

Waldemar Herka
Instytut Informatyki Teoretycznej i Stosowanej
ul. Dąbrowskiego, 73, 42-200 Częstochowa

HARMONOGRAMOWANIE PROCESÓW CIĄGLYCH W WARUNKACH NIEPEWNOŚCI JAKO WIELOKRYTERIALNY PROBLEM DECYZYJNY

Praca niniejsza zawiera pewne propozycje dotyczące optymalizacji planu produkcji (harmonogramowania) rzeczywistej walcowni oraz opis problemów tej dziedziny w warunkach rzeczywistych. Zwracamy szczególną uwagę na aspekty niepewności. Są one kluczem do uzyskania modelu dobrze odwzorowującego rzeczywistość, w której bardzo często spotykamy się z informacją nieprecyzyjną lub wręcz niepełną. Problemem, który chcemy uwypuklić w niniejszej pracy jest tzw. zagadnienie kolejnościowe bez postojów (no-wait flow shop lub continuous flow shop) ze względu na dziedzinę zastosowania opisywanych przez nas zagadnień. Przeprowadzona analiza rzeczywistego problemu, poparta przykładem ukazuje drogę od surowych danych do spójnej metody rozwiązywania problemu harmonogramowania. Opisano także często wykorzystywaną w zagadnieniach kombinatorycznych heurystyczną metodę lokalnych poszukiwań *Tabu Search*, która dzięki pewnym mechanizmom zapewniającym ukierunkowanie procesu poszukiwań jest użytecznym narzędziem w przypadku przestrzeni rozwiązań nie dających szans metodom analitycznym.

SŁOWA KLUCZOWE: Continuous flow shop scheduling, Tabu Search, Heurystyka, Harmonogramowanie, Graf dysjunkcyjny

1. Wstęp

Produkcja ukierunkowana na zaspokajanie konkretnych potrzeb klientów, a nie na zapełnianie magazynów to norma przyjęta przez każdy „zdrowy” podmiot gospodarczy . Wielkość produkcji musi odwzorowywać wielkość zamówień. Dawno minęły czasy, w których to klient dopasowywał swe wymagania do tego mu oferowano. W dobie walki o klienta, każde zamówienie – nawet najmniejsze – musi być traktowane poważnie. Dodatkowo, obniżanie kosztów produkcji przy podwyższaniu lub co najmniej utrzymaniu standardów jakości może stanowić użyteczne narzędzie pozwalające czynnie uczestniczyć w walce o klienta. Są to bardzo trudne zadania – szczególnie w branżach charakteryzujących się stosunkowo małą elastycznością możliwości zmian profilu produkcji (podyktowaną bardzo wysokimi kosztami modernizacji oraz ograniczeniami technologicznymi). Taką branżą jest z pewnością hutnictwo. Praca niniejsza zawiera pewne propozycje dotyczące optymalizacji planu produkcji (szeregowania zadań/zamówień) rzeczywistej walcowni oraz opis problemów tej dziedziny w warunkach rzeczywistych. Zwracamy szczególną uwagę na aspekty niepewności. Są one kluczem do uzyskania modelu dobrze odwzorowującego rzeczywistość, w której bardzo często spotykamy się z informacją nieprecyzyjną lub wręcz niepełną. Taki stan rzeczy jest zwykle spowodowany czynnikiem ludzkim (udział człowieka w łańcuchu czynności technologicznych) oraz sposobem opisu parametrów (wydajności) poszczególnych urządzeń. Parametry te z zasady podawane są jako przedział bądź wartość średnia oraz odchylenia od niej. Należy także wziąć pod uwagę pewien margines czasu związany z możliwymi awariami, który nie jest jednak możliwy do ścisłego oszacowania oraz czasu potrzebnego na bieżące konserwacje.

Szeregowanie zadań (harmonogramowanie) w ogólnej formie nie jest nowym zagadnieniem. Trudno znaleźć początki zainteresowania tym zagadnieniem. Nieformalne metody oparte na intuicji i doświadczeniu funkcjonują od momentu, kiedy pojawił się problem z zarządzaniem i współdzieleniem zasobów. Formalne

modele oparte na diagramach Gantta pojawiły się już podczas I Wojny Światowej. Dużym krokiem było opracowanie metody ścieżki krytycznej (CPM) w latach pięćdziesiątych. Zagadnienie szeregowania na jednej maszynie zostało opisane także w tym okresie. Jednymi z pierwszych, którzy zajmowali się problemem szeregowania byli J.R. Jackson i E.R. Smith. Problem szeregowania zadań w ujęciu deterministycznym może być określony w sposób następujący [10]. Jest dany zbiór zadań (jobs), z których każde jest skończoną sekwencją operacji uporządkowaną według pewnych ograniczeń kolejnościowych. Każda operacja jest wykonywana w ciągu określonego odcinka czasu wyłącznie przez jedną konkretną maszynę ze skończonego zbioru maszyn. Przy zachowaniu tych ograniczeń należy wyszukać optymalną kolejność wykonywania zadań.

Problemem, który chcemy uwypuklić w niniejszej pracy jest tzw. zagadnienie kolejnościowe bez postojów (no-wait flow shop lub continuous flow shop) ze względu na dziedzinę zastosowania opisywanych przez nas zagadnień. W przemyśle stalowym, w trakcie całego procesu produkcyjnego odpowiednia temperatura surowca bądź półproduktów często jest niezwykle istotnym wymogiem technologicznym. Jego dotrzymanie jest często kluczem do otrzymywania produktów o wysokiej jakości oraz powoduje, że produkcja staje się opłacalna (brak konieczności ponownego podgrzewania). W naszym konkretnym przykładzie opisanym w rozdziale 5 – podczas walcowania na gorąco, wymóg ten jest pierwszorzędny.

W literaturze jest opisanych wiele sposobów rozwiązywania problemu szeregowania zadań. Żywa dyskusja na ten temat jest spowodowana tym, że istnieje dosyć dużo modyfikacji i stopni uszczegóławiania problemu oryginalnego problemu oraz większość z nich nie doczekała się rozwiązania w czasie wielomianowym – należą do klasy zadań NP-trudnych. W [20] można znaleźć opisy wielu postaci problemu szeregowania deterministycznego włącznie z klasycznymi metodami ich rozwiązywania. W pracy [16] można znaleźć przegląd zastosowań algorytmów ewolucyjnych do wspomaganie decyzji planistycznych, w szczególności praca ta zawiera omówienie różnych sposobów reprezentacji – kodowania problemu w chromosom. W [17] przedstawiono technikę ulepszającą metodę lokalnego przeszukiwania będącą złożeniem technik: TS oraz Algorytmu Genetycznego (do dywersyfikacji przeszukiwania Tabu Serach - TS). W pracach [13]-[15] można znaleźć propozycje harmonogramowania w systemach czasu rzeczywistego z niepewnymi parametrami opisanymi przy pomocy liczb rozmytych. Istnieje wiele algorytmów stosowanych do rozwiązywania problemów szeregowania zadań (harmonogramowania), które można podzielić na dwie główne grupy: optymalizacyjne (dokładne) oraz aproksymacyjne (przybliżone). Pierwsza grupa to algorytmy gwarantujące znalezienie rozwiązania optymalnego. Potrzebują one bardzo dużo zasobów jak również czasu. Z praktycznego punktu widzenia, przy problemach większej skali stosuje się wyłącznie techniki aproksymacyjne, które nie gwarantują znalezienia optimum, ale wymagają mniej zasobów i są szybsze. Do tej grupy metod należy zaliczyć np. symulowane wyżarzanie (*Simulated Annealing*) [8] oraz Tabu Search [6]. Zbiór dopuszczalnych harmonogramów może być traktowany jako przestrzeń posiadająca wiele ekstremów lokalnych (ze względu na optymalizowaną funkcję celu). Główną trudnością podczas procesu przeszukiwania optimum w tego typu przestrzeni rozwiązań jest możliwość „utknięcia” w jednym z ekstremów lokalnych. Ciekawym sposobem zapobiegania temu zjawisku jest stosowanie mechanizmu dywersyfikacji bazującego na użyciu pamięci długo lub krótko-terminowej do zapamiętywania ruchów

prowadzących do lokalnych ekstremów i usunięcia ich z sąsiedztwa które jest przeszukiwane w danej iteracji. Mechanizm ten oraz kilka innych rozwiązań polepszających proces przeszukiwania jest zawarty w metodzie Tabu Search [6], której opis jest zawarty w rozdziale 4.

Praca jest zorganizowana według następującego porządku. Rozdział 2 w sposób formalny opisuje problem harmonogramowania. W rozdziale 3 przedstawiono najczęściej stosowany sposób prezentacji harmonogramu – graf dysjunkcyjny. W rozdziale 4 można znaleźć opis efektywnej metody lokalnych poszukiwań – Tabu Search. Rozdział 5 zawiera analizę problemu harmonogramowania rzeczywistej walcowni. Podsumowanie oraz perspektywy dalszych prac opisano w rozdziale 6.

2. Formalny opis problemu

Problem w sposób formalny może być opisany w sposób przedstawiony w [17]. Dany jest skończony zbiór n zadań (jobs) $\{J_i\}_{i=1}^n$, które muszą być wykonane przez skończone m maszyn ze skończonego zbioru $\{M_j\}_{j=1}^m$. Każde zadanie J_i jest sekwencją $n(i)$ operacji $\{w_{ik}\}_{k=1}^{n(i)}$, których uporządkowanie (kolejność) określa pewien zbiór ograniczeń opisywanych najczęściej przy pomocy grafu (w rozdziale 3 jest zawarty opis tej reprezentacji). Każda operacja jest wykonywana tylko przez jedną, określoną maszynę w ciągu wyspecyfikowanego przedziału czasu. Rozpoczęta operacja nie może być przerwana (*non pre-emption*). Czas potrzebny do wykonania wszystkich operacji danego harmonogramu jest określany jako makespan - C_{\max} .

Definicja klasycznego problemu szeregowania to znalezienie harmonogramu (*schedule*), który spełniałby następujące wymagania:

$$C_{\max}^* = \min_{i \in SolSpace} (C_{\max}(i)) \quad (1)$$

gdzie: $C_{\max}(i)$ - makespan i -tego harmonogramu; $SolSpace$ – przestrzeń rozwiązań (wszystkie dopuszczalne harmonogramy).

Dodatkowo należy wprowadzić założenia, które przybliżają problem do warunków rzeczywistych. Na przykład sekwencje operacji(zadania) mogą mieć różną wielkość dochodu uzyskanego z ich realizacji. De facto, na tą wielkość mają wpływ zarówno składniki dodatnie - zyski, jak również ujemne - koszty. Często zdarzają się sytuacje, w których realizacja zadania (zamówienia) wymagającego więcej czasu jest mniej opłacalna (w sensie doraźnego zysku finansowego) od realizacji zadania teoretycznie mniejszego. Z kolei realizacja zadania mniejszego może nieść duże ryzyko w postaci np. destrukcyjnego wpływu na park maszynowy. W tej sytuacji mamy do czynienia z kryteriami przeciwstawnymi, a problem szeregowania staje się po części problemem optymalizacji wielokryterialnej. Wzięcie pod uwagę ważniejszych kryteriów (bez zbytniego uszczegóławiania, które niesie ze sobą ryzyko rozmycia istoty problemu) – ich agregacja do kryterium globalnego daje możliwość podjęcia trafnej decyzji, przy minimalizacji ryzyka. Problem w ten sposób przedstawiony nosi miano szeregowania wielokryterialnego (*multicriteria scheduling*) [8]. Może być on modelowany na trzy różne sposoby. Jeśli kryteria są jednakowo ważne, generuje się wszystkie efektywne rozwiązania problemu, a następnie przy pomocy wielo-atrybutowych metod decyzyjnych dokonuje się

kompromisu między nimi. Jeśli kryteria mają różne wagi, możemy zdefiniować funkcję celu jako sumę ważonych funkcji celu poszczególnych kryteriów i przekształcić problem do szeregowania jednokryterialnego. Jeśli istnieje hierarchia poziomów priorytetu kryteriów, możemy w pierwszym rzędzie rozwiązać problem dla kryterium pierwszego priorytetu, ignorując inne kryteria a następnie rozwiązać to samo zadanie dla kryterium drugiego priorytetu przy ograniczeniu, że optymalne rozwiązanie kryterium pierwszego priorytetu się nie zmieni. Procedura ta jest kontynuowana do momentu rozwiązania problemu dla funkcji celu zbudowanej w oparciu o kryterium ostatniego priorytetu, a optymalne rozwiązania dla pozostałych kryteriów staną się ograniczeniami. W naszym przypadku preferowanym jest sposób drugi, który zapewnia możliwość zastosowania opracowanych już i sprawdzonych algorytmów dla problemu jednokryterialnego.

W pracach czysto teoretycznych można zaobserwować tendencję do układania harmonogramów wg różnych prostych reguł priorytetowania (np. LPT, SPT, EDF [3]). Z praktycznego punktu widzenia reguły te warto rozważać przy projektowaniu np. schedulera w jądrze systemu operacyjnego, gdzie można dopuścić do wywłaszczania, dynamicznego przydzielania priorytetu bądź odrzucania zadań w trakcie wykonywania. Warunki rzeczywiste (przemysłowe) nie tolerują takich zachowań – przy braku stabilności, przez źle ułożony harmonogram można stracić olbrzymie środki finansowe.

Spróbujmy, zatem przeformułować oryginalny problem (1), do takiego, który obejmowałby kilka aspektów(kryteriów) oceny korzyści oraz stopnia ryzyka związanego z realizacją danego zadania:

$$C_{Glob}^* = \max_{i \in SolSpace} D(C_0(i), C_1(i), \dots, C_n(i)) \quad (2)$$

gdzie: C_{Glob}^* - wartość optimum funkcji celu zagregowanej przez operację D

z kryteriów lokalnych; i -identyfikuje harmonogram z przestrzeni dopuszczalnych rozwiązań;

$C_j(i)$ -lokalne kryterium j , i -tego harmonogramu.

Przykładowymi kryteriami lokalnymi mogą być : wpływ finansowy związany z realizacją danego harmonogramu, kary umowne za niedotrzymanie terminów, koszty związane z eksploatacją maszyn w wyniku realizacji harmonogramu, koszt magazynowania produktów zbyt wcześnie wykonanych.

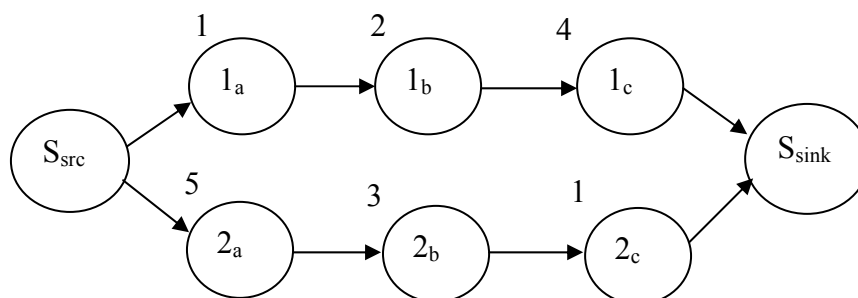
Należy dodać, że operację D można zrealizować na co najmniej kilka sposobów. Jeżeli analizowany przypadek jest opisany przy pomocy liczb rzeczywistych, a kryteria to różne funkcje celu, to operację D należy rozumieć jako sumę ważonych funkcji celu dla kryteriów lokalnych C_i . W przypadku, gdy posługujemy się pojęciem funkcji przynależności [24] do formalizacji poszczególnych kryteriów lokalnych, operacja D będzie jedną z t -norm [18]. Należy przy tym stwierdzić, że nie ma uniwersalnego sposobu agregacji – wyboru należy dokonać w zależności od dziedziny rozwiązywanego problemu.

3. Reprezentacja harmonogramu w postaci grafu

Według autorów pracy [17], każdy typ problemu harmonogramowania (JSP) może być przedstawiony w postaci dysjunkcyjnego grafu ważonego w węzłach (wierzchołkach). Niech $G = \{N, A \cup E\}$ będzie takim grafem. N – zbiór wierzchołków, A – zbiór łuków (krawędzi)

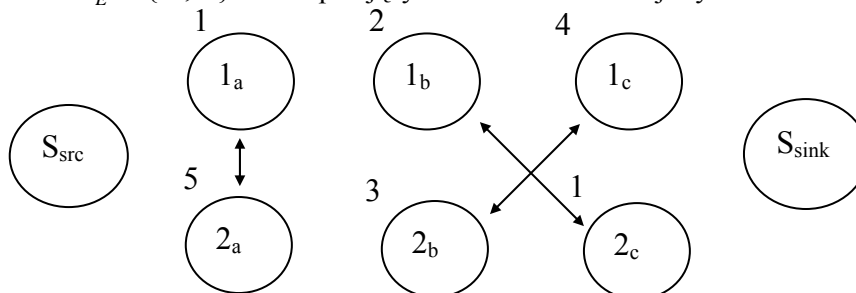
koniunkcyjnych, E – zbiór łuków dysjunkcyjnych. Każda operacja w_{xy} reprezentowana przez wierzchołek grafu jest wykonywana przez daną maszynę w ciągu czasu określonego przez wagę wierzchołka który ją reprezentuje. Dodatkowo wprowadzono dwa pozorne wierzchołki(operacje): źródło - S_{src} (bezpośredni poprzednik pierwszej operacji każdego zadania) oraz ujście - S_{sink} (bezpośredni następnik ostatniej operacji każdego zadania), których wagi równe są zero. Ograniczenia kolejnościowe między operacjami są reprezentowane przez łuki koniunkcyjne ze zbioru A . Dla każdej operacji w_{ik} istnieje łuk prowadzący od niej do operacji bezpośrednio następującej po niej - $w_{i(k+1)}$. Dla ostatniej operacji każdego zadania jej następnikiem jest S_{sink} . Dodatkowo istnieją łuki koniunkcyjne prowadzące od pozornej operacji źródła S_{src} do pierwszych operacji każdego z zadań. Łuk operacji w_{ab} do operacji w_{cd} reprezentuje wymóg, by operacja w_{ab} została zakończona przed rozpoczęciem operacji w_{cd} .

W celu zilustrowania tego typu opisu problemu harmonogramowania posłużmy się prostym przykładem. Nasz problem dotyczy dwóch zadań, z których każde składa się z trzech operacji. Poszczególne operacje każdego zadania przetwarzane są kolejno przez trzy różne maszyny. Zadanie J_1 składa się z następujących operacji $\{1a, 1b, 1c\}$, zadanie J_2 to zbiór $\{2a, 2b, 2c\}$. Zbiór maszyn $M = \{M_1, M_2, M_3\}$, gdzie poszczególne maszyny wykonują następujące operacje: $M_1 : \{1b, 2c\}$, $M_2 : \{1a, 2a\}$, $M_3 : \{1c, 2b\}$. Graf $G_A = \{N, A\} \subset G$ przykładowego problemu jest przedstawiony na rysunku 1.



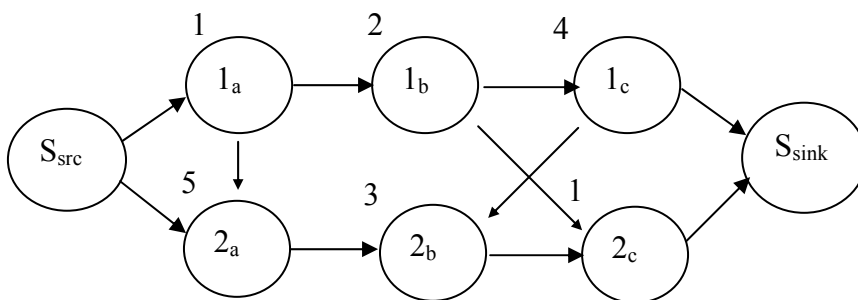
Rys.1. Graf G_A przykładowego problemu harmonogramowania (liczby 1,2,4,.. przy wierzchołkach oznaczają czasy realizacji poszczególnych operacji).

Łuki dysjunkcyjne ze zbioru E opisują przyporządkowanie operacji do konkretnych maszyn ze zbioru M . Każda para operacji w_{ab} i w_{cd} , które są wykonywane przez tą samą maszynę jest połączona łukiem dysjunkcyjnym. Każdy z łuków w E oznacza wymaganie wykonania operacji w_{ab} przed operacją w_{cd} lub w kolejności odwrotnej. Graf $G_E = \{N, E\} \subset G$ opisujący te zależności ilustruje rysunek 2.

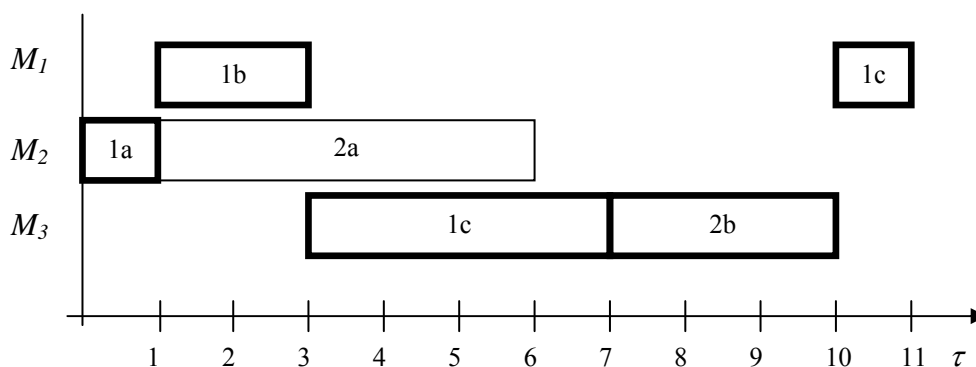


Rys.2. Graf G_E przykładowego problemu harmonogramowania.

Jeśli orientacja łuku między operacjami jest zaznaczona, determinuje ona kolejność przetwarzania tych operacji. Określenie kierunku każdego łuku dysjunkcyjnego w zbiorze E tak, by graf S osiągnięty z grafu początkowego G był acykliczny oznacza osiągnięcie rozwiązania dopuszczalnego. Graf S reprezentujący dopuszczalny harmonogram jest przedstawiony na rysunku 3, gdzie przy poszczególnych maszynach można zaobserwować następujące kolejności przetwarzania: $M1 : \{1b \rightarrow 2c\}$, $M2 : \{1a \rightarrow 2a\}$, $M3 : \{1c \rightarrow 2b\}$.



Rys.3. Graf S – dopuszczalne rozwiązanie przykładowego problemu.



Rys.4. Wykres Gantta harmonogramu z rysunku 3.

Makespan harmonogramu $C_{\max}(S)$ jest równy najdłuższej drodze od źródła S_{src} do ujścia S_{sink} w grafie S . Droga jest określana jako ścieżka krytyczna $C(S)$, a operacje które są w niej zawarte to operacje krytyczne. Na rysunku 4 można zaobserwować, że *makespan* przykładowego harmonogramu jest równy 11, ponieważ najdłuższa ważona ścieżka w grafie przechodzi przez operacje: 1a, 1b, 1c, 2b, 1c, dla których suma czasów wykonania równa się 11. Zaznaczono to także pogrubieniem na wykresie Gantta.

Problem harmonogramowania (szeregowania zadań) może być przedstawiony w tym kontekście jako zadanie orientacji (zaznaczania kierunku) wszystkich łuków dysjunkcyjnych tak, by graf wynikowy był acykliczny oraz jego ścieżka krytyczna była najkrótsza z możliwych.

4. Tabu Search

Wyszukiwanie najkrótszych ścieżek krytycznych w grafie reprezentującym problem jest zadaniem typowo kombinatorycznym. Trudno doszukiwać się w przestrzeni rozwiązań zależności funkcyjnych,

umożliwiających stosowanie metod analitycznych. Istnieją dwa zasadnicze kierunki, którymi można iść, by rozwiązać problem tej klasy. Pierwszym jest wygenerowanie wszystkich możliwych rozwiązań i kolejne porównywanie ich wszystkich. Mamy wtedy gwarancję osiągalności optimum. Niestety problem jest typowo kombinatoryczny i rozrasta się także kombinatorycznie. Ten sposób rozwiązywania jest użyteczny przy bardzo małych problemach. Przy problemach większej skali należy skorzystać z drugiego sposobu – zastosować jedną z technik zrandomizowanych. Charakteryzują się one tym, że przeszukiwanie przestrzeni rozwiązań jest ukierunkowane procesem losowania punktu, którego sąsiedztwo (w zależności od rozmiaru zadania jego rozmiar jest mniejszy bądź równy przestrzeni globalnej) stanowi obszar poszukiwań. Po przeszukaniu sąsiedztwa danego punktu, losowany jest następny punkt i proces powtarza się. Iteracje przerywane są po osiągnięciu pewnych warunków zatrzymania, które muszą być starannie określone, ponieważ gdy proces skończy się zbyt wcześnie rozwiązanie może być odległe od optimum. Sytuacja może wyglądać także diametralnie inaczej: jeśli warunki zatrzymania nie mogą zostać spełnione, po osiągnięciu optimum wyszukiwanie będzie trwało nadal w nieskończonej pętli.

TS to jedna z lokalnych metod poszukiwania (*local search method*) przeznaczona do rozwiązywania trudnych problemów optymalizacyjnych. Jest ona heurystyką (nie ma dowodów potwierdzających jej skuteczność z matematycznego punktu widzenia), a dokładniej meta-heurystyką zbudowaną na bazie kilku innych metod. Istnieje wiele rodzajów jej implementacji. Poniżej opiszemy wariant podstawowy zaproponowany przez Glovera [6]. TS może być postrzegane jako proces iteracyjny, w czasie którego przeszukiwana jest przestrzeń rozwiązań przez wielokrotne wykonywanie ruchów z jednego z rozwiązań p do innego p' zlokalizowanego w jego sąsiedztwie $N(p)$. Ruchy te wykonywane są po to by ostatecznie osiągnąć dobre rozwiązanie przez szacowanie wartości optymalizowanej funkcji celu $F(p)$. Jednak wartość funkcji $F(p')$ niekoniecznie musi wzrastać w każdej iteracji. Poszukiwanie kończy się po osiągnięciu warunków zatrzymania.

Jedną z głównych idei TS stanowi użycie pamięci krótko-, bądź długo terminowej - listy tabu (lista ruchów zabronionych) do przechowywania w niej pewnych ruchów w ciągu określonego odcinka czasu. W każdej iteracji TS, dany ruch zostanie zakwalifikowany do wpisania na listę tabu, jeśli jest on wybrany do prowadzenia procesu przeszukiwania (prowadzi do harmonogramu dopuszczalnego). Ruch ten nie zostanie wybrany w ciągu pewnej liczby iteracji - rozmiaru listy tabu. Kiedy lista zapełnia się, ruch wpisany najwcześniej jest z niej usuwany, a dopisywany bieżący. Dzięki właściwej konstrukcji listy tabu TS może skutecznie przeciwdziałać powstawaniu cykli (utknięcie w ekstremum lokalnym). Może zdarzyć się jednak, że ruch który normalnie (w najprostszej wersji TS) jest zakwalifikowany jako tabu dobrze byłoby dalej przetwarzać – może prowadzić w sposób pośredni do lepszych rozwiązań. Warto zatem wykorzystać tzw. kryterium aspiracji [6], poprawiające w tym względzie klasyczny algorytm TS. Ruch spełniający kryterium aspiracji jest usuwany z listy tabu i w dalszym ciągu bierze udział w procesie przeszukiwania. Oprócz kryterium aspiracji oraz zdolności do dywersyfikacji przestrzeni rozwiązań (poprzez listę tabu), niektóre implementacje TS posiadają także mechanizm intensyfikacji, który umożliwia zagęszczenie przeszukiwania w punktach spełniających pewne wymagania. Dodatkowo istnieją także implementacje adaptacyjnej metody TS – *Reactive Tabu Search* [2], w których TS potrafi dostroić niektóre parametry do przebiegu procesu

przeszukiwania. Ogólną postać algorytmu Tabu Search przedstawiono na rysunku 5. Należy jednak dodać, że metoda TS jest silnie związana z konkretnym problemem. Nie da się uniknąć pewnej pracy związanej z jej „dostrojeniem” do własnych potrzeb.

```

GenerateInitSol( $S_0$ ); //rozwiązanie początkowe
TL=0; //zerowanie listy tabu
Initialization(); //inicjalizacja: rozmiar listy tabu, warunki
// zatrzymania, poziom aspiracji
while( ! TermConditions() )
{
    N=Neighbourhood( $S_0$ ); //generacja sąsiedztwa rozwiązania
// początkowego
    if( ||M|| )
    {
        FinalSolution= $S_0$ ;
        return;
    }
    for( $x \in N$ )
    {
        if( $x \notin TL$ )
        {
            if (  $c(x) < c(S_0)$  ) //porównywanie funkcji celu
            {
                M = x;
            }
            else
            {
                if( aspiration( $x$ ) > max_asp ) //poziom aspiracji
                {
                    M = x;
                }
            }
        }
    }
    S0=M;
    Refresh(TL);
    FinalSolution= $S_0$ ;
}

```

Rys. 5. Ogólna postać algorytmu *Tabu Search*.

5. Przykład analizy rzeczywistego problemu harmonogramowania.

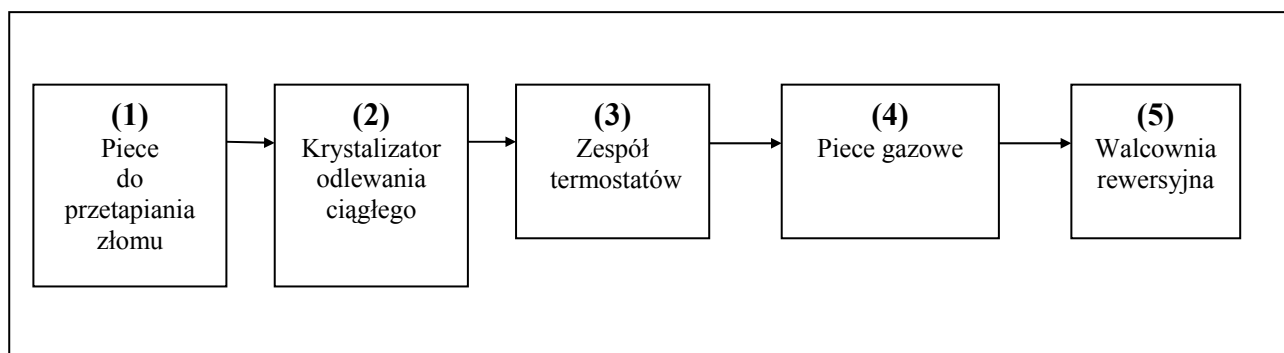
W literaturze jest opisanych wiele problemów testowych (*benchmark*) np. w [10], trudno jednak znaleźć przykład problemu rzeczywistego. Wobec tego postanowiliśmy przedstawić analizę problemu harmonogramowania w rzeczywistym zakładzie produkcyjnym – „walcowni na gorąco” w Białoruskiej Hucie BMZ w Żłobinie (Białoruś). Nie podajemy tu wyników w postaci wykresów przedstawiających odległość rozwiązania w danej iteracji od rozwiązania optymalnego (zbieżności), ponieważ naszym celem nie jest udowodnienie wyższości danej metody nad inną. W wielu pracach udowodniono, że Tabu Search jest pełnowartościową techniką – szczególnie przy przeszukiwaniu przestrzeni rozwiązań generowanych w sposób kombinatoryczny (bez zależności funkcyjnych). Uważamy, że pokazanie sposobu analizy rzeczywistego problemu, który nie należy do klasy problemów szablonowych będzie o wiele bardziej

korzystne niż powielanie dobrze znanych aspektów harmonogramowania. Skupiamy się jedynie na problemie uwzględnienia niepewności parametrów procesu produkcyjnego mającej często decydujący wpływ na sukces planowania.

5.1. Proces technologiczny

Ciąg technologiczny, który rozpatrujemy pod kątem harmonogramowania to zespół urządzeń pracujących wg schematu 24h x 30(31)dni, tzn. w sposób ciągły. Jest on przedstawiony w sposób uproszczony na rysunku 6.

Blok (1) to zespół pieców odpowiedzialnych za przetopienie złomu metali na surówkę o odpowiednim składzie chemicznym. Kontrola parametrów jest w pełni zautomatyzowana, a poziom błędów na poziomie ułamków procenta. Surówka jest odlewana w bloku (2) do postaci sztab. Sztaby są przekazywane do zespołu termostatów (3). Podczas walcowania sztaby muszą mieć odpowiednią temperaturę (odchylenia grożą pogorszeniem jakości produktów – spękaniami powierzchni). Jeśli temperatura sztab jest zbyt niska, muszą być podgrzane w bloku (4). Blok (5) to walec rewersyjny (powrotny) - kluczowy punkt ciągu. Należy dodać, że transport między blokami jest realizowany bez użycia dodatkowej energii (grawitacyjny), co w sposób znaczny obniża koszty.



Rys. 6. Schemat blokowy ciągu technologicznego

5.2. Parametry i dane

Wprowadźmy kilka parametrów opisujących zagadnienie:

- $P_{max} [T/h]$ – teoretyczna wydajność produkcyjna;
- $K = \frac{P_0}{P_N}$;

gdzie: K - współczynnik względnej zmiany kształtu; P_0 – obwód sztaby wejściowej; P_N – obwód produktu gotowego;

- $P [T/h]$ - wydajność walcowni (stała dla każdego K);
- $P_f [T/h]$ – faktyczna (rzeczywista) wydajność produkcyjna;
- $P_f = P * T_c$;

gdzie: T_c – czas pracy walcowni (czas kiedy metal jest w walcach + przerwa między przejściami metalu w walcu (rewersja) + przerwy między odrębnymi kawałkami (sztabami))

$$T_c = T_k - T_{kp} - T_{ppp} - T_{pp} - T_{cm}$$

gdzie: T_k - czas pracy Walcowni w miesiącu (24 x 30 (31));

T_{kp} – sumaryczny czas zaplanowanych remontów (np. 4h/mies.);

T_{ppp} – czas remontów prewencyjnych (np. 45h/mies);

T_{pp} – bieżący czas przestoju (wg przepisów = 0, *de facto* liczba = [0,10] - awarie);

T_{cm} – czas zmiany kalibrów (zmiana kształtu) i walców .

Tabela1. Wydajności walcowni oraz współczynnik K w zależności od profilu produktu.

	Profil wyjściowy	K	P
1	□ 125	2.8	68
2	□ 100	3.5	63.1
3	φ 150	2.98	79
4	φ 100	3.54	51.8
5	φ 90	3.92	42.6
6	φ 80	4.4	32.2

Jak można zauważyć w tabeli 1, wydajność walcowni zmienia się w zależności od profilu produktu wyjściowego. Spowodowane jest to różną liczbą rewersji potrzebnych do osiągnięcia produktu (pręta) o żądanym profilu. Ze względu na olbrzymie koszty walca, urządzenie to pracuje na zmianę z różnymi wymiennymi nakładkami profilowymi(kalibrami). Sytuacja ta narzuca konieczność zmian kalibrów, co z kolei pochłania dodatkowy czas potrzebny na tą czynność. Czasy zmian prezentuje tablica 2.

Tabela 2. Czasy zmian kalibrów przy zmianie profilu produktu wyjściowego.

Rodzaj zmiany	Czas potrzebny na zmianę[h]
{□ 100 ∨ φ 150 ∨ φ 100 ∨ φ 90 ∨ φ 80} ⇒ □ 125	1
{□ 125 ∨ φ 150 ∨ φ 100 ∨ φ 90 ∨ φ 80} ⇒ □ 100; {□ 125 ∨ □ 100 ∨ φ 100 ∨ φ 90 ∨ φ 80} ⇒ φ 150; {□ 125 ∨ □ 100 ∨ φ 150 ∨ φ 90 ∨ φ 80} ⇒ φ 100; {□ 125 ∨ □ 100 ∨ φ 150 ∨ φ 100 ∨ φ 80} ⇒ φ 90; {□ 125 ∨ □ 100 ∨ φ 150 ∨ φ 100 ∨ φ 90} ⇒ φ 80;	1.5

Nakładki walców (kalibry) podlegają naturalnemu zużyciu. Należy ten fakt wziąć pod uwagę. Czas potrzebny na zmianę zużytego kalibru to **0.66 h**. Przyjmuje się, że średnie miesięczne zużycie kalibrów kształtuje się na poziomie 15-20 szt. Zużycie to zależy głównie od wielkości produkcji.

Innych parametrów opisujących pozostałe bloki systemu nie umieszczamy, zakładając że nie ograniczają one w żaden sposób ogólnej wydajności całego ciągu, dlatego że dzisiaj wąskim gardłem całego systemu produkcyjnego jest walcownia.

5.3 Krytyczne sekcje systemu.

Łatwo można wywnioskować z powyższego opisu, że sekcją krytyczną systemu – jego wąskim gardłem jest walcownia oraz okoliczności zmian kalibrów. Ze względu na charakter procesu technologicznego, problem harmonogramowania w naszym przypadku można sprowadzić do zagadnienia przepływowego z ograniczeniem dotyczącym ciągłości (reżimy temperaturowe). W literaturze problem ten nosi miano *continuous flow shop*. Zachowanie stałej marszruty (*routing*) zadań, sekwencyjność ciągu technologicznego (linia bez rozgałęzień i cykli) oraz istnienie maszyny wyraźnie determinującej ogólną wydajność systemu (wąskie gardło) dodatkowo predestynują przedstawiony problem do miana *harmonogramowania jednomaszynowego* (*one-machine scheduling*). Jednak nie można użyć metod dokładnie zaczerpniętych z literatury dla tej klasy problemów – przyczyną są zmiany kalibrów, które komplikują problem.

5.4. Niepewność systemu

Jak można zauważyć w punkcie 5.2, niektóre z parametrów nie są określone w sposób deterministyczny. Parametry takie jak: T_{cm} , T_{pp} , T_{ppp} , T_{kp} nie mogą być określone jako z góry znane wielkości. Weźmy dla przykładu bieżący czas przestoju T_{pp} : kto podejmie się zaplanować z miesięcznym wyprzedzeniem ile będzie awarii oraz ile czasu będzie trwało ich usuwanie? Podobnie jest z określeniem dokładnego czasu zmiany kalibru, bądź walca T_{cm} : biorąc pod uwagę to, że zmiany dokonują ludzie oraz położenie walca/kalibru w magazynie może być w pewien sposób przypadkowe, wartości czasów zmian kalibrów z tabeli 2 oraz czasu wymiany zużytego walca należy traktować jako pewne przybliżenie. Powyższe parametry powodują automatycznie rozmycie wielkości T_c , ta z kolei rozmywa P_f . Zatem jedynym pozostałym parametrem, który można uważać za deterministyczny jest P_{max} oraz w pewnym zakresie także K (ze względu na bardzo dużą dokładność w zgodności z normami ISO9002). Zamiast w ciszy pomijać aspekt niepewności, lepiej wziąć go pod uwagę i wprowadzić opisywanie parametrów w postaci *interwałów* lub *liczb rozmytych*. Gwarantuje to otrzymanie modelu planowania maksymalnie zbliżonego do rzeczywistości w porównaniu z modelami czysto-deterministycznymi.

5.5. Określenie funkcji celu.

Klasyczną funkcją celu w harmonogramowaniu zadań jest minimalizacja czasu wykonania danego harmonogramu (wzór 1). Funkcja taka mogła by być użyteczna w przypadku klasycznej linii produkcyjnej. Nasz rzeczywisty przykład niesie także rzeczywiste problemy: harmonogramy ułożone w różny sposób (różna kolejność zadań) mogą mieć różne skutki finansowe, nie tylko czas wykonania (to jest oczywiste). Zamiast jednego, należy znaleźć kilka kryteriów dobrze opisujących problem. Nie należy także przesadzać ze zbyt dużą ich liczbą – tylko takie które mają *istotny* wpływ finansowy (negatywny bądź pozytywny,

bezpośredni bądź pośredni) na bilans realizacji zadań harmonogramu. Wzór 2 w sposób ogólny oddaje zasadę budowy funkcji celu w rzeczywistych warunkach.

W naszym przypadku można zauważyć, że najsłabszym punktem (wąskim gardłem) systemu jest walec i zmiany jego kalibrów. Dla przykładu jeśli system wytwarza produkty o profilu 3 i zmieniamy profil na 2, to trwa to 1.5 h. W tym czasie jesteśmy w stanie wyprodukować ≈ 50 ton (w najgorszym przypadku). Jedna zmiana w ciągu dnia, w ciągu miesiąca daje ≈ 1500 ton. Obrazuje to skalę problemu w sposób przejrzysty. Najważniejszym kryterium w naszym przypadku będzie zatem suma czasów zmian kalibrów oraz walców - $\sum T_{cm}$. Drugim kryterium będzie stopień dopasowania czasu realizacji zadania do przewidzianego w umowie $\sum T_{fit}$. Jak wiadomo, za zrealizowanie zamówienia po umówionym terminie przewidziane są kary finansowe za nie wywiązanie się z warunków umowy. Sytuacja przeciwna, tzn. realizacja zbyt wczesna zmusza do ponoszenia kosztów magazynowania. Najlepiej więc, gdy zadanie mieści się w terminie określonym w umowie.

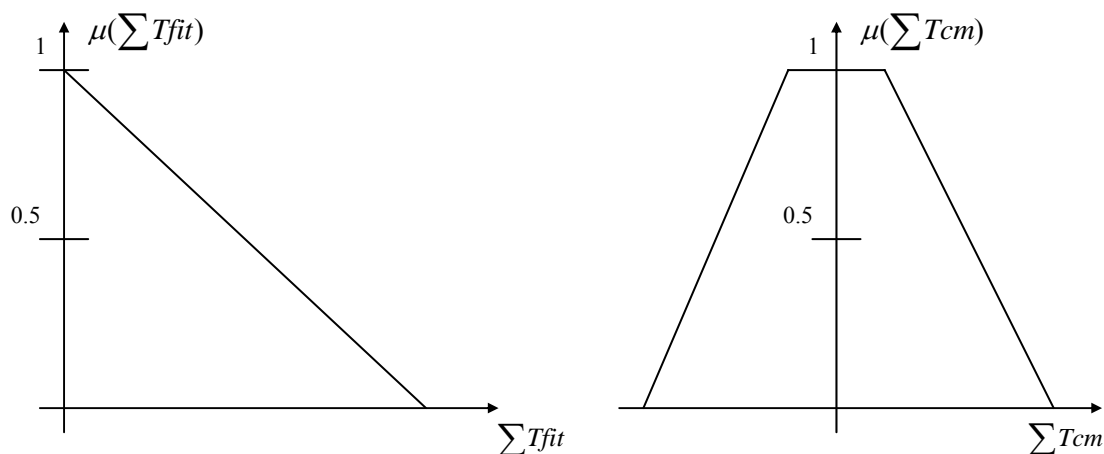
Dobrym sposobem rozwiązywania wielokryterialnych zadań decyzyjnych, do którego sprowadziliśmy problem przykładowy jest dobranie odpowiedniego kształtu funkcji użyteczności dla każdego z analizowanych kryteriów oraz określenie dla nich odpowiednich rang, jeśli kryteria te nie są równoważne. Dokładny opis tego postępowania można znaleźć w pracy [5]. Kształt funkcji przynależności dla przyjętych kryteriów przedstawia rysunek 7.

Funkcja celu dla naszego przykładu:

$$C^* \max = \max_{i \in SolSpace} D\{\mu(\sum T_{cm}(i)), \mu(T_{fit}(i))\}$$

przy ograniczeniu: $P_{\max} < 55\,000$ ton / miesiąc. (3)

gdzie: D jest jedną z metod agregacji kryteriów [5]: {maksymalnego pesymizmu, multiplikatywną, addytywną}.



Rys. 7. Kształty funkcji przynależności kryteriów użytych do budowy funkcji celu przykładowego problemu.

Mając określoną w ten sposób funkcję celu można bez większego problemu zastosować *Tabu Search* i przy jej pomocy porównywać poszczególne harmonogramy dopuszczalne, a w wyniku czego znaleźć optymalny lub zbliżony do optimum.

6. Podsumowanie

Problem harmonogramowania w warunkach rzeczywistych nie polega tylko na trudnościach typu kombinatorycznego. Jak przedstawiono w rozdziale 5, typowe kryterium – całkowity czas realizacji harmonogramu (*makespan*) nie koniecznie jest najbardziej trafnym. W warunkach rzeczywistych mogą istnieć jeszcze inne sytuacje, wymuszające stosowanie innych kryteriów pojedynczych lub zestawów kryteriów nie branych pod uwagę w dostępnej literaturze. Podejście proponowane w niniejszej pracy – przekształcenie klasycznego problemu harmonogramowania w problem decyzyjny udostępnia mechanizm łatwego rozszerzania modelu o kolejne kryteria bądź całkowitej wymiany zestawu kryteriów. Jest to bardzo skuteczne podejście. Jednak nie ukrywamy, że nie ma rozwiązań uniwersalnych i niezbędna jest dogłębna analiza konkretnego przypadku. Rzeczywistych problemów harmonogramowania nie da się uogólnić, są one silnie dziedzinowo-zależny.

W artykule przedstawiony jest opracowany schemat rozwiązania problemu optymalizowanego wielokryterialnego planowania produkcji rzeczywistej huty ze szczegółowym uwzględnieniem niepewności parametrów czasowych w realizacji metody *Tabu Search*.

Dalsze prace mają na celu dokładniejsze badania przedstawionego mechanizmu. Szczególnie w warunkach niepewności, kiedy parametry są opisane za pomocą liczb rozmytych. Przeprowadzone zostaną także wielokrotne testy, mające na celu optymalne „dostrojenie” *Tabu Search* oraz porównanie efektywności harmonogramów tworzonych przez doświadczonych planistów z tymi, otrzymanymi za pomocą systemu bazującego na opisywanych modelu .

Bibliografia

1. Адлер Ю.П., Маркова Е.В., Грановский Ю.В., Планирование эксперимента при поиске оптимальных условий, Наука, 1976, 279.
2. Battiti R. and Tecchiolli G., The reactive tabu search *ORSA Journal on Computing*, 1994, 6(2),126-140.
3. Cristofari M, Caron F, Tronci M. Dynamic scheduling approach in case of machine breakdown and preventive maintenance - 2000 Summer Computer Simulation Conf.
4. Dubois D., Fargier H., Prade H., Fuzzy constraint in job shop scheduling , Proceedings of the EURO XIII/OR conference, Glasgow, UK, July, 1994.
5. Dymowa L., Sewastjanow P.. The methodology of solution of joint problems of modeling, identification and multi-criterion optimization in quality control for metallurgical processes, *Informatyka w Technologii Materiałów*, Wydawnictwo Naukowe AKAPIT 2003, 1, 3 , 21-32.
6. Glover F., Tabu Search, part I, *ORSA J. Comput.* 1989, 1, 190-206.
7. Grabowski J., Pempera J., Sequencing of Jobs in some production system, *European Journal of Operational Research*, 2000, 125, 535-550.
8. Gupta J.N.D., Henning K. and Werner F., Local Search Heuristics for Two-Stage Flow Shop Problems with Secondary Croterion, *Otto-von-Guericke-Universität Magdeburg* 1999.

9. Huh W.T., Janakiraman G., Jackson P.L., Sawhney N., Minimizing flow time in cyclic schedules for identical jobs with acyclic precedence: the bottleneck lower bound, *Operation Research Letters* 2003, 31, 366-374.
10. Jain A.S., Meeran S., Deterministic job-shop scheduling: Past, present and future, *European Journal of Operational Research*, 1999, 113, 390-434.
11. Konno T., Ishii H., An open shop scheduling problem with fuzzy allowable time and fuzzy resource constraint, *Fuzzy Sets and Systems* 2000, 109, 141-147.
12. Lam S.S., Cai X., Single machine scheduling with nonlinear lateness cost functions and fuzzy due dates, *Nonlinear Analysis* 2002, 3, 307-316.
13. Litoiu M., Tadei R., Real time task scheduling allowing fuzzy deadlines, *European Journal of Operational Research* 1997, 100 475-481.
14. Litoiu M., Tadei R., Fuzzy scheduling with application to real time systems, *Fuzzy Sets and Systems* 2001, 121, 523-535.
15. Litoiu M., Tadei R., Real time task scheduling with fuzzy deadlines and processing times, *Fuzzy Sets and Systems* 2001, 117, 35-45.
16. Ławrynowicz A., Algorytmy ewolucyjne w zarządzaniu produkcją, *Informatyka Teoretyczna i Stosowana/Computer Science* 2003, 3(4), 149-168.
17. Murovec B., Suhel P., A repairing technique for the local search of the job shop problem, *European Journal of Operational Research* 2004, 153, 220-238.
18. Piegat A., Modelowanie i sterowanie rozmyte, *Akademicka Oficyna Wydawnicza EXIT*, Warszawa, 1999.
19. Slany W., Scheduling as a multiple criteria optimization problems, *Fuzzy Sets and Systems* 1996, 78, 197-222.
20. Sysło M., Deo N., Kowalik J., Algorytmy optymalizacji dyskretnej, *WN PWN*, Warszawa 1995.
21. Wang,J., A fuzzy project approach to minimize schedule risk for product development, *Fuzzy Sets and Systems* 2002, 127, 99-116.
22. Wang,J. A fuzzy robust scheduling approach for product development projects, *European Journal of Operational Research* 2004, 152, 180-194.
23. Youssef H., Adiche S. M. H., Evolutionary algorithms, simulated annealing and tabu search: a comparative study, *Engineering Applications of Artificial Intelligence* 2001, 14, 167-181.
24. L. A. Zadeh, Theory of fuzzy sets, Memo. No. UCB/ERL M77/1, University of California, Berkeley, CA, 1977.