Piotr MODLIŃSKI, Tadeusz WITKOWSKI
Politechnika Warszawska

# ROZWIĄZANIE ELASTYCZNEGO PROBLEMU GNIAZDOWEGO Z WYKORZYSTANIEM UOGÓLNIONEGO PRZESZUKIWANIA ZACHŁANNEGO JAKO PRZYKŁAD PROBLEMU ONLINE

**Streszczenie.** W pracy przedstawiono problem harmonogramowania w gniazdach jako problem online. Dokonano krótkiego przeglądu elastycznych problemów gniazdowych. Zaprezentowano i oceniono metaheurystyczną metodę poszukiwania rozwiązań jako hybrydę algorytmu zachłannego i przeszukiwania siłowego w pewnym otoczeniu rozwiązania wyjściowego. Określenie zależności jakości rozwiązania od rozmiaru tego otoczenia jest zasadniczym celem pracy.

# SOLUTION OF THE FLEXIBLE JOB SHOP PROBLEM WITH GENERALIZED GREEDY SEARCH AS THE EXAMPLE OF AN ONLINE PROBLEM

**Summary.** The paper presents job shop scheduling problem as an online problem. Short research of elastic job shop sheduling problems has been made. Presented and rated the metaheuristic search method – hybrid of greedy algorithm and force search in surrounding of starting solution. Defining dependency between quality of solution and size of that surrounding is the main goal of the work.

## 1. Wstęp

Problemem online można określić dowolny z problemów, w których od początku nie są znane kompletne dane niezbędne do jego rozwiązania. Kolejne informacje są niejako „odsłaniane" w trakcie rozwiązywania. Nauki informacyjne znają wiele klasycznych problemów online – np. problem dostępu do listy (List Access Problem) dyskutowany przez Ambühla (2000), search games zdefiniowany przez Isaacsa (1965), czy wielorękiego bandyty (multi-armed bandit problem) opisywany przez Whittle'a (1981). Innym przykładem może być problem sekretarki/problem stopu (secretary problem) rozważany m.in. przez Beardena i Neila (2006). Wiele z nich jest niezwykle trudnych obliczeniowo nawet w wersji offline – por. np. prace Ambühla (2000) Występowanie niekompletnej informacji dodatkowo utrudnia, a często wręcz uniemożliwia rozwiązanie zadania.

We wszystkich problemach online istnieje potrzeba przewidywania przyszłych zdarzeń i podejmowania decyzji na podstawie takiej prognozy. Najbardziej oczywistym rozwiązaniem wydaje się znalezienie najlepszej możliwej prognozy (najdokładniejszej i najdłużej) – z zastrzeżeniem czasu potrzebnego na jej wyznaczenie, a następnie podjęcie decyzji na jej podstawie.

Celem niniejszej pracy jest stwierdzenie, czy rzeczywiście „lepsza" prognoza

przekłada się na osiągnięcie „lepszego" rozwiązania.

Badania przeprowadzono w oparciu o problem harmonogramowania w gniazdach produkcyjnych, z uwzględnieniem różnych konfiguracji ograniczeń, oraz jego rozszerzeń. Dane do obliczeń zaczerpnięte zostały z literatury, jednak pomimo ich znajomości a'priori problem zaliczyć można do klasy problemów online, gdyż w danym momencie algorytm „posiada wiedzę" jedynie lokalną (z dokładnością do „prognozy") – nie zna długości, ani wymagań dotyczących alokacji kolejnych operacji do maszyn, bądź ich grup (w tym też całkowitej liczby operacji). Dzięki wykorzystaniu identycznych zestawów danych możliwe jest końcowe porównanie wyników dla różnych wartości parametrów.

## 2. Rozważane problemy

Badania zostały przeprowadzone w oparciu o szereg trudnych (por. np. Błażewicz i in. 1983, 1996), problemów harmonogramowania w gniazdach produkcyjnych będących uproszczeniami modelu elastycznego, uwzględniającego występowanie grup zamiennych maszyn (ozn. G), czasy przezbrojeń (P), partie produkcyjne (N), oraz możliwości wykonywania określonych operacji w dowolnej kolejności (V) zaproponowanego przez Witkowskiego (1993).

Rysunek 1 przedstawia taksonomię 16 rozważanych problemów poczynając od klasycznego problemu gniazdowego (Job Shop – JS), rozluźniając konkretne ograniczenia (wymienione wyżej), dochodząc w końcu do najbardziej elastycznego i jednocześnie złożonego modelu (JSAPNV). Każdy z badanych problemów zalicza się do klasy problemów online.

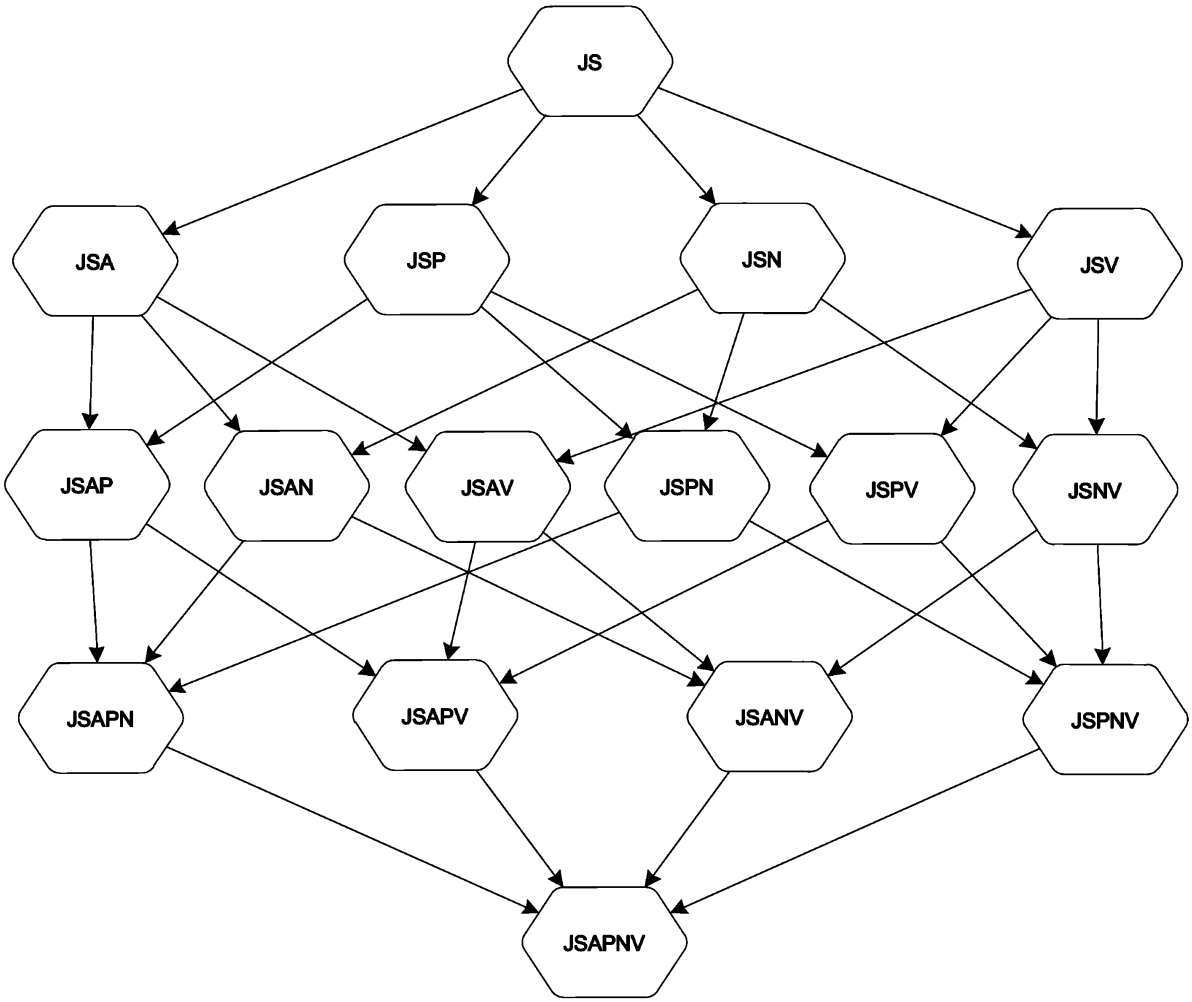
Dane testowe zaczerpnięto również z pracy Witkowskiego (1993), a także z innych, znanych z literatury testowych problemów gniazdowych (JS) używanych do weryfikacji jakości algorytmów optymalizacji zaproponowanych przez Fishera i Thompsona (1963), Lawrence'a (1984), oraz Storera, Wu i Vaccariego (1992).

## 3. Wykorzystywany algorytm

W ostatnich latach wykorzystywane i rozwijane jest wiele metaheurystycznych algorytmów, jak np. proponowany przez Smutnickiego i in. (2015) algorytm VESA oparty o ideę SA.

Podczas badań wykorzystano bardzo prosty, autorski algorytm „zachłanno-siłowy" (GB) – hybrydę podejścia zachłannego i brutalnego. Algorytm zachłanny na każdym etapie (punkt startowy oznaczono przez „St" na rysunku **??**) sprawdza konsekwencje każdego możliwego kroku (tu: dodania kolejnej operacji) i na ich podstawie wybiera ten, który jest lokalnie najlepszy. Przeszukiwane kroki oznaczone są cyfrą 0 na lewej części wspomnianego rysunku. Rozważana hybryda przeszukuje znacznie szersze otoczenie (zależne od parametru) – oznaczone odpowiednio cyframi 1, 2 itd. (dalszych numerów nie ma ze względu na rozmiar rysunku).

W rozważanym problemie istotne jest, że poszczególne kroki są nieporównywalne ze sobą. Nie można porównywać harmonogramu zbudowanego w części z już ukończonym. W każdym kroku zamiast podejmować decyzję na podstawie możliwych wyników jego zrobienia w perspektywie tylko jego, sprawdzane są (przeszukując siło-



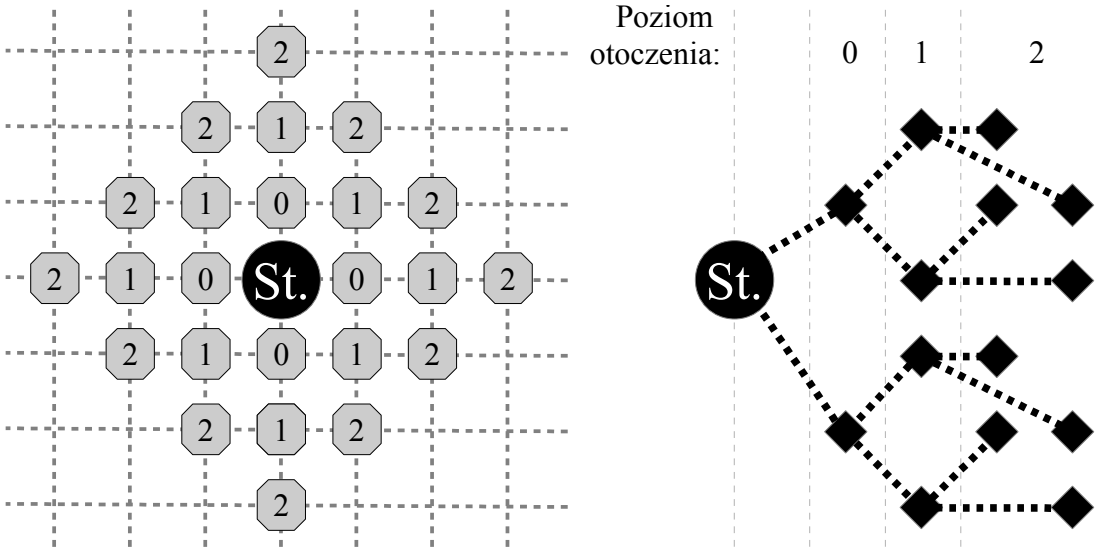

Rys. 1. Taksonomia rozważanych problemów harmonogramowania

wo) wszystkie rozwiązania, do których prowadzi taka, lub inna decyzja w określonej perspektywie (d kroków – parametr algorytmu). Na podstawie najlepszego znalezionego w ten sposób rozwiązania (rozwiązania optymalnego, nie suboptymalnego jako, że przeszukiwanie siłowe jest algorytmem dokładnym – mimo, że nieefektywnym pod względem czasowym) podejmowana jest decyzja lokalna, a następnie proces powtarza się od początku z przesuniętym już horyzontem przeszukiwania.

Rozwiązanie zaimplementowano w rekurencyjny sposób zaprezentowany odpowiednio na rysunkach: 2 – parametryzowane przeszukiwanie brutalne (siłowe), oraz 3 – algorytm zachłanno-brutalny.

Wprowadzony algorytm zachłanno-brutalny można określić wg. różnych kryteriów jako kombinatoryczno-grafowy optymalizacyjny algorytm pozwalający na uzyskanie przybliżonych rozwiązań w sposób deterministyczny. Algorytm ponadto pozwala na praktycznie darmowe (przy zaniedbywalnie małych narzutach) zrównoleglenie.

Jakkolwiek złożoność obliczeniowa algorytmu jest duża (zależność od wielkości problemu jest wprawdzie wielomianowa, jednak od wielkości parametru wykładnicza), bazuje jednak na dokładnej prognozie.




Rys. 2. Otoczenie rozwiązania

```
Brutal(obiekt, parametr)
    {
    Jeśli obiekt jest ukończony, lub parametr = 0
        Zwróć wartość obiektu
    W przeciwnym razie
        dla każdego dopuszczalnego kroku
            Y(krok) = Brutal(obiekt+krok, parametr-1)
        Zwróć najlepszy Y
    }
```

Rys. 3. Parametryzowane przeszukiwanie brutalne

Podczas symulacji zrezygnowano z wyznaczania jakości harmonogramu jedynie za pomocą jego długości — można pokazać, że w takiej sytuacji zawsze na początku szeregowane są zadania najkrótsze, a najdłuższe — te, które są najbardziej problematyczne, pozostają do wykonania na końcu. Zaproponowano własną funkcję oceny — odwzorowującą niejako efektywność wykorzystania czasu. Każdemu z zadań (prac – *job*) określono efektywny czas wykonywania (jest to suma czasów obróbki wszystkich elementów[1], lub całkowity czas, w przypadku, w którym nie rozważamy partii produkcyjnych[2]). Do efektywnego czasu nie zalicza się przezbrojeń — uważane są one za czas zmarnowany. Funkcją celu, jakości harmonogramu $Q(H)$ (teraz już maksymalizowaną) jest iloraz czasu efektywnego i całkowitej długości harmonogramu. Przedstawiono ją na

[1]W rozważanych problemach dopuszcza się możliwość występowania partii produkcyjnych – naturalnie występujące np. w przykładzie Witkowskiego (1993) składających się z wielu identycznych elementów danego produktu. Do wykonania każdego z nich wymagana jest realizacja wszystkich operacji, jednak przyjęto założenie, iż kolejna operacja na ***danym elemencie*** może rozpocząć się w momencie, w którym dany etap obróbki ***tego elementu*** się zakończył – nie ma konieczności oczekiwania na zakończenie obróbki całej partii

[2]Przypadek ten modelowany jest jako partia składająca się z jednego elementu

```
GB(obiekt, parametr)
    {
    Póki obiekt nie jest ukończony
        dla każdego dopuszczalnego kroku
            Y(krok) = Brutal(obiekt+krok,parametr)
        Dodaj do obiektu najlepszy krok (ten, który daje najlepszy Y(krok)
    Zwróć poprawiony obiekt
    }
```

Rys. 4. Parametryzowane przeszukiwanie brutalne

równaniu 1. Funkcje te obliczane są w każdym momencie, w którym dochodzi do oceny jakości harmonogramu (także częściowego). Łatwo pokazać, że optymalizacja obu funkcji jest równoważna (dla ukończonego harmonogramu).

$$Q(H) = \frac{\sum_j \sum_{o \in O_j : E(o,j)>0} \kappa_j t_{j,o}}{E(H)} \tag{1}$$

gdzie:

- $O_j$ – zbiór operacji pracy $j$
- $E(o, j)$ – chwila zakończenia operacji $o$ pracy $j$
- $\kappa_j$ – liczba elementów w ramach pracy $j$
- $t_{j,o}$ – czas potrzebny na wykonanie operacji $o$ pracy $j$ na pojedynczym elemencie

## 4. Eksperyment

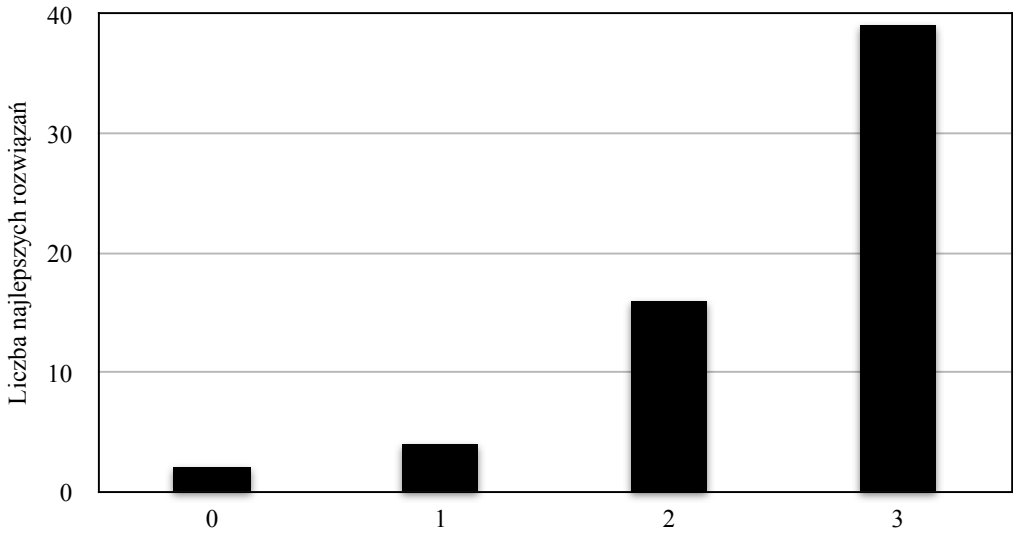
Przeprowadzono liczne (533) symulacje rozwiązywania problemów przedstawionych na rysunku 1, danych wyszczególnionych w rozdziale 2., a także dla różnych wartości parametru $d$.

Na podstawie przeprowadzonych badań udało się określić ogólną zależność pomiędzy liczbą najlepszych znalezionych rozwiązań, a długością prognozowania (parametr $d$). Zależność ta okazuje się prawdziwa bez względu na to, czy mamy do czynienia z problemem prostszym (JS), czy bardziej złożonym (elastycznym). Okazało się, że ze wzrostem parametru powstaje coraz większa szansa na znalezienie lepszego rozwiązania.

Na rysunku 4 przedstawiono opisaną zależność przy ograniczeniu się do parametrów ze zbioru $\{0, 1, 2, 3\}$.

Tylko dla tych wartości znaleziono rozwiązania wszystkich badanych problemów – niektóre okazały się zbyt duże do wyznaczenia z większymi wartościami w rozsądnym czasie przy wykorzystaniu posiadanego sprzętu – w szczególności problem zawierający 50 zadań (prac) zaproponowany przez Storera, Wu i Vaccariego (1992).

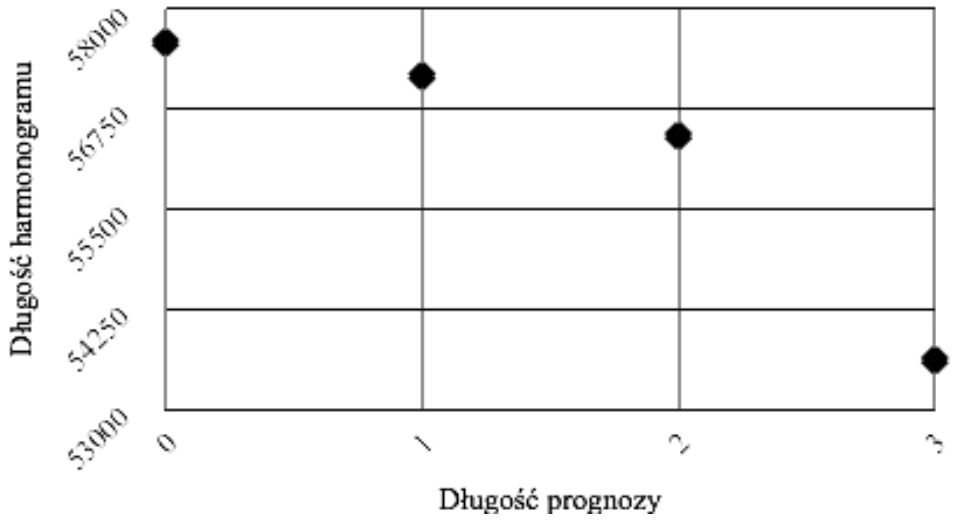
Okazuje się jednak, że zależność ta nie w każdym przypadku jest prawdziwa. Niejednokrotnie znajdowane w kolejnych krokach rozwiązania okazywały się optymalne w sensie Pareto, czasem jednak trafiały się wyniki zdominowane (w tym miejscu warto zwrócić uwagę, że to samo rozwiązanie – albo lepsze, znalezione dla krótszej prognozy zawsze będzie dominować to wyznaczone dla dłużej —- choćby ze względu




Rys. 5. Liczba najlepszych znalezionych rozwiązań w funkcji wartości parametru

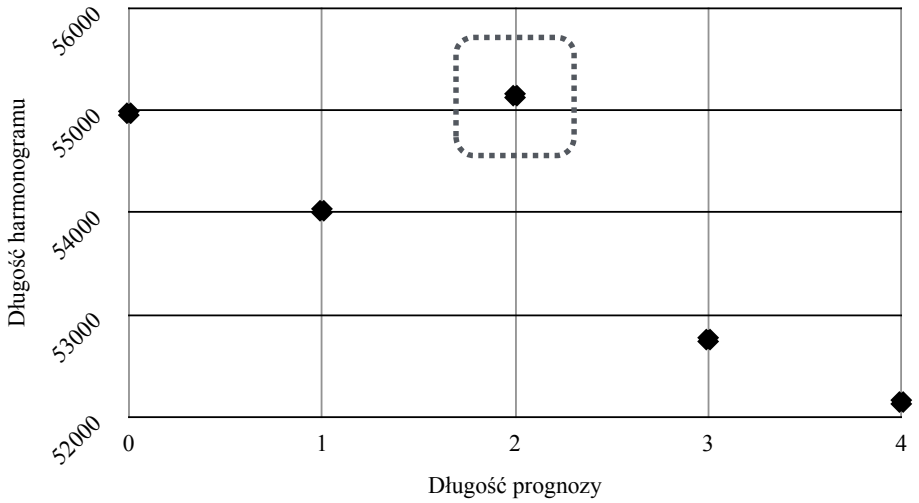
na czas obliczeń wymaganych do jej uzyskania). Zależności od wymagań pamięciowych nie rozważam, gdyż jest ona zależna w głównym stopniu od rozmiaru zadania (zależność od wartości parametru jest liniowa). Zależność czasu wykonywania od wielkości parametru jest wykładnicza.

Przykład ciągu rozwiązań optymalnych w sensie Pareto przedstawiono na rysunku 5. Rysunek 6 prezentuje często spotykaną sytuację, w której jedno z rozwiązań okazuje się gorsze od pozostałych -— niejako „wyłamuje się" z ogólnej tendencji.



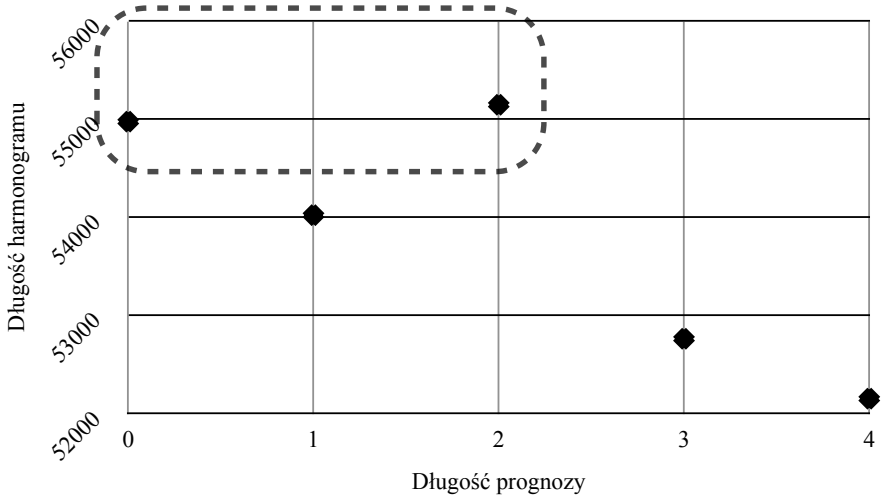

Rys. 6. Przykład ciągu rozwiązań Pareto-optymalnych

Zdarzają się też sytuacje takie, jak przedstawiona na rysunku 7, w której w pewnym momencie pojawia się szereg rozwiązań, które nawet się pogarszają. Analiza wielu




Rys. 7. Przykład pojedynczego rozwiązania zdominowanego

przypadków pozwoliła zauważyć, że również w takim przypadku dalsze wydłużanie prognozy pozwala na „przełamanie" tej tendencji i w końcu znalezienie rozwiązania niezdominowanego (oczywiście jeśli wcześniej nie udało się znaleźć rozwiązania optymalnego. . . )




Rys. 8. Przykład ciągu rozwiązań zdominowanych

## 5. Wnioski

Przeprowadzone badania częściowo potwierdziły intuicyjne podejrzenia, że zwiększenie długości prognozowania poprawi jakość rozwiązania. Faktycznie w większości przypadków tak się dzieje. Zwrócić jednak należy uwagę na fakt, że nie jest to regułą – zdarzają się sytuacje, w których występowanie dodatkowej informacji (tu: dłuższej prognozy) nie tyle nie pozwala osiągnąć lepszego rozwiązania, co pogarsza istniejące.

Ponieważ nie można a'priori określić, czy w badanym przypadku pojawi się taka nieregularność, słusznym wydaje się korzystanie z najlepszych (rozumianych jako najdokładniejszych i najdłuższych) dostępnych w danym momencie prognoz.

LITERATURA

1. Ambühl, C.: Offline list update is NP-hard. Springer Berlin Heidelberg, Algorithms-ESA 2000, p. 41-51
2. Błażewicz, J., Lenstra, J.K., Kan, A.R.: Scheduling subject to resource constraints: classification and complexity. Discrete Applied Mathematics 5(1), 1983, p. 11-24
3. Błażewicz, J., Drozdowski, M., deWerra, B., Węglrz, J.: Deadline scheduling of multiprocessor tasks. Discrete Applied Mathematics 65, 1996, p. 81-95
4. Bearden, J.N.: A new secretary problem with rank-based selection and cardinal payoffs. Journal of Mathematical Psychology 50(1), 2006, p. 58-59
5. Fisher, H., Thompson, G.L.: Probabilistic learning combinations of local job-shop scheduling rules Industrial Scheduling, Prentice Hall, Englewooc Cliffs, New Jersey 1963, p. 225-251
6. Isaacs, R: Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimizations, 1965
7. Lawrence, S.: Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement), Graduate School od Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania 1984
8. Smutnicki, C., i in.: A new approach for multi-criteria scheduling. Computers & Industrial Engineering 90, 2015, p. 212-220.
9. Storer, R.H., Wu, S.D., Vaccari, R.: New search spaces for sequencing instances with application to job shop scheduling. Management Science, 38, 1992, p. 1495-1509
10. Whittle, P.: Arm-acquiring bandits. The Annals of Probability, 9(2), 1981, p. 284-292.
11. Witkowski T.: Algorytm rozpoznawania problemu szeregowania zadań. Informatyka, 6, 1993, p. 20-26