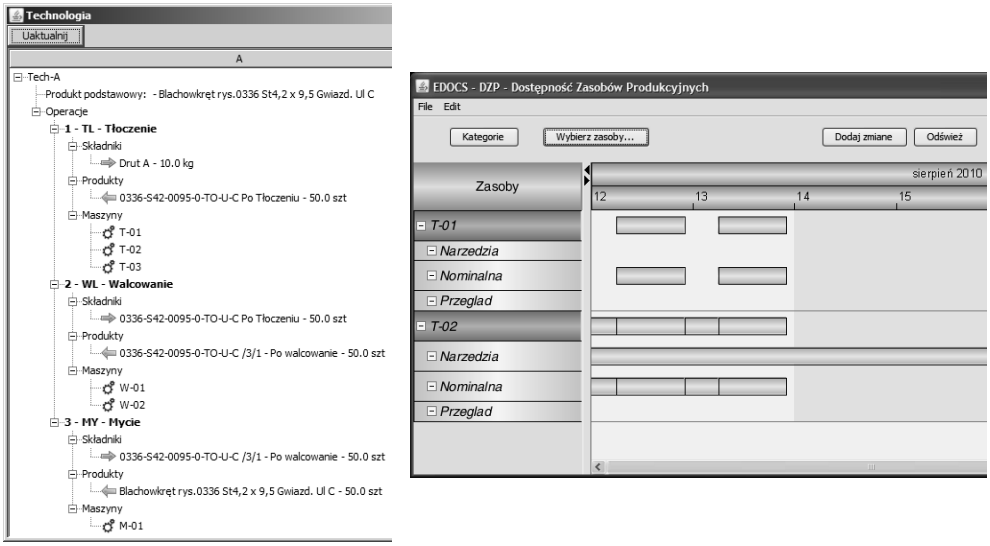
W przeprowadzonym teście oprogramowanie optymalizacyjne odnalazło pięć procesów, których eliminacja spowodowała wzrost zysku. W większości przypadków dodatkowa optymalizacja po usunięciu procesu poprawiała rezultaty w stosunku do wyników uzyskanych na podstawie bazowych wartości zmiennych decyzyjnych, co oznacza, że usunięcie procesu dezaktywowało taki podzbiór ograniczeń, iż możliwe stało się przemieszczenie w przestrzeni zmiennych decyzyjnych do punktu o mniejszej wartości funkcji kryterialnej, odpowiadającej większemu zyskowi. Poprawa nie musi wystąpić jednak w ogólności, jak pokazuje przykład procesu nr 49.

Wygenerowany przez oprogramowanie raport (tab. 2) podpowiada analitykowi, które z procesów będzie korzystnie usunąć z produkcji i w jakim stopniu się to opłaci. Oczywiście uzyskanie maksymalnego zysku w szczególnym przypadku może wiązać się z usunięciem nie jednego, lecz pary, trójki, czy większej liczby procesów. Może się także zdarzyć, że eliminacja jednego procesu nie spowoduje zwiększenia zysku dla bazowych wartości zmiennych decyzyjnych, ale zwiększy go po dodatkowej optymalizacji. Takich wariantów omówiona struktura optymalizacyjna nie będzie w stanie odnaleźć. Aby to umożliwić, należy albo zaprojektować nowy algorytm optymalizacyjny, bezpośrednio operujący na kosztach stałych jako integralnych danych wejściowych, albo rozbudować pętlę generującą zbiory usuwanych procesów w algorytmie z rysunku 3b. W obydwu przypadkach należy się liczyć ze wzrostem złożoności czasowej algorytmu. W wielu praktycznych zastosowaniach zaproponowany prosty mechanizm analizy kosztów stałych procesów powinien okazać się wystarczający.

4.3. Integracja z systemem klasy MRP/ERP

Moduł optymalizacyjny ma zostać docelowo przyłączony do systemu klasy MRP/ERP jako opcjonalny składnik usługowy. Do pierwszej integracji wybrano system EDOCS, wspomagający proces planowania i harmonogramowania produkcji. EDOCS jest oprogramowaniem wchodzącym w skład innowacyjnego systemu monitorowania i sterowania produkcji, rozwijanego w ramach Klastra Zielona Kuźnia, do którego przynależy Katedra Informatyki i Automatyki Politechniki Rzeszowskiej [7]. System EDOCS zawiera moduł definiowania technologii produkcji, pozwalający określić ciągi procesów produkcyjnych dla wyrobów i właściwe dla nich maszyny, wraz ze średnimi

wydajnościami przetwarzania maszyn, oraz moduł kontroli zasobów produkcyjnych, który umożliwia między innymi definiowanie kalendarzy dostępności maszyn (rys. 4a,b).



Rys. 4. Wybrane moduły sytemu EDOCS: a) moduł definiowania technologii, b) moduł kontroli zasobów produkcyjnych – kalendarze dostępności maszyn

Informacje określające technologie produkcji, wydajności maszyn i kalendarze ich dostępności, przechowywane w systemie EDOCS w jego obecnej wersji, mogą zostać w prosty sposób przekształcone do postaci danych reprezentujących macierz czasochłonności (2) oraz wektor ograniczeń dostępności czasowej procesów (9). Pozostałe dane potrzebne dla modułu optymalizacyjnego, posiadające charakter ekonomiczny, tj. koszty jednostkowe procesów, limity produkcji poszczególnych wyrobów oraz funkcje popytu, nie są w systemie EDOCS aktualnie przechowywane. Planowane jest rozszerzenie funkcjonalności systemu o zarządzanie wymienionymi informacjami, w celu umożliwienia integracji z modułem optymalizacyjnym.

5. Podsumowanie

W pracy przedstawiono koncepcję i implementację modułu optymalizującego asortyment produkcji, uwzględniającego liniowe funkcje popytu, kształtujące ceny jednostkowe wytwarzanych produktów. Motywacją projektu była rzeczywista potrzeba przedsiębiorstwa, które nie odnalazło oprogramowania właściwego do przeprowadzenia w wygodny sposób pożądanych analiz ekonomicznych.

Moduł optymalizacyjny, wraz z dedykowanym prostym oprogramowaniem klienckim, realizującym funkcję graficznego interfejsu użytkownika, opisanym w podrozdziale 4.1, stanowi kompletne narzędzie do rozwiązywania zadań o mniejszych rozmiarach. Może być ono wystarczające dla małych przedsiębiorstw z niewielkim koszykiem produktów.

Docelowo moduł przeznaczony jest dla problemów optymalizacyjnych o znacznych rozmiarach. Takie przeznaczenie wynika z możliwości reprezentacji problemu jako zadania programowania kwadratowego, które zostało dobrze zbadane i dla którego opracowano

efektywne algorytmów rozwiązań. W ramach testów prezentowanego modułu optymalizacyjnego, zadania złożone z 1000 produktów i średnio 15 procesów przypisanych każdemu produktowi rozwiązywane były w czasie kilku sekund na komputerze osobistym o średniej wydajności (procesor Intel Pentium Dual T2390@1.86GHz). W przypadku zadań o dużych rozmiarach, zakłada się bezpośrednie i automatyczne pobieranie danych do optymalizacji z zasobów informacyjnych systemu MRP/ERP. Do pierwszej integracji z modułem wybrano system EDOCS.

Moduł może być wykorzystany wraz z mówionym w podrozdziale 4.2 rozszerzeniem w postaci bloku analizy kosztów stałych procesów. Rozszerzenie to dodaje użyteczną funkcjonalność, pozostawiając bez zmian zasadniczy algorytm optymalizacyjny.

Literatura

1. Stachurski A.: Wprowadzenie do optymalizacji. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 2009.
2. Siudak M.: Badania operacyjne. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 1998.
3. Wayne L. W.: Microsoft Excel, Analiza i modelowanie danych. Microsoft Press, 2006.
4. Scilab Homepage, http://www.scilab.org/
5. Scilab Manual, http://www.scilab.org/download/5.2.2/manual_scilab-5.2.2_en_US.pdf
6. Schittkowski K.: QL: A Fortran Code for Convex Quadratic Programming - User's Guide, Version 2.11. Report, Department of Computer Science, University of Bayreuth, 2005.
7. Mączka T., Czech T., Żabiński T.: Innowacyjny system monitorowania i sterowania produkcją jako element fabryki przyszłości. Pomiary Automatyka Robotyka 2/2010, ss. 38-41.

Mgr inż. Andrzej BOŻEK
Katedra Informatyki i Automatyki
Politechnika Rzeszowska
35-959 Rzeszów, ul. W. Pola 2
tel.: (17) 865 17 66
e-mail: abozek@prz-rzeszow.pl