PL ISSN 0209-1593

1 1984

prace naukowo-badawcze

Instytutu Maszyn Matematycznych

rok XXVI

Druk IMM zam. 9/85 nakł. 280 egz.

I n s t y t u t M a s z y n M a t e m a t y c z n y c h

p r a c e n a u k o w o - b a d a w c z e
I n s t y t u t u
M a s z y n
M a t e m a t y c z n y c h

Wojciech Mokrzycki: O pewnej metodzie dyskretyzacji algebraicznych krzywych przestrzennych na jednorodnych siatkach prostokątnych s.3

Andrzej Rowicki Nonpreemptive Scheduling for Two-Processor Systems p. 41

W a r s z a w a 1984

Adres redakcyjny: Instytut Maszyn Matematycznych
Branżowy Ośrodek INTE
02-078 Warszawa, ul. Krzywickiego 34
tel. 28-37-29 lub 21-84-41 w. 244

Prace naukowo-badawcze Instytutu Maszyn Matematycznych

O pewnej metodzie dyskretyzacji algebraicznych krzywych przestrzennych na jednorodnych siatkach prostokątnych

Wojciech MOKRZYCKI

WSTĘP

W komputerowej generacji obrazów przestrzennych, numerycznym sterowaniu obrabiarkami i procesami technicznymi oraz w innych dziedzinach nauki i techniki zachodzi potrzeba dyskretyzacji /aproksymacji bądź generacji/ krzywych i powierzchni w trójwymiarowej przestrzeni.

W szczególności w obrazowaniu przestrzennym brył krzywopowierzchniowych dyskretyzacja krzywych występuje pod co najmniej trzema postaciami: jako /wytyczanie i/ dyskretyzacja krawędzi przecinania się obrazowanych powierzchni, jako /wytyczanie i/ dyskretyzacja krawędzi przecinania się powierzchni z granicami pola obrazowania oraz jako /wyznaczanie i/ dyskretyzacja granic widoczności obszarów i brył w zagadnieniach eliminacji ukrytych linii i powierzchni.

Literatura na ten temat jest stosunkowo uboga, często szczątkowa bądź niepełna. Wzrasta natomiast zapotrzebowanie na metody dyskretyzacyjne. Zachodzi więc potrzeba podjęcia własnych badań w tym zakresie.

Spośród kilku znanych reprezentacji krzywych przestrzennych najpowszechniej stosowana jest reprezentacja parametryczna. Reprezentacja ta zapewnia przejrzysty zapis krzywej oraz prosty związek między dwu, trój- i wielowymiarowym przedstawieniem krzywej. Dyskretyzacje /aproksymacje/ krzywych parametrycznych są proste algorytmicznie i w niewielkim stopniu zależne od stopnia krzywej i liczby wymiarów jej przedstawienia. Sprzętowa realizacja dyskretyzacji /generacja krzywej dyskretnej/ prowadzi do dobrze znanej techniki CAR /cyfrowych analizatorów różniczkowych/. W tej technice proste i łatwe jest przejście z dwóch do trzech czy większej liczby wymiarów /powielenie liczby takiego samego typu układów całkujących/. Istotnymi wadami metody parametrycznej jest natomiast trudne do przewidzenia niebezpieczeństwo kumulacji błędów oraz zmienna szybkość generacji. Dla pewnych zastosowań powyższe wady czynią tę metodę nieprzydatną.

Mniej znaną metodą dyskretyzacji krzywych jest metoda nieparametryczna. Dyskretyzacja krzywych nieparametrycznych wymaga innej techniki, która jednak wydaje się być porównywalna w kosztach z techniką dyskretyzacji krzywych parametrycznych, przy tej samej złożoności krzywej. Również sterowanie prędkością generacji /szczególnie w układowej realizacji generacji/ nie jest problemem, jeśli zostało przewidziane w rozwiązaniu. Błędy dyskretyzacji dają się natomiast kontrolować. Przejście w przedstawieniu krzywej z dwóch do trzech wymiarów jest jednak złożone, wymaga opracowania algorytmów od nowa i prowadzi do złożonych rozwiązań.

Autor niniejszej pracy prowadził badania w zakresie dyskretyzacji algebraicznych krzywych dwuwymiarowych [11] . Sądzi więc, że algorytmy dyskretyzacyjne płaskich krzywych algebraicznych dadzą się wykorzystać do konstrukcji algorytmów dyskretyzacji krzywych przestrzennych i powierzchni krzywoliniowych algebraicznych i że dyskretyzacja ta będzie miała - podobnie jak w wypadku krzywych płaskich - wiele pozytywnych cech.

W opisanym przez Lahna i Królaka /1965 r./ komputerowym systemie do wspomagania projektowania części mechanicznych punkty krzywej przecinania się dwóch powierzchni kwadratowych wyznaczane są za pomocą dwóch nierówności. Nierówności te wynikają z równań przecinających się powierzchni, gdy obliczane są pierwiastki tych równań dla jednej ze zmiennych x, y lub z. Zakłada się wówczas, że dla każdego z równań zachodzi $\sqrt{\Delta} > 0$. Wyrażenia na $\sqrt{\Delta}$, zawierające pozostałe dwie zmienne, powinny być równocześnie spełnione. Z tego właśnie warunku wyliczone są pozostałe dwie współrzędne bieżącego punktu na krzywej przecięcia. Niestety autorzy publikacji nie opisują dokładnie zastosowanej przez siebie metody, stąd też niemożliwe jest określenie jej pozytywnych i negatywnych cech. Z opisu można jednak wywnioskować, że jest to metoda ogólnogeometryczna i nie wynika ona ze specyfiki problemu, do rozwiązania którego została wykorzystana.

Weiss /w 1966 r./ opisuje procedurę znajdywania krzywych przecinania się powierzchni kwadratowych zadanych równościami $F_1(x,y,z) = 0$, $F_2(x,y,z) = 0$. Kolejne punkty na krzywej przestrzennej /metodą punkt po punkcie/ wyznacza się w ten sposób, że dla kolejno zmienianej /o elementarny przyrost równy rozdzielczości urządzenia obrazującego/ jednej ze współrzędnych, np. współrzędnej z w przedziale obrazowania /widoczności/ $z_{min} \leqslant z \leqslant z_{max}$, rozwiązywane są równocześnie obydwa równania dla pozostałych dwóch współrzędnych /tzw. metoda płaszczyzn tnących/.

Metoda powyższa dostosowana jest częściowo do wymagań /dyskretności obrazowania/ urządzeń graficznych. Autor twierdzi jednak, że program wyliczający krzywe przecięć tą metodą jest długi i złożony. Problemem jest również długi czas obliczeń.

Comba /1968/ opisuje procedurę wyznaczania obszarów wspólnych przecinających się obiektów ograniczonych powierzchniami kwadratowymi opisanych równościami $F_i(x,y,z) = 0$. Procedura ta może być wykorzystana również do określenia krzywych przecinania się powierzchni kwadratowych. W tym celu wykorzystywana jest tzw. funkcja charakterystyczna /karna/ Q, która jest tak zdefiniowana, że przyjmuje wartości ujemne wewnątrz wspólnej części przecinających się obszarów oraz wartości dodatnie na zewnątrz tej części. Problemem jest znajdowania tych punktów, dla których Q przyjmuje wartości zerowe lub bliskie zeru. Autor nie podaje jednak algorytmu znajdowania tych punktów. Wspomina tylko, że wykorzystywana jest do tego metoda gradientów. Metoda ograniczona jest tylko do powierzchni wypukłych 2-stopnia.

Danielsson /1972 r./ opisuje szczegółowo algorytm dyskretyzacji krawędzi przecinania się dwóch powierzchni, który krawędź tę zamienia w ciąg jednostkowych przesunięć wzdłuż jednej z osi układu. Kryterium wykorzystywanym do określenia kierunku ruchu w tym algorytmie jest niezwiększanie się bezwzględnych wartości funkcji $F_1(x,y,z)$ i $F_2(x,y,z)$ opisujących przecinające się powierzchnie. Autor podaje wyrażenia logiczne wyznaczone na podstawie znaków funkcji F_1 i F_2 oraz ich pochodnych w bieżącym punkcie, wyznaczające ruchy wzdłuż poszczególnych osi. Wyrażenia te są jednak złożone. Przed

każdym elementarnym ruchem należy wykonać dużą liczbę obliczeń pomocniczych. Konieczne jest również pamiętanie i uaktualnianie w punkcie bieżącym tak wartości funkcji $F_1(x,y,z)$ i $F_2(x,y,z)$, jak i ich wszystkich niestałych pochodnych oraz składowych wektora kierunkowego stycznej do krzywej, który określa się dość złożonym wyrażeniem. Podstawowym mankamentem algorytmu jest bardzo duża "chropowatość" krzywej dyskretnej, spowodowana ograniczeniem kierunku jednostkowego ruchu do kierunku jednej z osi układu.

Woon i Freeman /1971 r./ wyznaczają ciąg przylegających i równo odległych punktów krzywej przecinania się powierzchni drugiego stopnia w sposób następujący: Definiują siatkę przestrzenną określając trzy zbiory równoległych płaszczyzn: $x_i = i\cdot\delta$, $y_i = i\cdot\delta$, $z_i = i\cdot\delta$, $i = \pm1, \pm2, \pm3, \ldots$, a δ jest założoną rozdzielczością. Następnie poszukują takich punktów krzywej, których wzajemna odległość nie przekracza zadanej wielkości /np. 2δ/. Zakładając, że δ jest małe w porównaniu z promieniem krzywizny krzywej, kierunek przesunięcia wzdłuż krzywej z punktu P_{k-1} do punktu P_k wykorzystywany jest do określenia kierunku ruchu następnego, tj. kierunku ruchu do punktu P_{k+1}. Jeśli składowa x poprzedniego przesunięcia jest dłuższa od składowych dwóch pozostałych współrzędnych, wówczas wyliczona jest składowa x_{k+1} następnego przesunięcia

$$x_{k+1} = x_k + \delta \cdot \mathrm{SGN}(x_k - x_{k-1}),$$

Wartości współrzędnych y_{k+1} oraz z_{k+1} otrzymuje się po rozwiązaniu układu równań $F_1(x,y,z) = 0$, $F_2(x,y,z) = 0$ przecinających się powierzchni, wstawiając do tych równań wartość współrzędnej x_{k+1} wyznaczoną we wcześniejszej operacji.

Mahl /1974 r./ opisuje algorytm znajdowania krawędzi przecinających się płatów powierzchni. W tej metodzie przecinające się powierzchnie są cięte zbiorem równoległych wzajemnie i względem jednej z osi układu płaszczyzn, zwanych płaszczyznami tnącymi. Przecięcia powierzchni z płaszczyznami tnącymi obliczane są indywidualnie dla każdej z płaszczyzn tnących. Odległości między płaszczyznami określają dokładność /moduł/ dyskretyzacji. Metoda Mahla jest ulepszoną nieco metodą Weissa.

Sabin /1974 r./ do obrazowania przecięć algebraicznych powierzchni stosuje zapis Cayley'a krzywej, w którym krzywa jest reprezentowana jako funkcja F dwóch punktów P_1 i P_2. $F(P_1,P_2) = 0$ wtedy i tylko wtedy, gdy prosta $\overline{P_1P_2}$ przecina krzywą. Zastępując pozycję oka punktem P_1, równanie $F(P_1,P_2) = 0$ staje się równaniem punktów płaszczyzny leżących na projekcji krzywej.

Za pomocą tego równania można przeprowadzać wszystkie obliczenia w przestrzeni dwuwymiarowej obrazu zamiast trójwymiarowej przestrzeni krzywej.

Levin /1976 i 1979 r./ opisuje metody wyprowadzania parametrycznych form krzywych będących przecięciami powierzchni algebraicznych kwadratowych, wykorzystując pewne matematyczne zależności tych powierzchni.

W tej metodzie pierwszą czynnością jest wyprowadzenie parametryzującej powierzchni. Powierzchnia ta zawiera krzywą przecięcia, a typ tej powierzchni określa rodzaj parametryzacji. Następnym krokiem jest transformacja wyjściowej przestrzeni OXYZ w przestrzeń kanoniczną OUVW, w której parametryzująca powierzchnia określona jest w kanonicznej postaci. Z kolei wyznaczona jest krzywa przecięcia w kanonicznej przestrzeni OUVW, której współczynniki są następnie transformowane z powrotem do przestrzeni wyjściowej OXYZ.

W niniejszej pracy proponuje się inną niż wyżej omówione metody wytyczania dyskretnej krzywej przestrzennej, będącej krawędzią przecinania się powierzchni algebraicznych. Koncepcyjnie metoda ta jest najbliższa metodzie opisanej przez Danielssona /1970 r./. Nie zawiera jednak ograniczenia na kierunek elementarnych ruchów wzdłuż krzywej, które powodowało dużą "chropowatość" krzywej dyskretnej. Inne jest również kryterium wyboru punktów aproksymujących. Poza tym proponowany algorytm

dyskretyzacji dotyczy dyskretyzacji dowolnych algebraicznych krzywych przestrzennych, a nie tylko krzywych wyznaczonych powierzchniami kwadratowymi.

SFORMUŁOWANIE I ANALIZA ZAGADNIENIA DYSKRETYZACJI PRZESTRZENNYCH KRZYWYCH ALGEBRAICZNYCH

Sformułowanie zagadnienia

Niech w prostokątnym układzie współrzędnych OXYZ zadana będzie krzywa K będąca krawędzią przecinania się dwóch powierzchni algebraicznych zadanych równaniami w postaci uwikłanej:

$$P(x,y,z) = \sum_{i=0}^{n} \sum_{j=0}^{n-i} \sum_{k=0}^{n-i-j} a_{j,k,n-i-j-k}\, x^j y^k z^{n-i-j-k} = 0 \qquad /1/$$

$$Q(x,y,z) = \sum_{i=0}^{m} \sum_{j=0}^{m-i} \sum_{k=0}^{m-i-j} b_{j,k,m-i-j-k}\, x^j y^k z^{m-i-j-k} = 0$$

/$a_{i,j,k}$, $b_{i,j,k}$ - liczby rzeczywiste; m,n - liczby naturalne; x,y,z - zmienne rzeczywiste/

i niech powierzchnie te będą w rozważanym obszarze analityczne, tj. mają w każdym swoim punkcie ciągłe i jednocześnie nie równe zeru pochodne cząstkowe stopnia 1+n oraz 1+m odpowiednio.

Wprowadźmy uproszczone oznaczenia dla zapisu pochodnych cząstkowych:

$$\frac{\partial^r P(x,y,z)}{\partial x^r} \stackrel{df}{=} P_{rx}(x,y,z)$$

$$\frac{\partial^{r+s} P(x,y,z)}{\partial x^r \partial y^s} \stackrel{df}{=} P_{rxsy}(x,y,z)$$

$$\frac{\partial^{r+s+t} P(x,y,z)}{\partial x^r \partial y^s \partial z^t} \stackrel{df}{=} P_{rxsytz}(x,y,z)$$

Nadajmy w układzie OXYZ siatkę przestrzenną z modułem h $G(h^3)$. Przedtem jednak określmy kilka pomocniczych definicji.

Definicja 1

Rzędem spójności siatki BSG nazwiemy liczbę półprostych siatkowych wychodzących z jej węzła.

Definicja 2

Bezpośrednio spójnymi węzła siatki BSWG nazwiemy takie dwa węzły, które linia siatkowa łączy bezpośrednio /bez pośrednictwa innych węzłów/.

Definicja 3

Głównymi liniami siatki GLG nazwiemy te linie, wzdłuż których najkrótsza odległość między BSWG rów-

na jest modułowi siatki h, dwuprzekątnymi liniami siatki 2PLG - gdy ta odległość równa jest $h\sqrt{2}$, trójprzekątnymi 3PLG - gdy $h\sqrt{3}$.

Z definicji 3 wynika, że w trakcie ruchu wzdłuż głównej linii siatki zmienia się tylko jedna ze współrzędnych x,y lub z punktu bieżącego, gdy ruch ten odbywa się wzdłuż 2PLG, wtedy zmieniają się dwie współrzędne, natomiast gdy jest to ruch wzdłuż 3PLG - ulegają zmianie wszystkie trzy współrzędne x,y i z punktu przemieszczanego.

Wyznaczmy jednorodną siatkę przestrzenną z modułem h $G(h^3)$ za pomocą zbioru płaszczyzn zdefiniowanych niezależnymi równaniami:

$$1/\ y_i = h\cdot i + \frac{p}{q}x\ , \quad -\infty < x,z < +\infty\ ,$$

$$2/\ z_i = h\cdot i + \frac{p}{q}y\ , \quad -\infty < x,y < +\infty\ , \qquad /2/$$

$$3/\ x_i = h\cdot i + \frac{p}{q}z\ , \quad -\infty < y,z < +\infty\ ,$$

h - liczba naturalna, $i = -\left\lfloor\frac{m-1}{2}\right\rfloor(1)\left\lfloor\frac{m}{2}\right\rfloor$, p, q - liczby całkowite, $q\neq 0$, x,y,z - zmienne rzeczywiste, m - rozmiar liniowy obszaru dyskretnego /liczba naturalna/.

Linie siatki są krawędziami przecinania się płaszczyzn /2/

Przyjmując dla p i q różne wartości, otrzymać można różne rodzaje siatek. Zdefiniujmy dwie poniższe /najprostsze/, mające największe zastosowanie w praktyce:

a/ p=0, siatka przestrzenna 6-spójna $G_{6s}(h^3)$, której linie wyznaczone są za pomocą krawędzi przecinania się płaszczyzn określonych niezależnymi równaniami:

$$\begin{aligned} &1/\ x_i = h\cdot i,\ -\infty < y,z < +\infty\ ,\\ &2/\ y_i = h\cdot i,\ -\infty < x,z < +\infty\ ,\\ &3/\ z_i = h\cdot i,\ -\infty < x,y < +\infty\ ; \end{aligned} \qquad /3/$$

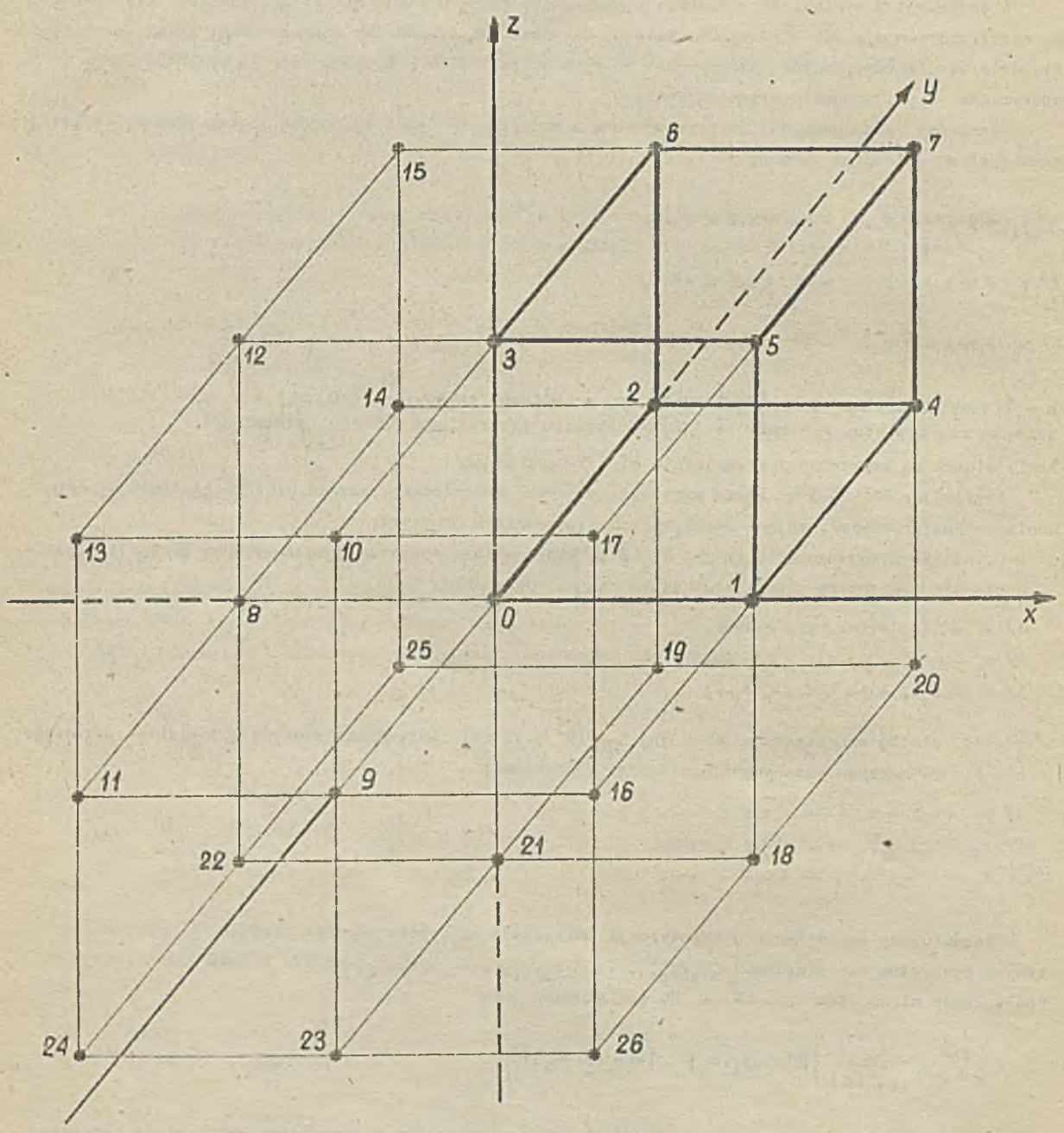
b/ $\frac{p}{q} = \pm 1$, siatka przestrzenna 26-spójna $G_{26s}(h^3)$, której linie wyznaczane są krawędziami przecinania się płaszczyzn zadanych niezależnymi równaniami:

$$\begin{aligned} &1/\ y_i = h\cdot i \pm x,\ -\infty < x,z < +\infty\ ,\\ &2/\ z_i = h\cdot i \pm y,\ -\infty < x,y < +\infty\ ,\\ &3/\ x_i = h\cdot i \pm z,\ -\infty < y,z < +\infty\ . \end{aligned} \qquad /3a/$$

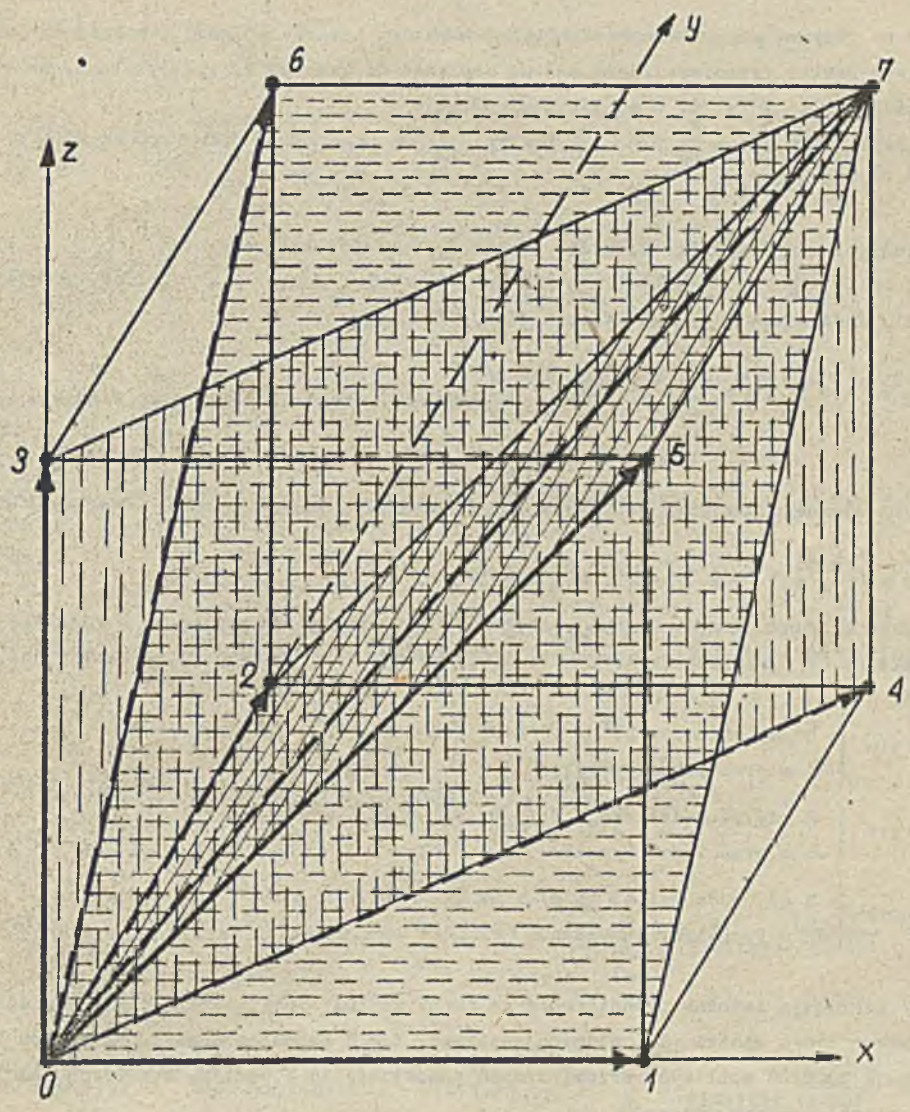
Formułujemy zagadnienie dyskretyzacji polegające na aproksymowaniu krzywej K postaci /1/ krzywą dyskretną H - ciągiem: $\{\langle x_p,y_p\rangle,\ldots,\langle x_i,y_i\rangle,\ldots,\langle x_k,y_k\rangle\}$ BSWP, w taki sposób, aby "odległość" między krzywymi K i H, definiowana jako

$$\rho^H_{P,Q} = \max_{p\leq i\leq k}\left\{\left|P(x_i,y_i,z_i)\right| + \left|Q(x_i,y_i,z_i)\right|\right\}_{||} \qquad /4/$$

była minimalna. Algorytm powinien być prosty i szybki w realizacji komputerowej, metoda natomiast nie powinna powodować błędów systematycznych.



Rys. 1. Ilustracja do określenia siatki przestrzennej



Rys. 2. Linie przestrzennej siatki $G_{26s}(h^3)$ wyznaczone przez krawędzie przecinających się płaszczyzn

Analiza zagadnienia

Zdefiniujmy na krzywej /1/ kierunek określając w każdym punkcie krzywej i w dostatecznie bliskim jej sąsiedztwie wektor przemieszczenia wzdłuż stycznej do krzywej $\vec{V}$, będący iloczynem wektorowym normalnych do powierzchni P i Q, w sposób następujący:

$$\vec{V} = \vec{V}(V^x, V^y, V^z) = \vec{V}(P_y Q_z - P_z Q_y,\ P_z Q_x - P_x Q_z,\ P_x Q_y - P_y Q_x) \qquad /5/$$

Wprowadźmy również zmienną logiczną B, która przyjmuje wartości logiczne:

B = 1 dla dodatniego kierunku wektora $\vec{V}(+\vec{V})$, tzn. dla

$$\vec{V} = \vec{V}(P_y Q_z - P_z Q_y,\ P_z Q_x - P_x Q_z,\ P_x Q_y - P_y Q_x)$$

oraz wartość

B = 0 dla ujemnego kierunku wektora $\vec{V}(-\vec{V})$, tzn. dla

$$\vec{V} = \vec{V}(P_z Q_y - P_y Q_z,\ P_x Q_z - P_z Q_x,\ P_y Q_x - P_x Q_y) \qquad /5'/$$

Określmy także kierunki przemieszczeń wzdłuż GLG, odpowiadające przyjętemu w /5/ kierunkowi ruchów wzdłuż wektora $\vec{V}$ /lokalizacja LV1, LV1: $\{x^x, y^y, v^z\}$, $V^x = V^y(\Delta x,0,0)$, $V^y = V^y(0,\Delta y,0)$, $V^z = V^z(0,0,\Delta z)$/:

$$\Delta x := \begin{cases} h \text{ gdy zachodzi } (V^x \geqslant \emptyset) \wedge \text{Bv } (V^x < \emptyset) \wedge \bar{B} = 1 \\ -h \text{ w przeciwnym wypadku} \end{cases}$$

$$\Delta y := \begin{cases} h \text{ gdy zachodzi } (V^y \geqslant \emptyset) \wedge \text{Bv } (V^y < \emptyset) \wedge \bar{B} = 1 \\ -h \text{ w przeciwnym wypadku} \end{cases} \qquad /6/$$

$$\Delta z := \begin{cases} h \text{ gdy zachodzi } (V^z \geqslant \emptyset) \wedge \text{Bv } (V^z < \emptyset) \wedge \bar{B} = 1 \\ -h \text{ w przeciwnym wypadku.} \end{cases}$$

Relacje /6/ nakładają istotne ograniczenia na ruchy wzdłuż linii siatki. Z sześciu możliwych przemieszczeń wzdłuż linii siatki G_{6s} eliminują połowę, tj. 3 przemieszczenia. Na siatce G_{26s} wzory /6/ eliminują 19 z 26 możliwych przemieszczeń pozostawiając 7 węzłów, z których jeden należy wybrać na węzeł ciągu aproksymującego.

Przyjmijmy miarę odległości punktu od krzywej przestrzennej. Niech to będzie suma modułów wartości funkcji

$$\varphi_{P,Q} \overset{\text{def}}{=} |P(x,y,z)| + |Q(x,y,z)| \qquad /7/$$

Kroczenie wzdłuż krzywej /1/ wymaga więc, aby na kolejny punkt ciągu aproksymującego wybrany został ten z BSWG, który można osiągnąć wykonując ruch określony przez relacje /6/ oraz, aby w tym węźle zachodziło

$$\varphi_{P,Q} \longrightarrow \min_{(x_i,y_j,z_k) \in \{\text{ciąg BSWG}\}} \qquad /8/$$

tj. spośród wszystkich rozważanych węzłów.

Należy więc wyznaczać moduły wartości funkcji P i Q w wyselekcjonowanych przez relację /6/ węzłach siatki.

Rozważmy rozwinięcia funkcji P i Q w szereg Taylora mające ogólną postać:

$$P(x+\Delta x,y+\Delta y,z+\Delta z) = P(x,y,z) + \sum_{i=1}^{n} \frac{1}{i!}\left(\Delta x \frac{\partial}{\partial x} + \Delta y \frac{\partial}{\partial y} + \Delta z \frac{\partial}{\partial z}\right)^{i} P(x,y,z)$$

/9/

$$Q(x+\Delta x,y+\Delta y,z+\Delta z) = Q(x,y,z) + \sum_{i=1}^{m} \frac{1}{i!}\left(\Delta x \frac{\partial}{\partial x} + \Delta y \frac{\partial}{\partial y} + \Delta z \frac{\partial}{\partial z}\right)^{i} Q(x,y,z)$$

i wprowadźmy poniższe operatory kierunkowe funkcji P i Q:

$$L^{x}P(x,y,z) \stackrel{def}{=} P(x+\Delta x,y,z) - P(x,y,z) = \sum_{i=1}^{n} \frac{(\Delta x)^{i}}{i!} P_{ix}(x,y,z)$$

$$L^{y}P(x,y,z) \stackrel{def}{=} P(x,y+\Delta y,z) - P(x,y,z) = \sum_{j=1}^{n} \frac{(\Delta y)^{j}}{j!} P_{jy}(x,y,z)$$

$$L^{z}P(x,y,z) \stackrel{def}{=} P(x,y,z+\Delta z) - P(x,y,z) = \sum_{k=1}^{n} \frac{(\Delta z)^{k}}{k!} P_{kz}(x,y,z)$$

$$L^{xy}P(x,y,z) \stackrel{def}{=} P(x+\Delta x,y+\Delta y,z) - P(x,y,z) = \sum_{i=1}^{n}\sum_{j=0}^{i} P_{(i-j)xjy} \frac{(\Delta x)^{i-j}}{(i-j)!} \frac{(\Delta y)^{j}}{j!}$$

$$L^{xz}P(x,y,z) \stackrel{def}{=} P(x+\Delta x,y,z+\Delta z) - P(x,y,z) = \sum_{i=1}^{n}\sum_{k=0}^{i} P_{(i-k)xkz} \frac{(\Delta x)^{i-k}}{(i-k)!} \frac{(\Delta z)^{k}}{k!}$$

/10/

$$L^{yz}P(x,y,z) \stackrel{def}{=} P(x,y+\Delta y,z+\Delta z) - P(x,y,z) = \sum_{j=1}^{n}\sum_{k=0}^{j} P_{(j-k)ykz} \frac{(\Delta y)^{j-k}}{(j-k)!} \frac{(\Delta z)^{k}}{k!}$$

$$L^{xyz}P(x,y,z) \stackrel{def}{=} P(x+\Delta x,y+\Delta y,z+\Delta z) - P(x,y,z) =$$

$$= \sum_{i=1}^{n}\sum_{j=0}^{i}\sum_{k=0}^{j} P_{(i-j)x(j-k)ykz} \frac{(\Delta x)^{i-j}(\Delta y)^{j-k}(\Delta z)^{k}}{(i-j)!(j-k)!\,k!}$$

Dla funkcji Q podobnie:

$$L^{x}Q\,(x,y,z) \stackrel{def}{=} Q(x+\Delta x,y,z) - Q(x,y,z) = \sum_{i=1}^{m} \frac{(\Delta x)^{i}}{i!}\, Q_{ix}(x,y,z)$$

$$L^{y}Q\,(x,y,z) \stackrel{def}{=} Q(x,y+\Delta y,z) - Q(x,y,z) = \sum_{j=1}^{m} \frac{(\Delta y)^{j}}{j!}\, Q_{jy}x,y,z)$$

$$L^{z}Q\,(x,y,z) \stackrel{def}{=} Q(x,y,z+\Delta z) - Q(x,y,z) = \sum_{k=1}^{m} \frac{(\Delta z)^{k}}{k!}\, Q_{kz}(x,y,z)$$

$$L^{xy}Q\,(x,y,z) \stackrel{def}{=} Q(x+\Delta x,y+\Delta y,z) - Q(x,y,z) = \sum_{i=1}^{m}\sum_{j=0}^{i} Q_{(i-j)xjy}\, \frac{(\Delta x)^{i-j}(\Delta y)^{j}}{(i-j)!\;j!}$$

$$L^{xz}Q\,(x,y,z) \stackrel{def}{=} Q(x+\Delta x,y,z+\Delta z) - Q(x,y,z) = \sum_{i=1}^{m}\sum_{k=0}^{i} Q_{(i-k)xkz}\, \frac{(\Delta x)^{i-k}(\Delta z)^{k}}{(i-k)!\;k!} \qquad /10'/$$

$$L^{yz}Q\,(x,y,z) \stackrel{def}{=} Q(x,y+\Delta y,z+\Delta z) - Q(x,y,z) = \sum_{j=1}^{m}\sum_{k=0}^{j} Q_{(j-k)ykz}\, \frac{(\Delta y)^{j-k}(\Delta z)^{k}}{(j-k)!\;k!}$$

$$L^{xyz}Q\,(x,y,z) \stackrel{dcf}{=} Q(x+\Delta x,y+\Delta y,z+\Delta z) - Q(x,y,z) =$$

$$= \sum_{i=1}^{m}\sum_{j=0}^{i}\sum_{k=0}^{j} Q(i-j)\;x\;(j-k)ykz\; \frac{(\Delta x)^{i-j}(\Delta y)^{j-k}(\Delta z)^{k}}{(i-j)!\;(j-k)!\;k!}\,.$$

Wzory z prawej strony równości /10/ i /10'/ w komputerowej realizacji są wygodniejszymi odpowiednikami wzorów wynikających z rozwinięcia /9/ funkcji P i Q w szereg Taylora.

Zakładając, że znane są wartości pochodnych cząstkowych dla funkcji P i Q rzędu 1+n i 1+m odpowiednio, wyznaczenie funkcji P i Q w sąsiednich węzłach sprowadza się więc do obliczenia operatorów /10/ i /10'/ dla tych funkcji, przy określonych przez relacje /6/ wartościach Δx, Δy i Δz oraz do dodania ich do wartości $P(x,y,z)$ i $Q(x,y,z)$.

Dalsze czynności prowadzące do wyboru kolejnego punktu ciągu wymagają przeprowadzenia wzajemnych porównań wartości $\wp_{P,Q}$ w analizowanych węzłach /wyznaczonych przez relacje /6/ /. Węzeł, dla którego zachodzi $\wp_{P,Q}=\min$ będzie węzłem krzywej dyskretnej H.

Wartości pochodnych cząstkowych rzędu odpowiednio 1+n i 1+m dla funkcji P i Q wyznaczyć można wprost z równań /1/, bądź też uaktualnić je, gdy znane są one w poprzednim punkcie,

wykorzystując ich rozwinięcie w szereg Taylora. Można w tym celu wykorzystać podane niżej wzory.

Wprowadźmy operatory kierunkowe pochodnych cząstkowych funkcji P i Q, podobnie jak dla funkcji P i Q:

$$
\begin{aligned}
L^{x} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x+\Delta x,y,z) - P_{rxsytz}(x,y,z) \\
L^{y} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x,y+\Delta y,z) - P_{rxsytz}(x,y,z) \\
L^{z} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x,y,z+\Delta z) - P_{rxsytz}(x,y,z) \\
L^{xy} P_{rxsytz}(x,y,z &\stackrel{def}{=} P_{rxsytz}(x+\Delta x,y+\Delta y,z) - P_{rxsytz}(x,y,z) \\
L^{xz} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x+\Delta x,y,z+\Delta z) - P_{rxsytz}(x,y,z) \\
L^{yz} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x,y+\Delta y,z+\Delta z) - P_{rxsytz}(x,y,z) \\
L^{xyz} P_{rxsytz}(x,y,z) &\stackrel{def}{=} P_{rxsytz}(x+\Delta x,y+\Delta y,z+\Delta z) - P_{rxsytz}(x,y,z) .
\end{aligned}
\qquad /11/
$$

Operatory te można wyznaczyć z następujących wzorów:

$$
\begin{aligned}
L^{x} P_{rxsytz}(x,y,z) &= \sum_{i=1}^{n-r-s-t} P_{(r+i)xsytz} \frac{(\Delta x)^{i}}{i!} \\
L^{y} P_{rxsytz}(x,y,z) &= \sum_{j=1}^{n-r-s-t} P_{rx(s+j)ytz} \frac{(\Delta y)^{j}}{j!} \\
L^{z} P_{rxsytz}(x,y,z) &= \sum_{k=1}^{n-r-s-t} P_{rxsy(t+k)z} \frac{(\Delta z)^{k}}{k!} \\
L^{xy} P_{rxsytz}(x,y,z) &= \sum_{i=1}^{n-r-s-t} \sum_{j=0}^{i} P_{(r+i-j)x(s+j)ytz} \frac{(\Delta x)^{i-j}(\Delta y)^{j}}{(i-j)!\ j!} \\
L^{xz} P_{rxsytz}(x,y,z) &= \sum_{i=1}^{n-r-s-t} \sum_{k=0}^{i} P_{(r+i-k)xsy(t+k)z} \frac{(\Delta x)^{j-k}(\Delta z)^{k}}{(i-k)!\ k!} \\
L^{yz} P_{rxsytz}(x,y,z) &= \sum_{j=1}^{n-r-s-t} \sum_{k=0}^{j} P_{rx(s+j-k)y(t+k)z} \frac{(\Delta y)^{j-k}(\Delta z)^{k}}{(j-k)!\ k!}
\end{aligned}
\qquad /11'/
$$

$$I^{xyz}P_{rxsytz}(x,y,z) = \sum_{i=1}^{n-r-s-t} \sum_{j=0}^{i} \sum_{k=0}^{j} P_{(r+i-j)x(s+j-k)y(t+k)z} \frac{(\Delta x)^{i-j}(\Delta y)^{j-k}(\Delta z)^{k}}{(i-j)!\,(j-k)!\,k!}$$

dla r,s,t = 1(1)n-1, r+s+t = 1(1)n-1.

Dla pochodnych funkcji Q operatory powyższe definiuje się podobnie. Uaktualnienie pochodnych wymaga dodania do pochodnej właściwego przemieszczenia i operatora tejże pochodnej.

Optymalizacja wyznaczania węzła na siatce $G_{26s}(h^3)$ metodą uściślenia położenia wektora stycznej

Określone relacjami /6/ kierunki elementarnych ruchów LV1 eliminują z rozważań na siatce $G_{26s}(h^3)$ 19 z 26 możliwych przemieszczeń do węzłów bezpośrednio spójnych.

W celu wyboru jednego z pozostałych 7 węzłów na kolejny punkt ciągu należy obliczyć moduły wartości funkcji w tych węzłach dla obydwu funkcji P i Q i porównując sumy modułów tych funkcji w tych węzłach wybrać węzeł, w którym suma ta jest minimalna. Możliwe jest jednak uproszczenie tych czynności przez uściślenie LV1 dotychczasowego położenia wektora stycznej $\vec{V}$ /z dokładnością do $\frac{\pi}{2}$/, które można osiągnąć przeprowadzając wzajemne porównanie modułów składowych V^x, V^y i V^z tego wektora.

Rozważmy kostkę sześcienną o boku h z węzłami ponumerowanymi od 1 do 7 jak na rys.2, w której zlokalizowane zostało /relacjami /6/ położenie wektora $\vec{y}$. Rozważmy również obszary, na które kostka ta została podzielona płaszczyznami /4/, których krawędzie przecinania się tworzą linie siatki. Płaszczyzny te dzielą kostkę na 6 obszarów - ostrosłupów o podstawie trójkątnej, z wierzchołkami w węzłach siatki /wierzchołkach kostki/. Każdy z tych ostrosłupów ma jeden wierzchołek w punkcie o numerze 0, pozostałe 3 w węzłach sąsiadujących ze sobą. Wzajemne relacje nierówności między składowymi V^x, V^y i V^z wektora $\vec{V}$ pozwalają na jego lokalizację w jednym z tych sześciu obszarów /lokalizacja LV2/, a więc uściślają położenie wektora $\vec{V}$. Eliminują w ten sposób 4 z 7 niewyeliminowanych dotąd węzłów.

Zachodzić mogą następujące, niesprzeczne przypadki porównań składowych wektora $\vec{V}$, tj. V^x, V^y i V^z. Odpowiedni obszar /ostrosłup przestrzenny/ wyznaczony przez płaszczyzny /opisane równaniami /4/ / ograniczony jest wtedy przez te płaszczyzny, które przechodzą przez węzeł aktualny /o numerze 0/ i węzły wymienione w prawych częściach relacji występujących w lokalizacji LV2:

gdy zachodzi $(|V^x|>|V^y|) \wedge (|V^x|<|V^z|)$ = 1, to węzeł 0 i węzły 3,5 i 7
gdy zachodzi $(|V^x|>|V^y|) \wedge (|V^x|\geq|V^z|) \wedge |V^y|>|V^z|$ = 1, to węzeł 0 i węzły 1,4 i 7
gdy zachodzi $(|V^x|>|V^y|) \wedge (|V^x|\geq|V^z|) \wedge |V^y|\leq|V^z|$ = 1, to węzeł 0 i węzły 1,5 i 7
gdy zachodzi $(|V^x|\leq|V^y|) \wedge (|V^x|\geq|V^z|)$ = 1, to węzeł 0 i węzły 2,4 i 7 /12/
gdy zachodzi $(|V^x|\leq|V^y|) \wedge (|V^x|<|V^z|) \wedge |V^y|>|V^z|$ = 1, to węzeł 0 i węzły 2,6 i 7
gdy zachodzi $(|V^x|\leq|V^y|) \wedge (|V^x|<|V^z|) \wedge |V^y|\leq|V^z|$ = 1, to węzeł 0 i węzły 3,6 i 7 .

W ostatnich relacjach znaki słabej nierówności - lokalizujące położenie wektora $\vec{V}$ wewnątrz określonego ostrosłupa w przypadku granicznym i oznaczające równoległość wektora stycznej do jednej z płaszczyzn tnących - ustawione zostały według układu priorytetów współrzędnych UP7:

- y ma priorytet nad x
- z ma priorytet nad y /13/
- x ma priorytet nad z.

Relacje /12/ i /13/ redukują więc z 7 do 3 liczbę węzłów, w których należy wyznaczać moduły wartości funkcji. Do dalszego rozstrzygnięcia pozostał jeszcze przypadek, dla którego wartości sumy modułów w dwóch z trzech węzłów są minimalne.

Wprowadźmy następujące kryterium rozstrzygające opisany przypadek: wybieramy węzeł bardziej odległy od węzła aktualnego. Dochodzimy w ten sposób do określenia priorytetu wyboru punktów-węzłów PWW:

- priorytet najwyższy ma węzeł o numerze 7
- priorytet niższy mają węzły o numerach 4,5 i 6 /14/
- priorytet najniższy mają węzły o numerach 1, 2 i 3 .

Ostatnie kryterium całkowicie rozstrzyga o wyborze węzła siatki na kolejny punkt aproksymujący, bowiem żaden z dwóch punktów w ramach wymienionych w relacjach trójek /12/ nie ma tego samego priorytetu.

Ustala się więc następujące zasady wyboru węzła ZWW na kolejny punkt ciągu:

- wybierany jest węzeł, dla którego zachodzi $|P| + |Q| = \min$ /14'/
- wybierany jest węzeł z wyższym priorytetem.

Optymalizacja wyznaczenia węzła na siatce $G_{26s}(h^3)$ metodą uproszczenia obliczeń wartości funkcji

Rozważmy operatory kierunkowe funkcji P i Q wprowadzone wzorami /10/. Dla każdej trójki punktów wyznaczonych nierównościami /12/ należy obliczyć odpowiednie operatory kierunkowe funkcji P i Q; w sumie 6 operatorów.

Złożoność wyznaczania tych operatorów jest różna. Najbardziej złożonych obliczeń wymaga wyznaczenie operatora L^{xyz}, który /dla siatki $G_{26s}(h^3)$/ należy wyznaczać dla każdej trójki punktów i obydwu funkcji P i Q. Najprostsze do wyznaczenia są operatory L^x, L^y i L^z.

Zauważmy jednak, że bardziej złożone operatory można rozłożyć na operatory proste i pewne operatory resztowe, prostsze do obliczeń niż operatory wyjściowe. Istnieją bowiem poniższe zależności:

$$L^{xy}P(x,y,z) = L^xP + L^yP + L^{Rxy}P$$

gdzie $$L^{Rxy}P(x,y,z) = \sum_{j=1}^{n-1}\sum_{k=1}^{j} P_{(j-k+1)xky}\frac{(\Delta x)^{j-k+1}(\Delta y)^k}{(j-k+1)!\ k!},$$

$$L^{xz}P(x,y,z) = L^xP + L^zP + L^{Rxz}P$$

gdzie $$L^{Rxz}P(x,y,z) = \sum_{j=1}^{n-1}\sum_{k=1}^{j} P_{(j-k+1)xkz}\frac{(\Delta x)^{j-k+1}(\Delta z)^k}{(j-k+1)!\ k!},$$

$$L^{yz}P(x,y,z) = L^{y}P + L^{z}P + L^{Ryz}P \qquad /15/$$

gdzie $$L^{Ryz}P(x,y,z) = \sum_{j=1}^{n-1}\sum_{k=1}^{j} P_{(j-k+1)ykz}\,\frac{(\Delta y)^{j-k+1}(\Delta z)^{k}}{(j-k+1)!\;k!}\,,$$

oraz zależność

$$L^{xyz}P(x,y,z) = L^{x}P + L^{y}P + L^{z}P + L^{Rxy}P + L^{Rxz}P + L^{Ryz}P + L^{Rxyz}P$$

gdzie

$$L^{Rxyz}P(x,y,z) = \sum_{i=1}^{n-1}\sum_{j=1}^{i}\sum_{k=1}^{j} P_{(i-j+1)x(j-k+1)ykz}\,\frac{(\Delta x)^{i-j+1}(\Delta y)^{j-k+1}(\Delta z)^{k}}{(i-j+1)!\;(j-k+1)!\;k!}\,.$$

Podobnie dla funkcji Q.

Operatory $L^{Rxy}(...)$, $L^{Rxz}(...)$, $L^{Ryz}(...)$ nazywać będziemy operatorami różnicowymi dwuparametrowymi, operator $L^{Rxyz}(...)$ natomiast operatorem różnicowym trójparametrowym.

Korzystając ze wzorów /15/ możemy więc wyznaczyć moduły wartości funkcji w każdym punkcie każdej trójki węzłów, wykonując obliczenia modułu wartości funkcji P i Q jedynie dla punktu o numerze 7, dla którego wyznaczenie operatora jest najbardziej złożone. Wartości funkcji w pozostałych węzłach nie wymagają wtedy wykonania praktycznie żadnych obliczeń, z wyjątkiem wykonania kilku operacji dodawania, tj. operacji najprostszych.

Przeprowadzone wyżej rozważania pozwalają skonstruować przedstawione w następnym rozdziale komputerowo realizowalne algorytmy dyskretyzacji /aproksymacji/ krzywych algebraicznych przestrzennych zadanych za pomocą uwikłanych równań przecinających się powierzchni.

ALGORYTMY DYSKRETYZACJI ALGEBRAICZNYCH KRZYWYCH PRZESTRZENNYCH

Algorytm dyskretyzacji na siatce $G_{6s}(h^3)$

Załóżmy, że znane są współczynniki $a_{j,k,n-i-j-k}$ oraz $b_{j,k,m-i-j-k}$ równań uwikłanych /1/ przecinających się powierzchni wyznaczających krzywą, współrzędne $<x_p,y_p,z_p>$ punktu początkowego na krzywej /lub w najbliższym jej sąsiedztwie/ pokrywającego się z jednym z węzłów siatki przestrzennej, a także wartości funkcji P i Q w tym punkcie. Niech zadany będzie również kierunek ruchu wzdłuż krzywej B oraz punkt końcowy $<x_k,y_k,z_k>$ leżący na tej krzywej /lub w najbliższym sąsiedztwie/. W celu wyznaczenia drugiego węzła ciągu aproksymującego wymagana jest znajomość wartości pochodnych cząstkowych wszystkich stopni dla obydwu funkcji P i Q w punkcie $<x_p,y_p,z_p>$. Pochodne te można wyznaczyć z następującego wzoru:

$$P_{rxsytz}(x,y,z) = \sum_{t=0}^{n-r-s-t} \sum_{j=r}^{n-i-s-t} \sum_{n=s}^{n-i-j-t} \prod_{p=n-i-j-k-t+1}^{n-i-k-j}(p) \cdot \prod_{q=j-r+1}^{j}(q) \cdot \prod_{u=k-s+1}^{k}(u) \cdot a_{j,k,n-i-j-k} \cdot x^{j-r} \cdot y^{k-s} \cdot z^{n-i-j-k-t} \quad /16/$$

dla $r+s+t = 1(1)n$, $r,s,t = 1(1)n$. Dla funkcji Q podobnie.

Następnie wyznaczyć należy składowe V^x, V^y, V^z wektora kierunkowego stycznej /relacje /5/ / oraz kierunki elementarnych ruchów wzdłuż głównych linii siatki, zdefiniowane relacjami /6/.

Następnie należy obliczyć moduły wartości funkcji P i Q w węzłach 1, 2 i 3. Posługując się pierwszymi trzema wzorami /10/ wyznaczyć można operatory $L^x(...)$, $L^y(...)$, $L^z(...)$ dla obydwu funkcji P i Q. Kolejnym punktem ciągu aproksymującego będzie węzeł siatki "/o numerze 1, 2 lub 3/, w którym suma modułów funkcji P i Q przyjmuje wartość minimalną /relacje /7/ /. Gdy kryterium to nie rozstrzyga wyboru, należy dodatkowo zastosować kryterium z układem priorytetu /14/ i /14'/.

Mając wyznaczony kolejny węzeł aproksymujący należy sprawdzić, czy punkt ten nie jest punktem końcowym $<x_k,y_k,z_k>$ aproksymowanego segmentu krzywej. Proces dyskretyzacji należy zakończyć, gdy osiągnięty punkt jest punktem $<x_k,y_k,z_k>$, czyli, gdy spełniony jest warunek końca dyskretyzacji WKD:

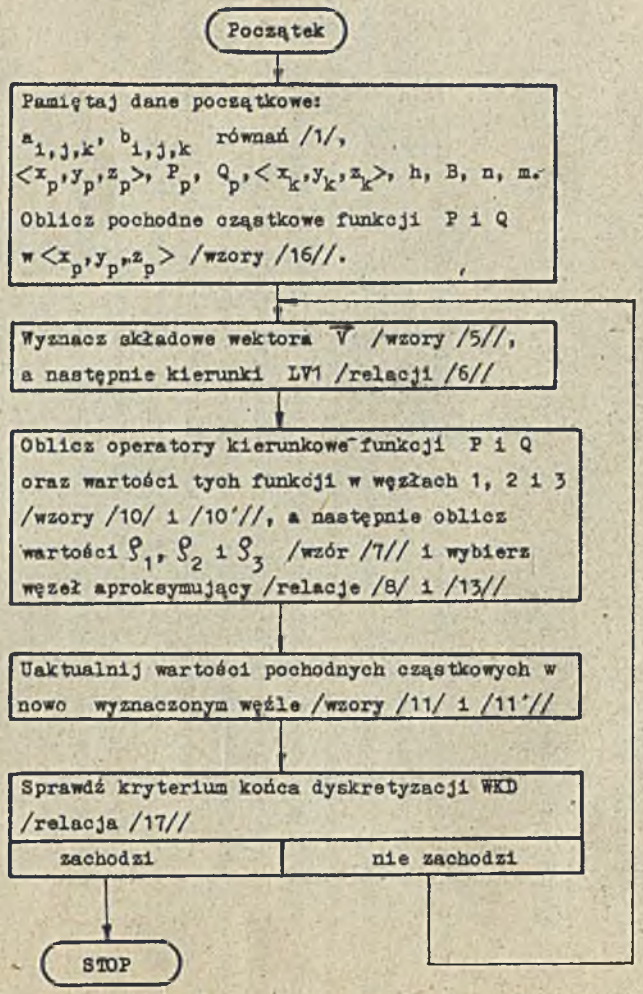
$$(|x-x_k|<h) \wedge (|y-y_k|<h) \wedge (|z-z_k|<h) = 1. \quad /17/$$

W przeciwnym razie należy uaktualnić wartości pochodnych cząstkowych funkcji P i Q w osiągniętym węźle /jeden z trzech pierwszych wzorów /11/ i /11'/ - odpowiedni do przemieszczenia/ i przystąpić do określania kolejnego punktu aproksymującego, wyznaczając składowe wektora $\vec{V}$, tj. V^x, V^y i V^z oraz kierunku elementarnych ruchów wzdłuż głównych linii siatki.

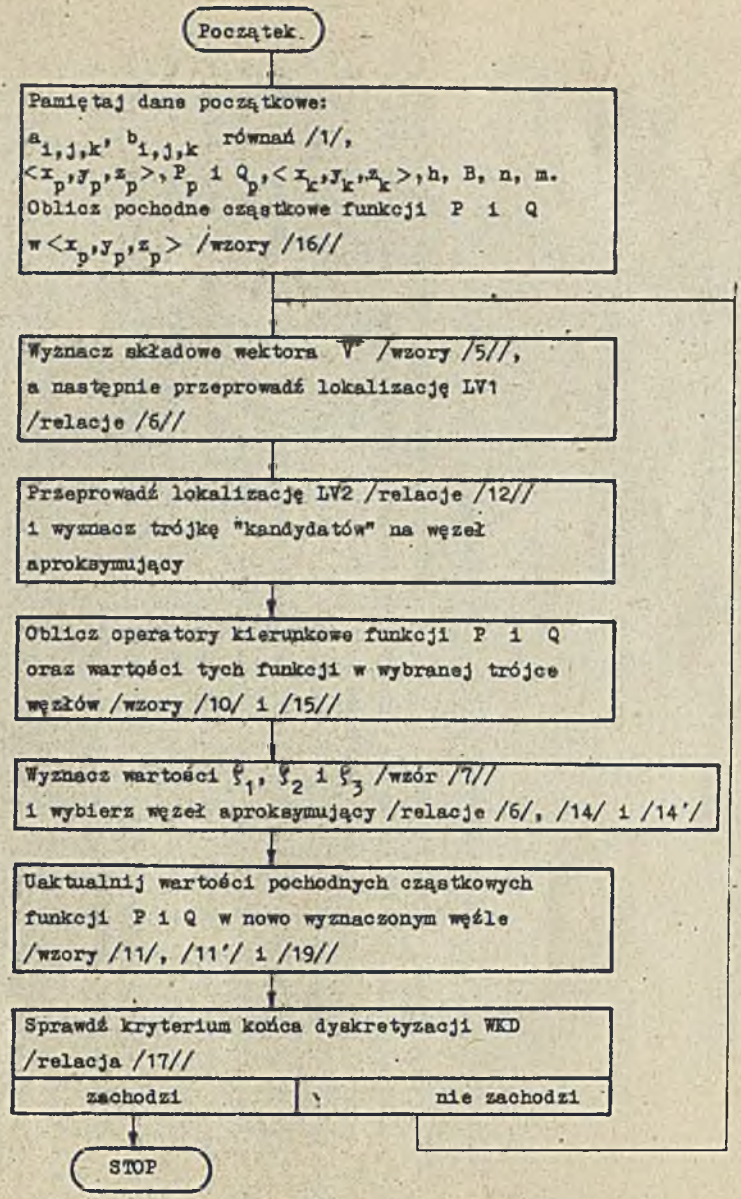
Schemat blokowy tego algorytmu przedstawiony został na rys.3.

Algorytm dyskretyzacji na siatce $G_{26s}(h^3)$

Załóżmy, że znane są, jak w wypadku siatki $G_{6s}(h^3)$ współczynniki $a_{j,k,n-i-j-k}$ oraz $b_{j,k,m-i-j-k}$ równań uwikłanych, przecinających się powierzchni wyznaczających krzywą /1/, współrzędne i wartości funkcji P i Q w punkcie początkowym, tzn. $<x_p,y_p,z_p>$ i P_p,Q_p oraz współrzędne punktu końcowego $<x_k,y_k,z_k>$, a także kierunek obiegu krzywej B. Wartości pochodnych cząstkowych funkcji P i Q w punkcie początkowym, które są niezbędne do wyznaczenia drugiego punktu ciągu aproksymującego krzywą, wyznaczymy ze wzorów /16/ podanych wyżej, w opisie algorytmu dyskretyzacji na siatce $G_{6s}(h^3)$. Kolejne dwie czynności algorytmu, tj. wyznaczenie składowych wektora stycznej /relacje /5/ / oraz elementarnych kierunków ruchu wzdłuż głównych linii siatki /relacji /6/ / są /podobnie jak czynności dotychczasowe algorytmu/ identyczne, jak dla algorytmu dyskretyzacji na siatce $G_{6s}(h^3)$. Operacją, która nie występowała w poprzednim algorytmie, jest lokalizacja wektora $\vec{V}$ /relacje /12/ /, w wyniku której wyselekcjonowane zostaną do dalszej analizy 3 z siedmiu RSWG.



Rys. 3. Algorytm dyskretyzacji krzywych przestrzennych na siatce $G_{26s}(h^2)$



Rys. 4. Algorytm dyskretyzacji krzywej przestrzennej na siatce G_{26s} h^3

Należy teraz obliczyć moduły wartości funkcji P i Q w tych węzłach. Trzeba posłużyć się tu pierwszymi trzema wzorami /10/ oraz dodatkowo wzorami /15/, wyznaczając operatory $L^x(...)$, $L^y(...)$, $L^z(...)$, $L^{Rxy}(...)$, $L^{Rxz}(...)$, $L^{Ryz}(...)$, a także $L^{Rxyz}(...)$. Kolejnym punktem ciągu aproksymującego będzie węzeł siatki, w którym funkcja φ_1 przyjmuje wartość minimalną /relacji /7/ /. Gdy kryterium to nie rozstrzyga, należy dodatkowo zastosować kryterium z układem priorytetu /14/. Mając wyznaczony kolejny punkt ciągu należy wyznaczyć wartości pochodnych cząstkowych funkcji P i Q w tym punkcie /odpowiedni do przemieszczenia jeden ze wzorów /11/ / i przystąpić do wyznaczenia kolejnego węzła aproksymującego, gdy kryterium końca generacji /17/ nie zachodzi /tzn. nie został osiągnięty końcowy punkt segmentu krzywej $<x_k, y_k, z_k>$/ lub zakończyć proces dyskretyzacji, gdy kryterium /17/ jest spełnione.

Określimy jeszcze, które operatory funkcji P i Q należy wyznaczać dla określonej trójki węzłów, wybranej za pomocą relacji /12/, oraz których operatorów należy używać w trakcie aktualizacji pochodnych cząstkowych w nowo wyznaczonym węźle aproksymującym.

Omówimy numerację węzłów kostki przestrzennej przedstawionej na rys.4. Ponieważ węzły oznaczone numerami 1, 2 i 3 położone są na GLG, osiągnięcie każdego z nich z aktualnego punktu /o numerze 0/ wymaga przemieszczenia jedynie wzdłuż jednej współrzędnej. Wymaga więc wyznaczenie jednego z operatorów $L^x(...)$, $L^y(...)$ lub $L^z(...)$ dla każdej z funkcji P i Q. Węzły o numerach 4, 5 i 6 leżą na płaszczyznach tworzących GLG /płaszczyznach równoległych do osi układu współrzędnych OXYZ/, a ich osiągnięcie z punktu O wymaga przemieszczenia wzdłuż dwóch /określonych/ współrzędnych. Wymaga więc wyznaczenia operatorów $L^{xy}(...)$, $L^{xz}(...)$ oraz $L^{yz}(...)$, jak również operatorów różnicowych $L^{Rxy}(...)$, $L^{Rxz}(...)$ oraz operatora $L^{Ryz}(...)$. Osiągnięcie natomiast punktu o numerze 7 wymaga zmiany wartości współrzędnych wzdłuż każdej z osi układu. Punkt ten leży bowiem na krawędzi przecinania się ukośnych płaszczyzn tworzących siatkę /na 2PLG/, dlatego też wymaga wyznaczenia operatora $L^{xyz}(...)$. Tym samym, jako jego składowych operatorów: jednoparametrowych $L^x(...)$, $L^y(...)$ i $L^z(...)$, dwuparametrowych różnicowych $L^{Rxy}(...)$, $L^{Rxz}(...)$ i $L^{Ryz}(...)$ oraz operatora różnicowego trójparametrowego $L^{Rxyz}(...)$.

W trakcie operacji lokalizacji LV2 wektora $\vec{V}$ /dokonywanej za pomocą relacji /12/ wyznaczyć należy następujące operatory /dla funkcji P i Q/:

$$
\begin{aligned}
&\text{- gdy zachodzi } (|v^x|>|v^y|)\wedge(|v^x|<|v^z|) = 1, && \text{to } L^z,\ L^{xz} \text{ i } L^{xyz}\\
&\text{- gdy zachodzi } (|v^x|>|v^y|)\wedge(|v^x|\geqslant|v^z|)\wedge(|v^y|\leqslant|v^z|) = 1, && \text{to } L^x,\ L^{xz} \text{ i } L^{xyz}\\
&\text{- gdy zachodzi } (|v^x|>|v^y|)\wedge(|v^x|>|v^z|)\wedge(|v^y|>|v^z|) = 1, && \text{to } L^x,\ L^{xy} \text{ i } L^{xyz}\\
&\text{- gdy zachodzi } (|v^x|\leqslant|v^y|)\wedge(|v^x|\geqslant|v^z|) = 1, && \text{to } L^y,\ L^{xy} \text{ i } L^{xyz}\\
&\text{- gdy zachodzi } (|v^x|\leqslant|v^y|)\wedge(|v^x|<|v^z|)\wedge(|v^y|>|v^z|) = 1, && \text{to } L^y,\ L^{yz} \text{ i } L^{xyz}\\
&\text{- gdy zachodzi } (|v^x|\leqslant|v^y|)\wedge(|v^x|<|v^z|)\wedge(|v^y|\leqslant|v^z|) = 1, && \text{to } L^z,\ L^{yz} \text{ i } L^{xyz}
\end{aligned}
\qquad /18/
$$

Wyznaczenie operatora $L^{xyz}(...)$ jest więc niezbędne w wypadku wyznaczenia dowolnej z trójki węzłów otrzymywanych w trakcie lokalizacji LV2 wektora $\vec{V}$.

Uaktualnienie pochodnych cząstkowych w nowowyznaczonym węźle jest operacją prostszą od poprzedniej. Wymaga bowiem wyznaczenia operatorów pochodnych cząstkowych funkcji P i Q tylko w jednym, wybranym na punkt aproksymujący węźle. Wzór, który należy w tym celu wykorzystać, zależy od wybranego węzła. I tak, gdy wybrany został punkt o numerze:

- 1, to należy korzystać z pierwszego ze wzorów /11'/
- 2, to należy korzystać z drugiego ze wzorów /11'/ /19/
- ⋮
- 7, to należy korzystać z siódmego ze wzorów /11'/

dla wartości r,s,t = 1(1)n-1/m-1/ r+s+t = 1(1)n-1/m-1/ funkcji P/Q/.

Algorytm dyskretyzacji krzywej przestrzennej na siatce $G_{26s}(h^3)$, w postaci blokowej przedstawiony został na rys.4.

Algorytm dla prostej przestrzennej

W przypadku linii prostej /przestrzennej/ algorytmy są niezwykle proste. W pętlach algorytmów wykonywanych jest bowiem niewiele prostych operacji komparacji modułów liczb i operacji na argumentach boolowskich. Algorytmy nadają się więc do realizacji układowej.

Niech przestrzenna prosta zadana będzie za pomocą dwóch przecinających się płaszczyzn:

$$P(x,y,z) = a_{100}x + a_{010}y + a_{001}z + a_{000} = 0$$
$$Q(x,y,z) = b_{100}x + b_{010}y + b_{001}z + b_{000} = 0 \quad . \qquad /20/$$

Wyznaczmy pochodne cząstkowe funkcji P i Q:

$$P_x = a_{100}, \quad P_y = a_{010}, \quad P_z = a_{001}$$
$$Q_x = b_{100}, \quad Q_y = b_{010}, \quad Q_z = b_{001} \qquad /21/$$

które, jak widać są stałe. Pochodne wyższych rzędów obydwu funkcji P i Q są tożsamościowo równe zero. Wektor $\vec{V}$ ma więc składowe stałe równe:

$$V^x = a_{010}b_{001} - a_{001}b_{010}$$
$$V^y = a_{001}b_{100} - a_{100}b_{001} \qquad /22/$$
$$V^z = a_{100}b_{010} - a_{010}b_{100} .$$

Kierunki elementarnych ruchów określone za pomocą operacji LV1 /wzór /6/ / wystarczy wyznaczyć raz na początku algorytmu. Są bowiem stałe. Również raz, na początku algorytmu uściślić należy /w przypadku siatki $G_{26s}(h^3)$/ położenie wektora $\vec{V}$ stosując operację lokalizacji LV2 /relacje /12 / oraz obliczyć właściwe operatory kierunkowe dla funkcji P i Q. Przyjmą one wtedy następujące wartości /wzory /10/ i /15/ /:

$$L^xP = \Delta x a_{100}, \quad L^yP = \Delta y a_{010}, \quad L^zP = \Delta z a_{001}$$
$$L^xQ = \Delta x b_{100}, \quad L^yQ = \Delta y b_{010}, \quad L^zQ = \Delta z b_{001} \qquad /23/$$

a operatory różnicowe funkcji P i Q są tożsamościowo równe zero. Stąd też

$$L^{xy}(\ldots) = L^{x}(\ldots) + L^{y}(\ldots)$$
$$L^{xz}(\ldots) = L^{x}(\ldots) + L^{z}(\ldots) \qquad /24/$$
$$L_{yz}(\ldots) = L_{y}(\ldots) + L^{z}(\ldots) .$$

Operatory te, tak dla funkcji P jak i Q, wyznacza się jedynie dla wyselekcjonowanej /w przypadku siatki $G_{26s}(h^3)$/ trójki węzłów. Kolejną operacją jest obliczenie modułów wartości funkcji P i Q we wspomnianej trójce węzłów:

$$P_1 = P_0 + L^{*}P_0, \quad P_2 = P_0 + L^{**}P_0, \quad P_3 = P_0 + L^{***}P_0$$
$$Q_1 = Q_0 + L^{*}Q_0, \quad Q_2 = Q_0 + L^{**}Q_0, \quad Q_3 = Q_0 + L^{***}Q_0 , \qquad /25/$$

gdzie znaki *, ** i *** oznaczają odpowiednie operatory: jednoparametrowe, dwuparametrowe i trójparametrowe wybrane wg relacji /17/ /dla siatki $G_{6s}(h^3)$ wyznacza się jedynie operatory jednoparametrowe/.

Sumy modułów funkcji P i Q w trójce węzłów wylicza się wtedy wg wzorów:

$$\rho_1 = |P_1| + |Q_1|$$
$$\rho_2 = |P_2| + |Q_2| \qquad /26/$$
$$\rho_3 = |P_3| + |Q_3|$$

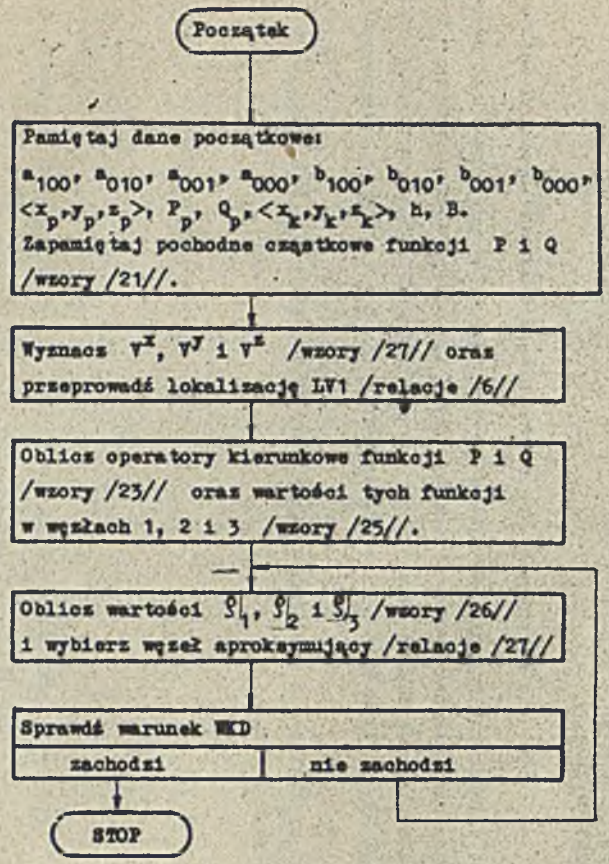
Na kolejny punkt aproksymujący wybierany jest ten z węzłów, dla którego zachodzi M=min lub dodatkowo, który ma również wyższy priorytet /zgodnie z ZWW - relacje /14'/ /:

- jeżeli zachodzi $(\rho_1 < \rho_2) \wedge (\rho_1 < \rho_3) = 1$ to punkt $P(\rho_1)$
- jeżeli zachodzi $(\rho_1 \geq \rho_2) \wedge (\rho_2 < \rho_3) = 1$ to punkt $P(\rho_2)$ /27/
- w przeciwnym wypadku punkt $P(\rho_3)$.

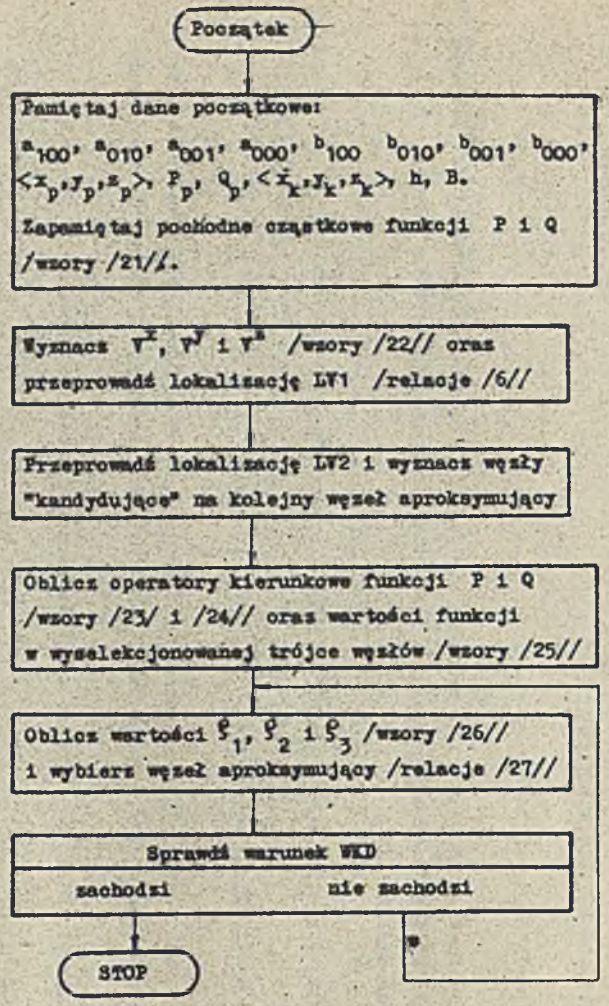
Pochodne funkcji P i Q są stałe. Nie ma potrzeby ich uaktualniania w nowo osiągniętym węźle.

Kolejną operacją jest sprawdzenie warunku WKD dyskretyzowanego odcinka: gdy wynik jest negatywny - przejście do wyznaczenia kolejnego węzła aproksymującego, tj. do operacji wyznaczania wartości funkcji w trójce węzłów, o tych samych numerach co poprzednio /wzory /25/ /, następnie obliczanie sumy modułów funkcji P i Q w tych węzłach /wzory /26/ / oraz sprawdzenie mniejszego modułu /relacje /27/ /.

Szczegółowe algorytmy dyskretyzacji prostej w przestrzeni na siatkach $G_{6s}(h^3)$ i $G_{26s}(h^3)$ przedstawione zostały na rys.5 i rys.6.



Rys. 5. Algorytm dyskretyzacji prostej przestrzennej na siatce $G_{6s}(h^3)$



Rys. 6. Algorytm dyskretyzacji prostej przestrzennej na siatce $G_{26s}(h^3)$

Algorytm dla płaskiej krzywej 2-stopnia określonej w przestrzeni

W przypadku krzywej przestrzennej płaskiej 2-stopnia algorytmy dyskretyzacji na siatkach $G_{6s}(h^3)$ i $G_{26s}(h^3)$ są bardziej złożone niż w przypadku prostej. Należy bowiem aktualizować w każdej pętli algorytmu współrzędne wektora stycznej, kierunki elementarnych ruchów wzdłuż głównych linii siatki oraz pochodne cząstkowe I-rzędu funkcji definiujących powierzchnię 2-stopnia. Niemniej jednak dla h=1 algorytmy dają się sprowadzić do takich postaci, w których, w pętlach algorytmów występują jedynie proste operacje arytmetyczne sumowania i komparacji modułów liczb oraz operacji na argumentach logicznych.

Niech krzywa w przestrzeni będzie krawędzią przecinania się powierzchni 2-stopnia oraz płaszczyzny, opisanych równaniami uwikłanymi:

$$P(x,y,z) = a_{200}x^2 + a_{110}xy + a_{101}xz + a_{020}y^2 + a_{011}yz + a_{002}z^2 + a_{100}x + a_{010}y + a_{001}z + a_{000} = 0 \qquad /28/$$

$$Q(x,y,z) = b_{100}x + b_{010}y + b_{001}z + b_{000} = 0 .$$

Wzory /11/ i /11'/ na pochodne cząstkowe funkcji P i Q, które dla rozważanego przypadku krzywej płaskiej 2-stopnia w przestrzeni przyjmują postać /29/, zawierają operacje mnożeń:

$$\begin{aligned} P_x &= 2a_{200}x + a_{110}y + a_{101}z + a_{100}, & P_{2x} &= 2a_{200} \\ P_y &= a_{110}x + 2a_{020}y + a_{011}z + a_{010}, & P_{2y} &= 2a_{020} \\ P_z &= a_{101}x + a_{011}y + 2a_{002}z + a_{001}, & P_{2z} &= 2a_{002} \\ P_{xy} &= a_{110}, & P_{xz} &= a_{101}, & P_{yz} &= a_{011} \\ Q_x &= b_{100}, & Q_y &= b_{010}, & Q_z &= b_{001} . \end{aligned} \qquad /29/$$

Pochodne cząstkowe wyższych rzędów obydwu funkcji P i Q tożsamościowo równe są zero.

Wektor kierunkowy stycznej $\vec{V}$, będący iloczynem wektorowym normalnych do powierzchni P i Q, zmienia się więc w każdym kroku dyskretyzacji. Składowe tego wektora V^x, V^y i V^z wyznaczymy w początkowej fazie algorytmów ze wzorów definicyjnych /5/. Wymaga to, jak widać, wykonania operacji mnożeń, których w algorytmie staramy się uniknąć. Wykorzystując zatem fakt, że pochodne cząstkowe składowych wektora stycznej stopnia od drugiego wzwyż są tożsamościowo równe zero, rozkładamy je w szereg Taylora w otoczeniu punktu bieżącego. Wykorzystując następnie to rozwinięcie aktualizujemy pochodne w nowo wyznaczonych punktach-węzłach. Korzystamy przy tym z następujących wzorów, w których m.in. argumentami mnożenia są wielkości Δx, Δy i Δz:

$$V^x(x+\Delta x,y+\Delta y,z+\Delta z) = V^x(x,y,z) + \Delta x\, V^x_x(x,y,z) + \Delta y\, V^x_y(x,y,z) + \Delta z\, V^x_z(x,y,z) \qquad /30/$$

$$V^y(x+\Delta x,y+\Delta y,z+\Delta z) = V^y(x,y,z) + \Delta x\, V^y_x(x,y,z) + \Delta y\, V^y_y(x,y,z) + \Delta z\, V^y_z(x,y,z) \qquad /31/$$

$$V^z(x+\Delta x,y+\Delta y,z+\Delta z) = V^z(x,y,z) + \Delta x\, V^z_x(x,y,z) + \Delta y\, V^z_y(x,y,z) + \Delta z\, V^z_z(x,y,z) .$$

Pochodne cząstkowe I-rzędu składowych V^x, V^y i V^z wektora stycznej - stałe, wystarczy wyznaczyć jedynie na początku algorytmu, wykorzystując poniższe wzory:

$$V^x_x = a_{110}b_{001} - a_{101}b_{010}, \quad V^x_y = 2a_{020}b_{001} - a_{011}b_{010}, \quad V^x_z = a_{001}b_{001} - 2a_{002}b_{010}$$

$$V^y_x = a_{101}b_{100} - 2a_{200}b_{001}, \quad V^y_y = a_{011}b_{100} - a_{110}b_{001}, \quad V^y_z = 2a_{002}b_{100} - a_{101}b_{001} \qquad /32/$$

$$V^z_x = 2a_{200}b_{010} - a_{110}b_{100}, \quad V^z_y = a_{110}b_{010} - 2a_{020}b_{100}, \quad V^z_z = a_{101}b_{010} - a_{011}b_{100} .$$

Prostota wyznaczenia składowych wektora stycznej za pomocą powyższych wzorów wynika z faktu, że wartości Δx, Δy i Δz przyrostów współrzędnych bieżącego punktu mogą przyjmować wartość $\pm h$ i 0, a h=1 /zwykle/. Stąd też występujące we wzorach /32/ mnożenia sprowadza się co najwyżej do zmiany znaku lub wyzerowania odpowiedniego składnika sumy po prawej stronie każdego ze wzorów /32/.

Aktualizację kierunków elementarnych ruchów należy również przeprowadzać w pętlach algorytmów. Skorzystać tutaj można z relacji /6/ słusznych dla przypadku ogólnego krzywych przestrzennych. Relacje te zawierają bowiem proste operacje testowania znaków liczb oraz operacje sumowań i mnożeń na argumentach logicznych. Są więc operacjami prostymi w realizacji.

Przeprowadzić należy również lokalizację LV2 /relacji /12/ /, która wymagana jest na każdym kroku dyskretyzacji.

Wyznaczenie operatorów kierunkowych funkcji P i Q, niezbędnych do określenia tych funkcji w punktach wynikających z lokalizacji LV2, należy przeprowadzić wykorzystując wzory /33/, będące odpowiednikami wzorów /15/ i /11/ dla n=2 i m=1:

$$L^xP = \Delta x P_x + \frac{(\Delta x)^2}{2} P_{2x}, \quad L^yP = \Delta y P_y + \frac{(\Delta y)^2}{2} P_{2y}, \quad L^zP = \Delta z P_z + \frac{(\Delta z)^2}{2} P_{2z}$$

$$L^xQ = \Delta x Q_x, \quad L^yQ = \Delta y Q_y, \quad L^2Q = \Delta z Q_z \qquad /33/$$

$$L^{Rxy}P = \Delta x \Delta y P_{xy}, \quad L^{Rxz}P = \Delta x \Delta z P_{xz}, \quad L^{Ryz}P = \Delta y \Delta z P_{yz}$$

$$L^{Rxyz} = 0, \quad L^{Rxy}Q = L^{Rxz}Q = L^{Ryz}Q = 0 .$$

Aktualizację pochodnych cząstkowych funkcji P i Q w nowo wyznaczonym punkcie można natomiast przeprowadzić wykorzystując wzory /34/:

$$L^xP_x = \Delta x P_{2x}, \quad L^xP_y = \Delta x P_{xy}, \quad L^xP_z = \Delta x P_{xz}$$

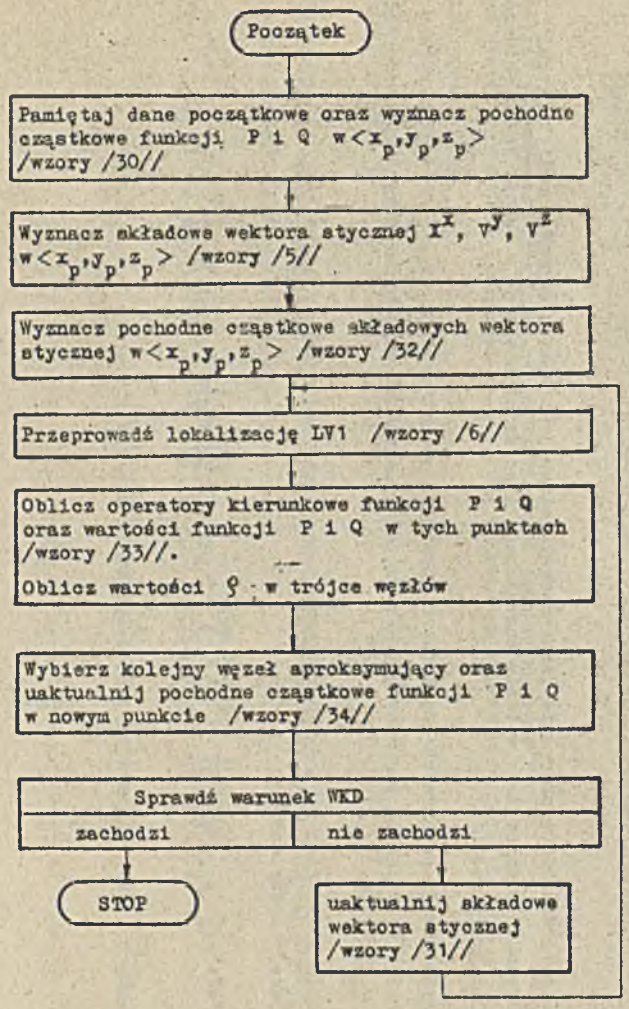
$$L^yP_x = \Delta y P_{xy}, \quad L^yP_y = \Delta y P_{2y}, \quad L^yP_z = \Delta y P_{yz}$$

$$L^zP_x = \Delta z P_{xz}, \quad L^zP_y = \Delta z P_{yz}, \quad L^zP_z = \Delta z P_{2z} \qquad /34/$$

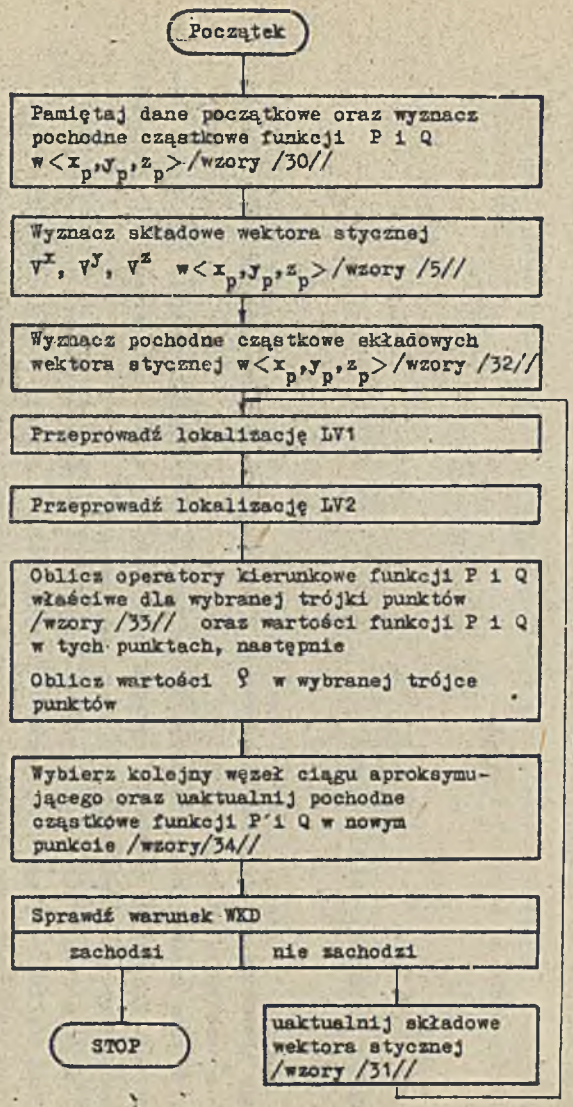
$$L^{Rxy}P_y = L^{Rxz}P_x = L^{Ryz}P_x = L^{Rxyz}P_x = 0$$

$$L^{Rxy}P_y = L^{Rxz}P_y = L^{Ryz}P_y = L^{Rxyz}P_y = 0$$

$$L^{Rxy}P_z = L^{Rxz}P_z = L^{Ryz}P_z = L^{Rxyz}P_z = 0 .$$



Rys. 7. Algorytm dyskretyzacji krzywej przestrzennej płaskiej 2-stopnia na siatce $G_{6s}(h^2)$



Rys.8. Algorytm dyskretyzacji krzywej płaskiej 2-stopnia w przestrzeni na siatce $G_{26s}(h^2)$

Pozostałe operatory kierunkowe pochodnych cząstkowych funkcji P oraz wszystkie operatory pochodnych cząstkowych funkcji Q są tożsamościowo równe zero.

Schematy blokowe algorytmów dyskretyzacji płaskich krzywych przestrzennych na siatkach $G_{6s}(h^3)$ i $G_{26s}(h^3)$ przedstawione są na rys.7 i rys.8.

ANALIZA ZŁOŻONOŚCI OBLICZENIOWEJ ALGORYTMÓW

Oszacujemy złożoność obliczeniową opracowanych w poprzednim rozdziale algorytmów jako funkcję stopni n i m zadających krzywą powierzchni.

Załóżmy, że czas trwania każdej operacji w algorytmie jest krotnością trwania określonej operacji elementarnej. Wyznaczymy złożoność każdej z tych operacji, a następnie sumaryczną złożoność wszystkich operacji występujących w pętlach obydwu algorytmów.

Szacując złożoność obliczeniową algorytmów, opierać się będziemy na pojęciu "operacji głównej": oceniać będziemy nie liczbę wszystkich operacji w algorytmach, a jedynie liczbę operacji najbardziej czasochłonnych komputerowo. Sumowanie i inne proste operacje /np. operacje na argumentach logicznych/ celowo pomijamy. Złożoność algorytmów oceniać będziemy więc liczbą niezbędnych do wykonania operacji mnożeń oraz operacji potęgowania, dzielenia i operacji silnia, sprowadzonych do odpowiedniej liczby mnożeń.

Obliczanie złożoność algorytmów rozpoczniemy od obliczenia złożoności wyznaczenia pochodnych cząstkowych w punkcie początkowym, chociaż operacje te, występując poza pętlą algorytmu, wykonywane są tylko jeden raz, niezależnie od długości dyskretyzowanego odcinka krzywej.

Złożoność obliczeniowa wyznaczenia jednego węzła ciągu dla h≠1

Złożoność obliczeniowa wyznaczenia jednego węzła ciągu jest sumą złożoności obliczeniowej operacji wykonywanych w algorytmie w fazie początkowej jego działania, oraz w czasie jednokrotnego przebiegu pętli algorytmu.

1/ Złożoność wyznaczenia pochodnych cząstkowych w punkcie początkowym $<x_p, y_p, z_p>$ wyznaczymy jako sumę złożoności wyznaczenia pochodnych względem jednej, dwóch i trzech zmiennych /wg wzoru /16/ / dla obydwu funkcji P i Q:

- Wyznaczenie wszystkich pochodnych względem jednej zmiennej wymaga /dla funkcji P/ wykonania następującej liczby mnożeń:

$$L_j^0\left(L^{x,y,z}{}_P(x_p,y_p,z_p)\right) = \sum_{r=1}^{n}\sum_{i=0}^{n-r}\sum_{j=r}^{n-i}\sum_{k=0}^{n-i-j}(n-1) = \frac{n^5}{30} + \frac{5}{24}n^4 + \frac{5}{12}n^3 + \frac{7}{24}n^2 + \frac{n}{20} \quad ;$$

dla zmiennych x,y,z liczba ta jest trzykrotnie większa i wynosi:

$$L_j^0\left(L^{x}{}_P + L^{y}{}_P + L^{z}{}_P\right) = \frac{n^5}{10} + \frac{5}{8}n^4 + \frac{5}{4}n^3 + \frac{7}{8}n^2 + \frac{3}{20}n \quad .$$

- Wyznaczenie pochodnych mieszanych względem dwóch zmiennych, dla jednej pary zmiennych wymaga wykonania liczby mnożeń wyrażającej się wzorem:

$$L_m^0\left(L^{xy,xz,yz}P(x_p,y_p,z_p)\right) = \sum_{r=1}^{n-1}\sum_{s=1}^{n-r}\sum_{i=0}^{n-r-s}\sum_{j=r}^{n-i-s}\sum_{k=s}^{n-i-j}(n-1) =$$

$$= \frac{n^6}{144} + \frac{3}{80}n^5 + \frac{7}{144}n^4 - \frac{n^3}{48} - \frac{n^2}{18} - \frac{n}{60} \quad ;$$

dla par zmiennych xy, xz, yz liczba ta jest trzykrotnie większa i określa się wzorem:

$$L_j^0\left(L^{xy}P+L^{xz}P+L^{yz}P\right) = \frac{n^6}{48} + \frac{9}{80}n^5 + \frac{7}{18}n^4 - \frac{n^3}{16} - \frac{n^2}{6} - \frac{n}{20} \quad .$$

- Wyznaczenie pochodnych mieszanych względem trzech zmiennych wymaga wykonania liczby mnożeń równej

$$L_m^0\left(L^{xyz}P(x_p,y_p,z_p)\right) = \sum_{r=1}^{n-2}\sum_{s=1}^{n-r-1}\sum_{t=1}^{n-r-s}\sum_{i=0}^{n-r-s-t}\sum_{j=r}^{n-s-t-i}\sum_{k=s}^{n-t-i-j}(n-i) =$$

$$= \frac{n^7}{840} + \frac{n^6}{240} - \frac{n^5}{240} - \frac{n^4}{48} - \frac{n^3}{240} + \frac{n^2}{60} + \frac{n}{140} \quad .$$

Łączna liczba niezbędnych do wykonania mnożeń w czasie wyznaczania pochodnych cząstkowych funkcji P w punkcie początkowym $\langle x_p,y_p,z_p\rangle$ wyraża się więc wzorem:

$$L_{\sum P}^0\left(P_{rxsytz}(x_p,y_p,z_p)\right) = \frac{n^7}{840} + \frac{n^6}{40} + \frac{5}{24}n^5 + \frac{3}{4}n^4 + \frac{121}{60}n^3 + \frac{29}{40}n^2 + \frac{15}{140}n \quad , \qquad /35/$$

i nie zależy od rodzaju siatki, na której dokonywana jest dyskretyzacja.
Podobnym wzorem wyraża się złożoność wyznaczania pochodnych cząstkowych funkcji $Q(x_p,y_p,z_p)$ w punkcie początkowym:

$$L_{\sum Q}^0\left(Q_{rxsytz}(x_p,y_p,z_p)\right) = \frac{m^7}{840} + \frac{m^6}{40} + \frac{5}{24}m^5 + \frac{3}{4}m^4 + \frac{121}{60}m^3 + \frac{29}{40}m^2 + \frac{15}{140}m \quad .$$

Tak więc złożoność wyznaczenia pochodnych cząstkowych w punkcie początkowym krzywej wymaga wykonania liczby mnożeń określonej następującym wzorem:

$$L_{\sum}^0\left(P_{rxsytz} + Q_{rxsytz}(x_p,y_p,z_p)\right) = \frac{n^7+m^7}{840} + \frac{n^6+m^6}{40} + \frac{5}{24}(n^5+m^5) + \frac{3}{4}(n^4+m^4) +$$

$$+ \frac{121}{60}(n^3+m^3) + \frac{29}{40}(n^2+m^2) + \frac{15}{140}(n+m). \qquad /36/$$

2/ Wyznaczenie składowych wektora stycznej V^x, V^y i V^z /relacje /5// wymaga wykonania dwóch mnożeń dla każdej ze składowych, niezależnie od stopnia obydwu powierzchni P i Q. Tak więc

$$L^0_v(V^x+V^y+V^z) = 6 \quad . \qquad /37/$$

3/ Wyznaczenie elementarnych kierunków ruchu wymaga wykonania jedynie prostych operacji na argumentach logicznych. Podobnie lokalizacja wektora $\vec{V}$ nie wymaga wykonywania mnożeń lub innych, podobnie złożonych operacji arytmetycznych.

4/ Obliczanie modułów wartości funkcji P wymaga wyznaczenia wszystkich operatorów jednoparametrowych L^x, L^y i L^x, wszystkich operatorów różnicowych dwuparametrowych L^{Rxy}, L^{Rxz}, L^{Ryz} oraz operatora różnicowego trójparametrowego L^{Rxyz} /operatory różnicowe oblicza się tylko dla siatki $G_{26s}(h^3)$/. Operacje te wymagają wykonania następującej liczby mnożeń:

- Wyznaczenie dowolnego z operatorów jednoparametrowych /w przypadku funkcji P/ wymaga wykonania liczby mnożeń

$$L^0_\ell\left(L^{x,y,z}\,P\right) = \sum_{i=1}^{n} 2i = n^2 + n \quad .$$

- Wyznaczenie dowolnego z operatorów różnicowych dwuparametrowych /w przypadku funkcji P/ wymaga wykonania liczby mnożeń

$$L^0_\ell\left(L^{Rxy,Rxz,Ryz}\,P\right) = \sum_{i=1}^{n-1}\sum_{j=1}^{i} 2(i+1) = \frac{2}{3}n^3 - \frac{2}{3}n \quad .$$

- Wyznaczenie operatora różnicowego trójparametrowego $L^{Rxyz}P$ wymaga wykonania liczby mnożeń

$$L^0_\ell\left(L^{Rxyz}P\right) = \sum_{i=1}^{n-1}\sum_{j=1}^{i}\sum_{k=1}^{j} 2(i+2) = \frac{n^4}{4} + \frac{n^3}{2} + \frac{n^2}{4} - \frac{n}{2} \quad .$$

Wyznaczenie wszystkich operatorów funkcji P niezbędnych do obliczenia modułu funkcji w trójce węzłów wymaga więc wykonania liczby mnożeń określonej wzorem:

$$L^0_\ell\left(\sum L^{(\cdot)}\,P26s\right) = \frac{n^4}{4} + \frac{5}{2}n^3 + \frac{11}{4}n^2 + \frac{n}{2} \qquad /38/$$

dla siatki $G_{26s}(h^3)$, oraz wzorem:

$$L^0_\ell\left(\sum L^{(\cdot)}\,P6s\right) = 3(n^2+n) \qquad /39/$$

dla przypadku siatki $G_{6s}(h^3)$.

5/ Złożoność wyznaczenia operatorów pochodnych cząstkowych, niezbędnych do aktualizacji tych pochodnych w nowo wyznaczonym punkcie, określimy poniższymi wzorami:

- Złożoność wyznaczenia operatorów jednoparametrowych dla pochodnych wg jednej /dowolnej/ zmiennej wyraża się wzorem

$$L^0_\ell(\Sigma L^{x,y,z}P_{rx}) = \sum_{r=1}^{n-1}\sum_{i=1}^{n-r} 2i = \frac{1}{3}(n^3-n).$$

Dla zmiennych x, y i z /tzn. wyznaczenie $\Sigma L^x P_{rx}$, $\Sigma L^x P_{sy}$ oraz $\Sigma L^x P_{tz}$/ złożoność wyraża się liczbą trzykrotnie większą, określoną za pomocą wzoru:

$$L^0_\ell(\Sigma L^x P_{rx} + \Sigma L^y P_{sy} + \Sigma L^z P_{tz}) = n^3 - n$$

/dla n=5 $L^0_\ell = 120$/.

- Złożoność wyznaczenia operatorów jednoparametrowych L^x, L^y lub L^z dla pochodnych względem dwóch zmiennych wyraża się wzorem

$$L^0_\ell(\Sigma L^x P_{rxsy}) = \sum_{r=1}^{n-2}\sum_{s=1}^{n-1-r}\sum_{i=1}^{n-1-s} 2i = \frac{n^4}{12} - \frac{n^3}{6} - \frac{n^2}{12} + \frac{n}{6}.$$

Dla par zmiennych xy, xz, yz złożoność określa się liczbą krzykrotnie większą:

$$L^0_\ell(\Sigma L^x P_{rxsy} + \Sigma L^x P_{rxtz} + \Sigma L^x P_{sytz}) = \frac{n^4}{n} - \frac{n^3}{z} - \frac{n^2}{n} + \frac{n}{z}$$

/dla n=5 L^0=90/.

- Złożoność wyznaczenia operatorów jednoparametrowych dla pochodnych względem trzech zmiennych wyraża się wzorem:

$$L^0_\ell(\Sigma L^x P_{rxsytz}) = \sum_{r=1}^{n-3}\sum_{s=1}^{n-r-2}\sum_{t=1}^{n-r-s-1}\sum_{i=1}^{n-r-s-t} 2i = \frac{n^5}{60} - \frac{n^4}{12} + \frac{n^3}{12} + \frac{n^2}{12} - \frac{n}{10}$$

/dla n=5 $L^0_\ell = 12$/.

Łączna liczba mnożeń, które należy wykonać aktualizując pochodne cząstkowe w nowo wyznaczonym punkcie, po przemieszczeniu wykonanym wzdłuż jednej z linii GLG wyraża się wzorem:

$$L^0_\ell(\Sigma L^x P_{rxsytz}) = \frac{n^5}{60} + \frac{11}{12}n^4 - \frac{23}{12}n^3 - \frac{11}{12}n^2 + \frac{19}{10}n \qquad /40/$$

/dla n=5 $L^0_\ell = 222$/.

- Złożoność aktualizacji pochodnych cząstkowych, gdy przemieszczenie odbywało się wzdłuż linii 2PLG /operatory dwuparametrowe/, określa się poniższym wzorem:

- aktualizacja pochodnych względem jednej zmiennej wymaga wykonania liczby mnożeń określonej przez

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rx}) = \sum_{r=1}^{n-1}\sum_{i=1}^{n-r}\sum_{j=0}^{i} 2i = \frac{n^4}{6} + \frac{n^3}{3} - \frac{n^2}{6} - \frac{n}{3}$$

/dla n=5 $L_{\ell}^{0} = 140$/;

- aktualizacja pochodnych dla wszystkich trzech zmiennych wymaga wykonania liczby mnożeń określonej za pomocą wzoru:

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rx} + \Sigma L^{xy}P_{sy} + \Sigma L^{xy}P_{tz}) = \frac{n^4}{2} + n^3 - \frac{n^2}{2} - n$$

/dla n=5 $L_{\ell}^{0} = 420$/;

- aktualizacja pochodnych względem dwóch zmiennych wymaga wykonania liczby operacji określonej za pomocą wzoru:

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rxsy}) = \sum_{i=1}^{n-2}\sum_{s=1}^{n-1-r}\sum_{i=1}^{n-r-s}\sum_{j=0}^{i} 2i = \frac{n^5}{30} - \frac{n^3}{6} + \frac{2}{15}n$$

/dla n=5 $L_{\ell}^{0} = 84$/;

dla przemieszczeń wzdłuż linii 2PLG złożoność obliczeń wyraża się liczbą trzykrotnie większą, określoną za pomocą wzoru:

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rxsy} + \Sigma L^{xy}P_{rstz} + \Sigma L^{xy}P_{sytz}) = \frac{n^5}{10} - \frac{n^3}{2} + \frac{2}{5}n$$

/dla n=5 $L_{\ell}^{0} = 252$/;

- aktualizacja pochodnych względem trzech zmiennych wymaga wykonania liczby mnożeń określonej za pomocą wzoru:

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rxsytz}) = \sum_{r=1}^{n-3}\sum_{s=1}^{n-r-2}\sum_{t=1}^{n-r-s-1}\sum_{i=1}^{n-r-s-t}\sum_{j=0}^{i} 2i = \frac{n^5}{180} - \frac{n^5}{60} - \frac{n^4}{36} + \frac{n^3}{12} + \frac{n^2}{45} - \frac{n}{16}$$

/dla n=5 $L_{\ell}^{0} = 28$/;

Złożoność aktualizacji pochodnych cząstkowych, w wypadku ruchu wzdłuż dwóch współrzędnych, określa się więc następującym wzorem:

$$L_{\ell}^{0}(\Sigma L^{xy}P_{rxsytz}) = \frac{n^6}{180} + \frac{n^5}{12} + \frac{17}{36}n^4 + \frac{7}{12}n^3 - \frac{43}{90}n^2 - \frac{2}{3}n \qquad /41/$$

/dla n=5 $L_{\ell}^{0} = 700$/.

- Złożoność aktualizacji pochodnych cząstkowych w wypadku, gdy przemieszczenie odbywało się wzdłuż linii 3PIG /operatory trójparametrowe/, określa się następującymi wzorami:

- aktualizacja pochodnych względem jednej zmiennej:

$$L^0_\ell(\Sigma L^{xyz}P_{rx}) = \sum_{r=1}^{n-1}\sum_{i=1}^{n-r}\sum_{j=0}^{i}\sum_{k=0}^{j} 2i = \frac{n^5}{20} + \frac{n^4}{4} + \frac{n^3}{4} - \frac{n^2}{4} - \frac{3}{10}n$$

/dla $n=5$ $L^0_\ell = 336$/;

dla zmiennych x, y i z złożoność aktualizacji tych pochodnych określa się liczbą trzykrotnie większą:

$$L^0_\ell(\Sigma L^{xyz}P_{rx,sy,tz}) = \frac{3}{20}n^5 + \frac{3}{4}n^4 + \frac{3}{4}n^3 - \frac{3}{4}n^2 - \frac{9}{10}n \quad ;$$

- aktualizacja pochodnych mieszanych dla pary dwóch zmiennych wymaga wykonania liczby mnożeń określonej wzorem:

$$L^0_\ell(\Sigma L^{xyz}P_{rxsy}) = \sum_{r=1}^{n-2}\sum_{s=1}^{n-1-r}\sum_{i=1}^{n-r-s}\sum_{j=0}^{i}\sum_{k=0}^{j} 2i = \frac{n^6}{120} + \frac{n^5}{40} - \frac{n^4}{24} - \frac{n^3}{8} + \frac{n^2}{30} + \frac{n}{10}$$

/dla $n=5$ $L^0_\ell = 168$/;

dla par zmiennych xy, xz oraz yz złożoność określa się liczbą trzykrotnie większą, wyznaczoną wg wzoru:

$$L^0_\ell(\Sigma L^{xyz}P_{rxsy,rxtz,sytz}) = \frac{n^6}{40} + \frac{3}{40}n^5 - \frac{1}{8}n^4 - \frac{3}{8}n^3 + \frac{n^2}{10} + \frac{3}{10}n \quad ;$$

- aktualizacja pochodnych mieszanych względem wszystkich trzech zmiennych wymaga wykonania liczby mnożeń określonej wzorem:

$$L^0_\ell(\Sigma L^{xyz}P_{rxsytz}) = \sum_{r+1}^{n-3}\sum_{s=1}^{n-2-r}\sum_{t=1}^{n-1-r-s}\sum_{i=1}^{n-r-s-t}\sum_{j=0}^{i}\sum_{k=0}^{j} 2i = \frac{n^7}{840} - \frac{n^5}{60} + \frac{7}{120}n^3 - \frac{3}{70}n$$

/dla $n=5$ $L^0_\ell = 48$/.

Złożoność aktualizacji pochodnych cząstkowych w wypadku ruchu wzdłuż linii 3PIG określa się zatem następującym wzorem:

$$L^0_\ell(\Sigma L^{xyz}P_{rxsytz}) = \frac{n^7}{840} + \frac{n^6}{40} + \frac{5}{24}n^5 + \frac{5}{8}n^4 + \frac{13}{30}n^3 - \frac{13}{20}n^2 - \frac{9}{14}n \; . \qquad /42/$$

Podobnymi wzorami określa się złożoność wyznaczenia odpowiednich funkcji i aktualizacja pochodnych funkcji Q.

Złożoność przebiegu jednokrotnego algorytmu /jednego przebiegu pętli algorytmu/ wyraża się zatem następującymi wyrażeniami:

1/ w przypadku algorytmu dyskretyzacji na siatce $G_{6s}(h^3)$:

$$L_{\ell}^{p}(6s) = \frac{1}{60}(n^5+m^5) + \frac{1}{6}(n^4+m^4) + \frac{7}{12}(n^3+m^3) + \frac{17}{6}(n^2+m^2) + \frac{12}{5}(n+m) + 6, \qquad /43/$$

2/ w przypadku algorytmu na siatce $G_{26s}(h^3)$:

- gdy przemieszczenie odbywało się wzdłuż GLG:

$$L_{\ell}^{p}(26s1p) = \frac{1}{60}(n^5+m^5) + \frac{7}{6}(n^4+m^4) + \frac{7}{12}(n^3+m^3) + \frac{11}{6}(n^2+m^2) + \frac{12}{5}(n+m) + 6, \qquad /44/$$

- gdy przemieszczenie odbywało się wzdłuż linii 2PLG /wzdłuż dwóch współrzędnych xy, xz bądź yz/:

$$L_{\ell}^{p}(26s2p) = \frac{1}{180}(n^6+m^6) + \frac{1}{12}(n^5+m^5) + \frac{13}{18}(n^4+m^4) + \frac{37}{12}(n^3+m^3) + \frac{400}{180}(n^2+m^2) - \frac{n+m}{6} + 6, \qquad /45/$$

- gdy przemieszczenie odbywało się wzdłuż linii 3PLG /wzdłuż trzech współrzędnych x, y i z/:

$$L_{\ell}^{p}(26s3p) = \frac{1}{840}(n^7+m^7) + \frac{1}{40}(n^6+m^6) + \frac{5}{24}(n^5+m^5) + \frac{7}{8}(n^4+m^4) + \frac{44}{15}(n^3+m^3) + \frac{21}{10}(n^2+m^2) + \frac{1}{7}(n+m) + 6 . \qquad /46/$$

Złożoność obliczeniowa wyznaczenia jednego węzła ciągu w przypadku h=1

W przypadku h=1 złożoność poszczególnych operacji algorytmu jest prawie dwukrotnie mniejsza, chociaż odpowiednie wzory przyjmują podobną postać. Dla poszczególnych operacji wzory te są następujące:

1/ Złożoność obliczeniowa wyznaczenia pochodnych cząstkowych w punkcie początkowym nie zależy od wielkości $\bar{h}$, tzn. jest jednakowa tak dla przypadku $h \neq 1$, jak i $h=1$ i wyraża się wzorem /4.1/.

2/ Również wyznaczenie wektora $\vec{V}$ wymaga wykonania /niezmienionej liczby/ 6 operacji mnożeń.

3/ Wyznaczenie elementarnych kierunków ruchu i lokalizacja wektora $\vec{V}$ nie wymagają wykonania mnożeń lub innych, podobnie złożonych operacji.

4/ Wyznaczenie operatorów funkcji P/Q/ niezbędnych do wyznaczenia modułów tej funkcji w trójce węzłów wymaga wykonania operacji określających się wzorami:

- wyznaczenie dowolnego z operatorów jednoparametrowych

$$L_1^{O}(L^{x,y,z}\ p) = \sum_{i=1}^{n} i = \frac{n^2+n}{2};$$

- wyznaczenie dowolnego z operatorów różnicowych dwuparametrowych

$$L_1^0({}_L R_{xy}, R_{xz}, R_{yz}\ {}_P) = \sum_{i=1}^{n-1} \sum_{j=1}^{i} (i+1) = \frac{n^3}{3} - \frac{n}{3} \ ;$$

- wyznaczenie operatora różnicowego trójparametrowego:

$$L_1^0({}_L R_{xyz}{}_P) = \sum_{i=1}^{n-1} \sum_{j=1}^{i} \sum_{k=1}^{j} (i+2) = \frac{n^4}{8} + \frac{n^3}{4} - \frac{n^2}{8} - \frac{n}{4} \ ,$$

Wyznaczenie wszystkich operatorów funkcji P, niezbędnych do obliczenia modułu tej funkcji w trójce węzłów, wymaga więc wykonania liczby mnożeń określonej wzorem:

$$L_1^0\left(L^{(\cdot)}{}_{P26s}\right) = \frac{n^4}{8} + \frac{5}{4} n^3 + \frac{11}{8} n^2 + \frac{n}{4} \qquad /47/$$

dla siatki $G_{26s}(h^3)$, oraz wzorem:

$$L_1^0\left(L^{(\cdot)}{}_{P6s}\right) = \frac{3}{2} n^2 + n \qquad /48/$$

dla siatki $G_{6s}(h^3)$.

5/ Złożoność wyznaczania operatorów pochodnych cząstkowych, niezbędnych do aktualizacji tych pochodnych w nowo wyznaczonym punkcie dla ruchu wzdłuż linii GLG, jest również dwukrotnie mniejsza. Wzory na złożoność wyznaczenia poszczególnych operatorów są następujące:

- dla operatorów jednoparametrowych i pochodnych wg jednej /dowolnej/ zmiennej

$$L_1^0\left(\Sigma L^{x,y,z}{}_{P_{rx}}\right) = \sum_{r=1}^{n-1} \sum_{i=1}^{n-r} i = \frac{1}{6} n^3 - n \ ,$$

- dla operatorów jednoparametrowych i pochodnych wg dwóch zmiennych

$$L_1^0\left(\Sigma L^{x}{}_{P_{rxsy}}\right) = \sum_{r=1}^{n-2} \sum_{s=1}^{n-1-r} \sum_{i=1}^{n-r-s} i = \frac{n^4}{24} - \frac{n^3}{12} - \frac{n^2}{24} + \frac{n}{12} \ ,$$

- dla operatorów jednoparametrowych i pochodnych wg trzech zmiennych

$$L_1^0\left(\Sigma L^{x}{}_{P_{rxsytz}}\right) = \sum_{r=1}^{n-3} \sum_{s=1}^{n-2-r} \sum_{t=1}^{n-r-s-1} \sum_{i=1}^{n-r-s-t} i = \frac{n^5}{120} - \frac{n^4}{24} + \frac{n^3}{24} + \frac{n^2}{21} - \frac{n}{20} \ .$$

Złożoność aktualizacji pochodnych cząstkowych po przemieszczeniu wykonanym wzdłuż jednej z linii GLG x, y lub z określa się więc wzorem

$$L_1^0\left(\Sigma L^{x}P_{rxsytz}\right) = \frac{n^5}{120} + \frac{11}{24}n^4 - \frac{23}{24}n^3 - \frac{11}{24}n^2 + \frac{19}{20}n \,. \qquad /49/$$

6/ Gdy przemieszczenie odbywa się wzdłuż linii 2PLG, odpowiednie wzory na złożoność mają następującą postać:

- dla aktualizacji pochodnych wg jednej zmiennej

$$L_1^0\left(\Sigma L^{xy}P_{rx}\right) = \sum_{r=1}^{n-1}\sum_{i=1}^{n-r}\sum_{j=0}^{i} i = \frac{n^4}{12} + \frac{n^3}{6} - \frac{n^2}{6} - \frac{n^2}{12} - \frac{n}{6} \,,$$

- dla aktualizacji pochodnych wg dwóch zmiennych

$$L_1^0\left(\Sigma L^{xy}P_{rxsy}\right) = \sum_{r=1}^{n-1}\sum_{n=1}^{n-1-r}\sum_{k=1}^{n-r-s}\sum_{j=0}^{i} i = \frac{n^5}{60} - \frac{n^3}{12} + \frac{n}{15} \,,$$

- dla aktualizacji pochodnych względem trzech zmiennych

$$L_1^0\left(\Sigma L^{xy}P_{rxsytz}\right) = \sum_{r=1}^{n-3}\sum_{s=1}^{n-2-r}\sum_{t=1}^{n-1-r-s}\sum_{i=1}^{n-r-s-t}\sum_{j=0}^{i} i = \frac{n^6}{360} - \frac{n^5}{120} - \frac{n^4}{72} + \frac{n^3}{24} + \frac{n^2}{90} - \frac{n}{30} \,;$$

Złożoność aktualizacji pochodnych cząstkowych dla przemieszczenia wzdłuż linii 2PLG określa się wzorem:

$$L_1^0\left(\Sigma L^{xy}P_{rxsytz}\right) = \frac{n^6}{360} + \frac{n^5}{24} + \frac{17}{72}n^4 + \frac{7}{24}n^3 + \frac{43}{180}n^2 - \frac{n}{3} \,. \qquad /50/$$

7/ Złożoność aktualizacji pochodnych cząstkowych w wypadku, gdy przemieszczenie odbywało się wzdłuż linii 3PLG określa się poniższymi wzorami:

- dla aktualizacji pochodnych względem jednej zmiennej

$$L_1^0\left(\Sigma L^{xyz}P_{rx}\right) = \sum_{r=1}^{n-1}\sum_{i=1}^{n-r}\sum_{j=0}^{i}\sum_{k=0}^{j} i = \frac{n^5}{40} + \frac{n^4}{8} + \frac{n^3}{8} - \frac{n^2}{8} - \frac{3}{20}n \,;$$

- dla aktualizacji pochodnych względem dwóch zmiennych

$$L_1^0\left(\Sigma L^{xyz}P_{rxsy}\right) = \sum_{r=1}^{n-2}\sum_{s=1}^{n-1-r}\sum_{i=1}^{n-r-2}\sum_{j=0}^{i}\sum_{k=0}^{j} i = \frac{n^6}{240} + \frac{n^5}{80} - \frac{n^4}{48} - \frac{n^3}{16} + \frac{n^2}{60} + \frac{n}{20} \,,$$

● dla aktualizacji pochodnych względem trzech zmiennych

$$L_1^0\left(\Sigma L^{xyz}{}_{P_{rxsytz}}\right) = \sum_{r=1}^{n-3}\sum_{s=1}^{n-2-r}\sum_{t=1}^{n-1-r-s}\sum_{i=1}^{n-r-s-t}\sum_{j=0}^{i}\sum_{k=0}^{j} 1 = \frac{n^7}{1680} - \frac{n^5}{120} + \frac{7}{240}n^3 - \frac{3}{140}n .$$

Złożoność aktualizacji pochodnych cząstkowych w wypadku przemieszczenia wzdłuż linii 3PLG określa się więc wzorem:

$$L_1^0\left(\Sigma L^{xyz}{}_{P_{\Sigma rxsytz}}\right) = \frac{n^7}{1680} + \frac{n^6}{80} + \frac{5}{48}n^5 + \frac{5}{16}n^4 + \frac{13}{60}n^3 - \frac{13}{40}n^2 - \frac{9}{28}n . \qquad /51/$$

Podobnymi wzorami określa się złożoność wyznaczania odpowiednich funkcji i aktualizacja pochodnych dla funkcji Q.

Złożoność obliczeń wykonywanych w trakcie jednokrotnego przebiegu pętli algorytmów można zatem wyznaczyć za pomocą poniższych wzorów:

1/ w przypadku algorytmu dyskretyzacji na siatce $G_{6s}(h^3)$

$$L_1^p(6s) = \frac{1}{120}(n^5+m^5) + \frac{1}{12}(n^4+m^4) + \frac{7}{24}(n^3+m^3) + \frac{7}{12}(n^2+m^2) + \frac{6}{5}(n+m) + 6, \qquad /52/$$

2/ w przypadku algorytmu na siatce $G_{26s}(h^3)$:

● gdy przemieszczenie odbywało się wzdłuż linii OLG:

$$L_1^p(26s1p) = \frac{1}{120}(n^5+m^5) + \frac{7}{12}(n^4+m^4) + \frac{7}{24}(n^3+m^3) + \frac{11}{12}(n^2+m^2) + \frac{6}{5}(n+m) + 6, \qquad /53/$$

● gdy przemieszczenie odbywało się wzdłuż linii 2PLG:

$$L_1^p(26s2p) = \frac{1}{360}(n^6+m^6) + \frac{1}{24}(n^5+m^5) + \frac{13}{36}(n^4+m^4) + \frac{37}{24}(n^3+m^3) + \frac{409}{360}(n^2+m^2) - \frac{n+m}{12} + 6, \qquad /54/$$

● gdy przemieszczenie odbywało się wzdłuż linii 3PLG:

$$L_1^p(26s3p) = \frac{1}{1680}(n^7+m^7) + \frac{1}{80}(n^6+m^6) + \frac{5}{48}(n^5+m^5) + \frac{7}{16}(n^4+m^4) + \frac{22}{15}(n^3+m^3) + \frac{21}{20}(n^2+m^2) + \frac{1}{14}(n+m) + 6. \qquad /55/$$

Tab.1. Złożoność obliczeniowa wyznaczenia jednego węzła ciągu aproksymującego krzywą płaską w przestrzeni trójwymiarowej /m=1/ w funkcji stopnia krzywej n i rzędu spójności siatki dla różnych modułów siatki /h=1, h≠1/

m=1 n	$L^{O}_{6s}(h=1)$	$L^{O}_{26s}(h=1)$	$L^{O}_{6s}(h\neq1)$	$L^{O}_{26s}(h\neq1)$
1	12	12	12	12
2	21	27	33	36
3	42	78	102	129
4	85	205	272	399
5	165	465	629	1059
6	303	933	1304	2490
7	527	1703	2487	5325
8	873	2889	4443	10557
9	1386	4626	7530	19874
10	2121	7071	12219	34824
11	3144	10404	19116	59013
12	4533	14829	28986	96339
13	6379	20575	42779	152285
14	8787	27897	61658	233934
15	11877	37077	87029	350529
16	15785	48425	120573	513581
17	20664	62280	164280	737928
18	26685	79011	220485	1041228
19	34038	99018	291906	1445529
20	42933	122733	381684	1977399
21	53601	150621	493425	2668719
22	66295	183181	631244	3557442
23	81291	220947	799811	4688421
24	98889	264489	1004399	6114309
25	119416	314414	1250934	7896534

Tab.2 Złożoność obliczeniowa wyznaczenia jednego węzła ciągu aproksymującego krzywą leżącą na powierzchni 2-stopnia /m=2/ w funkcji stopnia krzywej n i rzędu spójności siatki, dla różnych modułów siatki /h=1, h≠1/.

m=2 n	$L^{O}_{6s}(h=1)$	$L^{O}_{26s}(h=1)$	$L^{O}_{6s}(h\neq1)$	$L^{O}_{26s}(h\neq1)$
1	21	27	33	36
2	30	42	54	80
3	51	93	123	153
4	94	220	293	428
5	174	480	650	1093
6	312	948	1325	2514
7	536	1718	2508	5349
8	882	2904	4464	10581
9	1395	4641	7551	19698
10	2130	7086	12240	34848
11	3153	10419	19137	59037
12	4542	14844	29007	96363
13	6388	20590	42800	152289
14	8796	27912	61679	233958
15	11886	37092	87050	350583
16	15794	48440	120594	513705
17	20673	62295	164301	737952
18	26694	79026	220506	1041252
19	34047	99033	291927	1445553
20	42942	122748	381705	1977423
21	53610	150636	493446	2668743
22	66304	183196	631265	3557466
23	81300	220962	799832	4688445
24	98898	264504	1004420	6114383
25	119427	314429	1250955	7896558

PODSUMOWANIE I ZAKOŃCZENIE

Z literatury znany jest w zasadzie jeden szczegółowy opis algorytmu dyskretyzacji zadanych algebraicznie linii w trójwymiarowej przestrzeni i dotyczy on dyskretyzacji prostej /Danielsson 1970 r./. Inne, nieliczne opisy metod dyskretyzacji linii przestrzennych są pobieżne i fragmentaryczne, zbyt ogólne na to, aby można było przeprowadzić wzajemne porównanie tych metod. Występują co prawda, odwołania do bardziej szczegółowych źródeł, są to jednak publikacje niedostępne /raporty dla określonych firm bądź instytucji/ czy też opracowania niepublikowane. Nie ma zatem możliwości porównania zaproponowanych algorytmów z algorytmami literaturowymi. Jedyne możliwe porównanie może dotyczyć algorytmu dyskretyzacji prostej, zaproponowanego przez Danielssona, z algorytmami opisanymi w niniejszej pracy. Wypada ono na korzyść tych ostatnich tak ze względu na większą "gładkość" krzywej dyskretnej /w wypadku algorytmu dyskretyzacji na siatce $G_{26s}(h^3)$/, jak też ze względu na liczbę niezbędnych do wykonania operacji w jednym kroku dyskretyzacyjnym /niewielka przewaga algorytmu dyskretyzacyjnego na siatce $G_{26s}(h^3)$ i ok. 1,5 krotna przewaga algorytmu dyskretyzacyjnego na siatce $G_{6s}(h^3)$, a przede wszystkim ze względu na uniwersalność: zaproponowane algorytmy dyskretyzacyjne obejmują dyskretyzację algebraicznych krzywych przestrzennych dowolnego stopnia; algorytm Danielssona - tylko prostą w przestrzeni.

Opisane algorytmy dyskretyzacji są obecnie w stadium implementacji na minikomputerze SM-4 w zapisie zmiennoprzecinkowym pojedynczej precyzji. Będą one kodowane oddzielnie dla prostej, oddzielnie dla krzywych płaskich 2-stopnia /szczególna klasa krzywych przestrzennych/ oraz dla krzywych zwichrowanych wyższych stopni. Poprawność ich działania zostanie sprawdzona na dostatecznie dużej, reprezentatywnej grupie przykładów. Sądzimy, że ta implementacja potwierdzi prawidłowość przeprowadzonych rozważań teoretycznych i poprawność konstrukcji algorytmów. Jej wyniki zostaną opublikowane po zakończeniu tych doświadczeń.

Przeprowadzona ocena złożoności obliczeniowej algorytmów w ich komputerowej realizacji, mierzona liczbą niezbędnych do wykonania operacji na jednostkę długości dyskretyzowanej krzywej w funkcji stopni przecinających się powierzchni /zadających krzywą przestrzenną/, rodzaju siatki oraz wielkości jej modułu, pozwoli ocenić szybkość działania algorytmów będącą, dla wielu z ich zastosowań, jednym z podstawowych parametrów użytkowych tych algorytmów.

LITERATURA

[1] Luh J.Y.S., Królak R.J.: A mathematical model for mechanical part description. Communications of the ACM 1965 t.8 nr 2 s.125-129

[2] Weiss R.A.: BE Vision: A package of IBM 7090 FORTRAN programs to draw ortographic views of combinations of planes and quadric surfaces. Journal of the Association for Computing Machinery 1966 t.13 nr 2 s.194-204

[3] Gomba P.G.: A procedure for detecting intersections of three - dimensional objects. Journal of the Association for Computing Machinery 1968 t.15 nr 3 s.354-366

[4] Danielsson E.: Incremental curve generation. IEEE Transactions on Computers 1970 t.C-19 nr 9 s.783-793

[5] Woon P., Freeman H.: A procedure for generating visible - line projections of solids bouded by

quadric surfaces. Information Processing 1971 s.1120 - 1125

[6] Mahl R.: Visible surface algorithm for quadric paches. IEEE Transactions on Computers 1972 t. C-21 nr 1 s.1-4

[7] Sabin M.A.: A method for displaying the interessection curve of two quadric surfaces. The Computer Journal 1974 t.19 nr 4 s.336 - 338

[8] Braid J.C.: A synthes of solids bounded by many faces. Communications of the ACM 1975 t.18 nr 2 s.209 - 216

[9] Levin J.: A parametric algorithm for drawing pictures of solid objects composed of quadric surfaces. Communications of the ACM 1975 t.19 nr 6 s.555 - 563

[10] Levin J.Z.: Mathematical models for determining the intersections of quadric surfaces. Computer Graphics and Image Processing 1979 t.11 nr 1 s.73 - 78 - 87

[11] Mokrzycki W.: Całkowitoliczbowe dyskretyzacje krzywych algebraicznych na jednorodnych siatkach kwadratowych. Prace naukowo-badawcze IMM 1983 nr 3

A method of algebraic discretization of 3-D space curves on homogeneous rectangular grids

S u m m a r y

In this paper a new method of discretization /intiger approximation/ of algebraic 3-dimensional space curves in form of intersecting surfaces $P(x,y,z)= 0$, $Q(x,y,z)= 0$ is analized.

An implementation to homogeneous space grids $G(h^3)$ discretization is being of the method.

The introduction of the paper presents a review of literature concerning methods and algorithms of 3-D space curve discretization and compares them each other.

In the main past of the paper two new algorithms of discretization/on 6-connective grids $G_{6s}(h^3)$and 26-connective grids $G_{26s}(h^3)$, based on the method above are presented. Three implementation forms of the algorithms: for straight lines, 2-degree flat curves and universal algebraic curves in 3-D space are made.

The computation complexity of each algorithm /as the function of n- and m-degrees of intersecting surfaces/ is evaluated.

Об некотором методе дискретизации алгебраических пространственных кривых на однородных пространственных сетках

Р е з ю м е

В публикации предоставлено новый метод дискретизации /целочисленной аппроксимации/ алгебраических пространственных кривых, заданных в виде двух пересекающихся поверхностей вида $P(x,y,z)= 0$ $Q(x,y,z)= 0$

Суть представляемого метода дискретизации заключается в применении однородных пространственных сеток $G(h^3)$.

В начальной части публикации описывается публицированные до сих пор метода дискретизации пространственных алгебраических кривых, переводя их взаимное сравнения.

В дальнейшей - главной части публикации проанализировано два новые дискретизационные алгоритма /алгоритм дискретизации на 6-связанной сетке $G_{6s}(h^3)$ и алгоритм дискретизации на 26-связанной сетке $G_{26s}(h^3)$ /, разработанные на основе этого метода дискретизации.

Приведено 3 реализационные вида алгоритмов: для прямой и плоской кривой 2-степени в трехмерном пространстве, а также общую форму для пространственных кривых произвольной степени.

Проанализировано тоже вычислительную сложность алгоритмов в функции степени n и m определяющих кривую поверхностей.

Prace naukowo-badawcze Instytutu Maszyn Matematycznych

Nonpreemptive Scheduling for Two-Processor Systems

by Andrzej ROWICKI

The purpose of the paper is to consider an algorithm for nonpreemptive scheduling for two-processor systems with identical processors. Computations submitted to the system are composed of dependent tasks with the same execution times and contain no loops and have only one output. Moreover, the algorithm determines the total execution time of the computation. It has been proved that this algorithm is optimal, that is, the total execution time (schedule length) is minimized. Extensions of the nonpreemptive algorithm are considered both for preemptive case by assumption that all tasks of the computation have the same execution times and for preemptive case provided that all tasks of the computation have the mutually commensurable execution times.

List of symbols

Logical calculus:

$=, \neq, \Rightarrow, \Leftrightarrow, \vee$.

Set theory:

$\in, \notin, \subset, \not\subset, \subsetneq, \cap, -, \cup, \bigcup, \prec, \beta, /$.

Arithmetics:

$+, -, /, <, \leqslant, \sum$.

Greek alphabet:

Γ.

Brackets:

$( ), [\,], <>, \{\}, \| \|, \lceil \rceil$.

Marks:

$*$ asterisk

1. Introduction

Multiprocessor computer systems are potentially very effective in decreasing the computation times of programs. This can be especially important for programs or computations which are executed many times, particularly in real-time applications. In these cases, the problem of optimal scheduling of multiprocessor computer systems is of great importance.

The purpose of this paper is to consider algorithms for optimal scheduling of two-processor systems. We shall consider systems in which there are two identical processors and assume that the computations submitted to the systems are composed of dependent tasks with the same execution time. Moreover, we assume that once a processor is assigned to a task it must work continuously on this task until it is completed. It should be noted that we have assume that the execution time for each task and the precedence relation are known a priori. This means that we only consider the static problem. From a more general point of view it is clear that the static scheduling model we have introduced is most applicable to those cases in which a program or computation is to be executed many times, particularly in real time situations.

As a model of computation we admit a directed connected graph which contains no circuits and has one terminal vertex. This model we can interpret as follows: Only one task of the computation is attached to each vertex of the graph. The attachement of tasks of the computation to the vertices of the graph is such that the order of execution of the tasks is in accordance with the succession relation of the vertices of the graph.

Algorithms for optimal nonpreemptive scheduling of two-processor systems has been presented in [1], [2] and other papers. It has been assumed that all tasks have the same execution time and a model of computation is a directed acyclic graph. The time complexity of the algorithm considered in [1] is $O(n^2)$ and for algorithm considered in [2] is $O(n^4)$ [3]. It turns out [4] that the algorithm considered in [1] construct an optimal schedule only when there are no transitive arcs in the computation. The best known algorithm deleting transitive arcs is of time complexity $O(n^{2,8})$ [5]. And thus if we admit trasitive arcs in a model of computation and want to use the algorithm given in [1] first we must delete transitive arcs using algorithm given in [5] or other one and then we can apply the algorithm given in [1]. For this case the time complexity of this process is at least $O(n^{2,8})$.

In this paper, we shall consider an algorithm for optimal nonpreemptive scheduling of two-processor systems which determines the total execution time and has at most time complexity $O(n^3)$. The algorithm considered is valid for computations admitting transitive arcs and thus has no constraints on the structure of the computation as was in the algorithm considered in [1]. The problem of determining the execution time has been not considered in [1] and [2]. Of course, if scheduling or computation is completed, then we can determine the execution time of computation by counting the number of assignment of processors to the tasks. Using the algorithm presented in this paper, we can determine the execution time before

completing the scheduling. This algorithm is optimal in the sense that the total execution time (schedule length) is minimized. We shall consider an extension of this nonpreemptive algorithm for preemptive case by assumption that all tasks of the computation have the mutually commensurable execution times i.e., we make the same assumption as in [6] . In this case, we assume that is permitted to interrupt any processor at stated times. This means that we admit a limited split of any task. Moreover, we shall consider an extension of this nonpreemptive algorithm for preemptive case by assumption that all tasks of the computation have the same execution times.

Description of the algorithm considered in this paper is strongly dependent on the structure of the computation. This fact allows us to inquire into the structure of the computation and to recognize the limitations in realization of the computation.

The scheduling algorithm considered, for nonpreemptive case, has been programmed in PASCAL for ODRA 1304 small computer [7] . There are given no information about programming and computational results for algorithms presented in [1] and [2] .

The results obtained in this paper are an extension of the results presented in [8] , [9] and [10] . The algorithm considered in this paper is so general that can be extended for computations consisting of tasks having an arbitrary execution times [11] , [12] , [13] , [14] and thus having no constraints on execution time given in [6] and [1] .

2. Basic notions and definitions

Let us consider a graph $G = \langle X , \Gamma , x_0 \rangle$, where X is a finite set of vertices, Γ is a relation defined on the set X and x_0 is a terminal vertex of the graph G, that is such a vertex that $\Gamma(x_0) = \emptyset$, where $\emptyset$ is a void set.

The graph $G = \langle X , \Gamma , x_0 \rangle$ will be called the network if, and only if, it is connected and contains no circuits and has one terminal vertex.

In further considerations we confine ourselves to the networks. In the sequel, the networks will be denoted by a capital S.

A network $S = \langle X', \Gamma' , x'_0$ will be called a reduct of the network $S = \langle X , \Gamma, x_0 \rangle$ if, and only if:

i) $X' \subset X$ and $\Gamma' \subset \Gamma$

ii) if $x \in X'$ then $\Gamma(x) \subset X'$

It is easy to see that if the network $S' = \langle X', \Gamma' , x'_0 \rangle$ is the reduct of the network $S = \langle X , \Gamma , x_0 \rangle$ then $x'_0 = x_0$.

A reduct $S' = \langle X', \Gamma', x'_0 \rangle$ is said to be a direct reduct of the network $S < X , \Gamma , x_0 \rangle$ if, and only if:

i) $1 \leqslant \| X - X' \| \leqslant 2$

ii) $\Gamma (X - X) \subset X'$

where $\| X - X' \|$ denotes the cardinality of the set $X - X'$.

The sequence of networks $S_1,..,S_i...,S_k$ will be called a direct reduction of the network S (in symbols R (S)) if, and only if :

i) $S_1 = S$

ii) for every $i < k$, S_{i+1} is direct reduct of S_i

iii) for S_k there exists no direct reduct

Let $R(S) = S_1, S_2, .., S_k$. The cardinality of the set $\{S_1, S_2, .., S_k\}$ will be called the length of the reduction R(S) and denoted by $d(R(S))$.

Let $L(i)$ for $0 < i \leqslant p$ be a subset of X defined by induction in the following way :

i) $L(p) = \{x_o\}$

ii) if $x \in X - \bigcup_{j=i+1}^{p} L(j)$ and $\Gamma(x) \subset \bigcup_{j=i+1}^{p} L(j)$ then $x \in L(i)$

then the set $L(i)$ will be called i-th level of the network $S = \langle X, \Gamma, x_o \rangle$, where p-1 is the length of the longest path of the network S .

3. A general idea of the scheduling algorithm

To describe the algorithm some notions will be needed. In order to make easier understanding of essential notions we shall first sketch the basic step of the algorithm. To avoid obscuring the basic idea some details will be omitted here. Informally speaking, the heart of the scheduling algorithm consists in searching for sets of the mutually independent vertices of the network in question.

We now proceed to an informal description of the algorithm. Let us assume that i-1 - th step of the algorithm has been executed, which means that a reduct $S_i = \langle X_i, \Gamma_i, x_o \rangle$ of the network $S = \langle X, \Gamma, x_o \rangle$ is created such that

$$\bigcup_{j=1}^{i-1} L(j) \cap X_i = \emptyset$$

In the i-th step of the algorithm we create a reduct S_{i+1} of the network S by removing a set F(i) of vertices from the set X_i of vertices of the reduct S_i . The set F(i) will be choosen by considering the sets

$$\begin{aligned} &L^*(i,0) = L(i) \cap X_i \\ &L^*(i,1) = L(i+1) \cap X_i \\ (1)\quad &L^*(i,2) = L(i+2) \cap X_i \\ &\ldots\ldots\ldots\ldots\ldots \\ &L^*(i,p-i) = L(p) \cap X_i \end{aligned}$$

as follows:

If the cardinality of the set $L^*(i,0)$ is an even number (that is $L^*(i,0) = 2 \cdot f(i)$) then we set $F(i) = L^*(i,0)$. In the case where the cardinality of the set $L^*(i,0)$ is an odd number (that is $L^*(i,0) = 2 \cdot f(i) - 1$) then we consider the following cases :

1. If there exists no vertex $x \in L^*(i,0)$ such that $L^*(i,0) - \Gamma(x) \neq \emptyset$ then we analyze the sequence of the sets (1) until we determine the smallest j satisfying the following relation:

(2) $L^*(i,j) - \bigcup_{n=1}^{j} \Gamma^n(L^*(i,0)) \neq \emptyset$

where Γ^n is n-th superposition of the relation Γ.

If there exists j satisfying the relation (2) then we set

$F(i) = L^*(i,0) \cup \{s_g^*\}$

where s_g^* is a vertex belonging to the set $L^*(i,j) - \Gamma(L^*(i,j-1))$ for which the number of immediate successors in the set $L^*(i,j+1)$ is maximal.

If there exists no j satisfying the relation (2) then we set

$F(i) = L^*(i,0)$

2. If there exists a vertex $x \in L^*(i,0)$ such that $L^*(i,0) - \Gamma(x) \neq \emptyset$ then we consider the following cases:

i) If $\|L^*(i,1)\| = 2$ then we analyse the sequence of the sets (1) until we determine the smallest j for which there exists a vertex $x_i \in L^*(i,0)$ satisfying the relation

(3) $\|L(i,j) \cap \Gamma^j x_i\| \geq 2$

Next, we set

$F(i) = L^*(i,1) \cup \{s_n^*\}$

where $\{s_n^*\} = L^*(i,1) \cap \Gamma x_i$

If there exists no j satisfying the relation (3) then we choose from the set $L^*(i,0)$ a vertex x_j such that $L^*(i,0) - \Gamma(x_j) \neq \emptyset$ and set

$F(i) = L^*(i,1) \cup \{s_n^*\}$

In this case s_n^* is an arbitrary vertex from the set $L^*(i,0) - \Gamma(x_j)$.

ii) If $\|L^*(i,0)\| = 2n + 1$ $(n = 1,2,3,\ldots)$ then we chose a vertex $x_i \in L^*(i,0)$ satisfying the relation $L^*(i,1) - \Gamma(x) \neq \emptyset$ and we set

$F(i) = L^*(i,0) \cup \{s_h^*\}$

where $s_h^* \in L(i,1) - \Gamma(x_i)$

iii) If $\| L^*(i,0) \| = 2n + 2 \quad n = 1,2,3,\ldots$ then we choose from the set $L^*(i,0)$ a subset $H(i)$ of vertices satisfying the relation $L^*(i,1) - \Gamma(x) \neq \emptyset$. From the set $H(i)$ we choose a subset of vertices $H'(i)$ that for every element of this subset there exists such a vertex $y \in L^*(i,1) - \Gamma(x)$ that $L^*(i,2) - \Gamma(y) = \emptyset$. Next, we set

$$F(i) = L^*(i,0) \left\{ s_w^* \right\}$$

where s_w^* is an arbitrary vertex from the set $H'(i)$.

Let x_i be an arbitrary and fixed vertex belonging to the set $H(i)$. In the case where the set $H'(i)$ is empty then s_w is an arbitrary vertex from the set $L^*(i, 1) - \Gamma(x_i)$.

The vertices s_g^* , s_n^* , s_h^* , s_w^* will be called sigular vertices and in order to simplify description of the algorithm the vertices s_n^* , s_h^* and s_w^* will be denoted by s_i^* .

If the set $F(i)$ is determined, then we create a disjoint partition

$$P(F(i)) = P(1,i),\ P(2,i),\ \ldots,\ P(f(i),i)$$

of the set $F(i)$ so as to obtain the following sequence

$$S_i\ ,\ S_{i,1}\ ,\ \ldots,\ S_{i,f(i)},\ S_{i+1}$$

of direct reducts for which

$$X_{i,1} = X_i - P(1,i)\ ,\ X_{i,2} = X_{i,1} - P(2,i) \ldots\ldots\ldots\ldots$$

$$\ldots\ X_{i+1} = X_{i,f(i)-1} - P(f(i),i)$$

If $\| L^*(i,0) \| = 2 \cdot f(i) - 1$ and the vertex s_i^* exists, then we create the disjoint partition $P(F(i))$ in such a way that

$$P(f(i),i) = \left\{ s_i,\ s_i^* \right\}$$

Since S_{i+1} is the direct reduct of $S_{i,f(i)-1}$ this means that s_i is a vertex chosen from the set $L^*(i,0)$ in such a way as to do not violate the precedence relation among the vertices, that is, $s_i^* \notin \Gamma(s_i)$

The disjoint partition $P(F(i))$ determines the sets of mutually independent vertices and thus gives an assignment of the tasks to the processors. The tasks corresponding to the sets $P(j,i)\ (j = 1,2,\ldots,f(i))$ can be concurrently executed

4. Description of the algorithm

To describe the algorithm some notions will be needed. This notions can be defined formally as follows:

Let $S = \langle X\ ,\ \Gamma,\ x_0 \rangle$ be an arbitrary but fixed network and let $S_i = \langle X_i, \Gamma_i\ x_{0i} \rangle$ be the reduct of the network S created as a result of executing i-th step of the scheduling algorithm.

A set $L^*(i,n)$ for $0 < i \leqslant p$ and $0 \leqslant n \leqslant p-i$ is said to be a restriction of the set $L(i)$ if it is defined as follows:

$$L^*(i,n) = L(i+n) - \bigcup_{j=0}^{i-1} F(j)$$

where $F(j)$ is a subset of set X created as a result of executing j-th step of the scheduling algorithm.

In the set $L^*(i,n)$ we distinguish a subset $H(i)$ defined in the following way:

$$H(i) = \left\{ x \in L^*(i,0) : L^*(i,1) - \Gamma(x) \neq \emptyset \right\}$$

Under assumption that $H(i) = \emptyset$, we define a function g in the following way:

$$g(i) = \begin{cases} \text{the smallest } 1 < n \leq p-i \text{ such that} \\ L^*(i,n) - \bigcup_{j=1}^{n} \Gamma^j(L^*(i,0)) \neq \emptyset \\ 1 \qquad \text{otherwise} \end{cases}$$

where Γ^j is a relation defined by recursion as follows:

$$\Gamma^1(x) = \Gamma(x)$$
$$\Gamma^{j+1}x = \Gamma(\Gamma^j(x))$$

Basing on the function g, we define sets $D_g(i)$ and $D'_g(i)$ in the following way:

$$D_g(i) = L^*(i,g(i)) - \bigcup_{j=1}^{g(i)} \Gamma^j(L^*(i,0))$$
$$D'_g(i) = \left\{ x \in D_g(i) : L^*(i,g(i)+1) - \Gamma(x) = \emptyset \right\}$$

It is easy to see that if $g(i) = 1$ then $D_g(i) = \emptyset$. It should be noted that there exist some cases where $D_g(i) = \emptyset$. It should be noted that there exist some cases where $D_g(i) = \emptyset$ and $D'_g(i) = \emptyset$.

For the sake of convenience, we introduce a set $D^*_g(i)$ defined as follows:

$$D^*_g(i) = \begin{cases} D_g(i) & \text{if } D'_g(i) = \emptyset \\ D'_g(i) & \text{if } D'_g(i) \neq \emptyset \end{cases}$$

Now, we distinguish some vertex s^*_g satisfying the relation $s^*_g \in D^*_g(i)$. The vertex s^*_g will be called singular vertex of the network S induced by the function g.

Under assumption that $H(i) \neq \emptyset$, we define a function n in the following way:

$$n(i) = \begin{cases} \text{the smallest } 1 < n \leq p-1 \text{ for which there exists an} \\ x \in L^*(i,0) \text{ such that } \left\| \Gamma^n(x) \cap L^*(i,n) \right\| \geq 2 \\ 1 \qquad \text{otherwise} \end{cases}$$

where $\| X \|$ denotes the cardinality of the set X.

In the set $H(i)$ we distinguish a subset $H_n(i)$ defined as follows:

$$H_n(i) = \left\{ x \in H(i) : L^*(i,n(i)) - \Gamma^{n(i)}(x) \neq \emptyset \right\}$$

It is easy to verify that if $n(i) = 1$ then $H_n(i) = H(i)$.
It should be noted that there exist some cases where $H_n(i) = \emptyset$ and $H(i) = \emptyset$

We distinguish some vertices s_n and s^*_n satisfying the relations $s_n \in H^*_n(i)$ and $s^*_n \in D_n(i)$. The vertices s_n and s^*_n will be called singular vertices of the network S induced by the function n. Now, we define the sets $H^*_n(i)$ and $D_n(i)$ in the following way:

$$H^*_n(i) = \begin{cases} H(i) & \text{if } H_n(i) = \emptyset \\ H_n(i) & \text{if } H_n(i) \neq \emptyset \end{cases}$$

Let us assume that s_n is an arbitrary but fixed vertex belonging to the set $H_n^*(i)$. By this assumption we define the set $D_n(i)$ as follows :

$$D_n(i) = L^*(i,1) - \Gamma(s_n)$$

It is easy to verify that if $H(i) \neq \emptyset$ then $D_n(i) \neq \emptyset$.

Let $H_w(i)$ be a subset of the set $H(i)$ defined as follows:

$$H_w(i) = \left\{ x \in H(i) : (\forall y \in L^*(i,1) - \Gamma(x)) \ (L^*(i,2) - \Gamma(y) = \emptyset) \right\}$$

We distinguish some vertices s_w and s_w^* satisfying the relations $s_w \in H^*(i)$ and $s_w^* \in D_w^*(i)$. The vertices s_w and s_w^* will be called singular vertices of the network S . Now, we define the sets $H_w^*(i)$, $D_w(i)$, $D_w'(i)$ and $D_w^*(i)$ in the following way:

$$H_w^*(i) = \begin{cases} H(i) & \text{if } H_w(i) = \emptyset \\ H_w(i) & \text{if } H_w(i) \neq \emptyset \end{cases}$$

Let s_n be an arbitrary but fixed vertex belonging to the set $H_w^*(i)$. By this assumption we define the sets $D_w(i)$, $D_w'(i)$ and $D_w^*(i)$ as follows :

$$D_w(i) = L^*(i,1) - \Gamma(s_w)$$

$$D_w'(i) = \left\{ x \in D_w(i) : L^*(i,2) - \Gamma(x) = \emptyset \right\}$$

It is easy to verify that if $H_w(i) = \emptyset$ then $D_w'(i) = \emptyset$

$$D_w^*(i) = \begin{cases} D_w(i) & \text{if } H_w(i) = \emptyset \\ D_w(i) & \text{if } H_w(i) \neq \emptyset \end{cases}$$

Moreover, we distiguish some vertices s_h and s_h^* called singular vertices of the network S and satisfying the relations $s_h \in H(i)$ and $s_h^* \in D_h(i)$. Now, we define the set $D_h(i)$ in the following way:

Let us assume that s_h is an arbitrary but fixed vertex belonging to the set $H(i)$. Under this assumption we define the set $D_h(i)$ as follows:

$$D_h(i) = L^*(i,1) - \Gamma(s_h)$$

To formulate the algorithm, we define a function f . For each i ($0 \leqslant i \leqslant p$) the function f associates with the set $L^*(i,0)$ a natural number in the following way:

$$f(i) = \left\lceil \frac{\| L^*(i,0) \|}{2} \right\rceil$$

where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

Let r(i) denote the remainder of the division $\| L^*(i,0) \|$ by 2.

We are now in a position to describe the process of creation the sets F(i). This process will be called an initial sequencing algorithm.

The initial sequencing algorithm is defined by induction in the following way:

1. Basic step.

$F(0) = \emptyset$

2. Induction step.

Case 1.

If $r(i) = 0$ then $F(i) = L^*(i,0)$

Case 2.

If $r(i) \neq 0$ then

a) if $H(i) = \emptyset$ then

i) if $g(i) = 1$ then $F(i) = L^*(i,0)$

ii) if $g(i) \neq 1$ then $F(i) = L^*(i,0) \cup \{s_g^*\}$

b) if $H(i) \neq \emptyset$ then

i) if $\|L^*(i,1)\| = 2$ then $F(i) = L^*(i,0) \cup \{s_n^*\}$

ii) if $\|L^*(i,1)\| = 2n + 1$ $(n = 1,2,3,\ldots)$
then $F(i) = L^*(i,0) \cup \{s_h^*\}$

iii) if $\|L^*(i,1)\| = 2n + 2$ $(n = 1,2,3,\ldots)$
then $F(i) = L^*(i,0) \cup \{s_w^*\}$

For the sake of convenience, the singular vertices s_n^*, s_h^*, s_w^* and s_n, s_h, s_w determined as result of executing i-th step of the initial sequencing algorithm will be denoted by s_i^* and s_i respectively.

As a result of the action of the initial sequencing algorithm we obtain a disjoint partition $\{F(i)\}_{i=1,2,\ldots,p}$ of the set X of the network S, that is, for every $i,j \leq p$ if $i \neq j$ then $F(i) \cap F(j) = \emptyset$ and $\bigcup_{j=1}^{p} F(j) = X$. If there exists no such a number $1 \leq i \leq k$ that $s_i^* \in F(i)$ then the disjoint partition $\{F(i)\}_{i=1,2,\ldots,p}$ determines the sets of mutually independent tasks of the computation which is represented by the network S and thus the tasks corresponding to the sets F(i) can be concurrently executed. On the other hand, if $s_i^* \in F(i)$ then the set $F(i) - \{s_i^*\}$ determines the set of mutually independent tasks of the computation. The tasks corresponding to the vertices s_i and s_i^* can be concurrently executed after completing the tasks corresponding to the set $F(i) - \{s_i, s_i^*\}$. Obviously, the tasks corresponding to the set $F(i) - \{s_i, s_i^*\}$ can be concurrently executed.

What remains now is to solve the problem of assigning the tasks to the different processors, that is, the problem of creation of a disjoint partition of the sets F(i) in such a way so as to minimize the total execution time of the computation and to do not violate the precedence relation among the tasks. This process will be called a partitioning algorithm.

Let the sequence

$$P(F(i)) = P(1,i), \ldots, P(n,i), \ldots, P(k,i)$$

be a fixed disjoint partition of the set F(i) defined as follows:

i) $\|P(j,i)\| = 2$ for $j<k$

ii) $P(k,i) = F(i) - \bigcup_{j=1}^{k-1} P(j,i)$ and $\|P(k,i)\| \leqslant 2$

iii) $\{s_1, s_i^w\} \subset P(k,i)$

To describe the way in which tasks are assigned to the processor, we define now a function q mapping the set X into the set of natural numbers. Let us assume that for each set $F(i)$ (for $0<i\leqslant p$) the disjoint partition $P(F(i))$ is given, provided that this partition exists. The function q assigns to the element of the partition $P(F(i))$ natural numbers in the following way:

$$q(x) = \begin{cases} \sum_{j=1}^{i} f(j) - f(i) + n & \text{for } x \in P(n,i) \text{ if } P(F(i)) \text{ exists} \\ \sum_{j=1}^{i} f(j) & \text{for } x \in F(i) \text{ if no such } P(F(i)) \text{ exists} \end{cases}$$

Now, we introduce a disjoint partition of the set X induced by the function q. Let I denote the set of values of the function q and let $Q(i)$ be a subset of the set X defined in the following way:

$$Q(i) = \begin{cases} \emptyset & \text{for } i = 0 \\ \{x \in X : q(x) = i\} & \text{for } i \in I \end{cases}$$

It is easy to verify that the family of sets $\{Q(i)\}_{i \in I}$ is a disjoint partition of the set X of the network S, that is, for every $i,j \in I$ if $i \neq j$ then

$$Q(i) \cap Q(j) = \emptyset \quad \text{and} \quad \bigcup_{i \in I} Q(i) = X$$

Moreover, for every $i \in I$ we have $1 \leqslant \|Q(i)\| \leqslant 2$.

The disjoint partition $\{Q(i)\}_{i \in I}$ determines an assignment of the tasks to the processors. It will be interpreted in the next section.

Finally, it is noteworthy that the total execution time $T(S)$ of the computation (schedule length) which is represented by the network S is given by the following formula

$$T(S) = \sum_{i=1}^{p} f_i$$

and can be determined before determining the disjoint partition $\{Q(i)\}_{i \in I}$.

5. Optimality and interpretation

In this section we shall prove some fundamental facts concerning the scheduling algorithm and its optimality. On the ground of these facts we shall interpret basic properties of the algorithm considered.

Theorem 1.

Let R_i for $1 \leqslant i \leqslant k$ be a graph defined in the following way:

$$(*)\ R_i = \langle \bigcup_{j=i}^{k} Q(j),\ \Gamma / \bigcup_{j=i}^{k} Q(j),\ x_0 \rangle$$

If $k = \max_{x \in X} \{q(x)\}$, then the sequence

$$R_q(S) = R_1, \ldots, R_i, \ldots, R_k$$

is the direct reduction of the network $S = \langle X, \Gamma, x_0 \rangle$.

Proof. By definitions of the function q and the sets $Q(i)$ we obtain that $R_1 = S$ and $R_k < Q(k), \Gamma / Q(k), x_0 >$ and thus in order to prove the theorem it is necessary to show that for every $i < k$ the graph R_{i+1} is the direct reduct of the network R_i. We proceed to prove the theorem by induction on i.

1. Basic step. For $i = 1$ in virtue of $(*)$ we obtain that $X_1 - X_2 = Q(1)$. From definition of the function q it follows that $Q(1) \subset F(1)$ which by definition of the set $F(1)$ yields $\Gamma(Q(1)) \subset X_2$. It means that the graph R_2 is the reduct of the network R_1. On the other hand, by definition of the sets $Q(i)$ we obtain that $1 \leqslant \|Q(1)\| \leqslant 2$, which means that the reduct R_2 is the direct reduct of the network R_1.

2. Induction step. Let us assume that the induction hypothesis holds for $i = n - 1 < k$, that is, that the network

$$R_n = \langle \bigcup_{j=n}^{k} Q(j),\ \Gamma / \bigcup_{j=n}^{k} Q(j),\ x_0 \rangle$$

is the direct reduct of the network R_{n-1}.

To prove the theorem, it is sufficient to show that the graph R_{n+1} is the reduct of the network R_n.

Since $1 \leqslant \|Q(n)\| \leqslant 2$ (see definition of the sets $Q(i)$), and thus in order to prove the theorem it suffices to show that the following relation

$$(1)\quad \Gamma(Q(n)) \subset \bigcup_{j=n+1}^{k} Q(j)$$

holds.

We shall show now, by leading to a contradiction that the relation (1) holds. Suppose the contrary: the relation (1) does not hold, that is

$$\Gamma(Q(n)) \not\subset \bigcup_{j=n+1}^{k} Q(j)$$

which by definition of the sets $Q(i)$ yields

$$(2)\quad \Gamma(Q(n)) \cap \bigcup_{j=1}^{n} Q(j) \neq \emptyset$$

On the other hand, in virtue of initial sequencing algorithm and by definition of the sets $Q(i)$ we obtain that

$$(3)\quad \Gamma(Q(n)) \cap \bigcup_{j=1}^{n} Q(j) \neq \emptyset$$

In virtue of (2) and (3) we get a contradiction, and thus the relation (1) holds.

From (3) it follows that

$$(4) \quad \Gamma(Q(k)) \cap \bigcup_{j=1}^{k} Q(j) = \emptyset$$

On the other hand, by definition of the sets Q(i) we have

$$(5) \quad Q(k) = X - \bigcup_{j=1}^{k-1} Q(j)$$

From (4) and (5) it follows that for R_k there exists no direct reduct. Thus the theorem has been proved.

In further considerations, the direct reduction of the network S induced by the function q , that means the sequence $R_q(S) = R_1, \ldots, R_i, \ldots, R_k$ will be called the q-reduction of the network S . From theorem 1 it follows that if $x,y \in Q(i)$ then neither $x \in \Gamma(y)$ nor $y \in \Gamma(x)$ and, moreover, if $x \in Q(n)$, $y \in Q(m)$ and $n<m$ then there exists no path leading from the vertex y to the vertex x .

In this connection, the partition $\{Q(i)\}_{i \in I}$ of the set X of the network S can be interpreted as follows : let T_x and T_y be tasks attached to the vertices x and y , respectively. If $x,y \in Q(i)$ then the tasks T_x and T_y are concurrently executed. Moreover, if $x \in Q(n)$ and $y \in Q(m)$ and $n<m$ then the task T_y is executed after the execution of the task T_x .

Let us introduce a partial ordering relation $\prec$ on the set X as follows :

$$x \prec y \Leftrightarrow q(x) < q(y)$$

It is easy to see that if $x \in Q(n)$ and $y \in Q(m)$ and $n<m$ then $x \prec y$.

We shall now show that the sequencing of the tasks of the computation based on the partial ordering relation $\prec$ is optimal, that is, the total execution time of the computation (schedule length) is minimized.

Theorem 2 .

Let $d(R(S))$ and $d(R_q(S))$ be the length of the direct reduction of the network S and the length of the q-reduction of the network S , respectively. Then for the network S there is no such direct reduction R(S) that $d(R(S)) < d(R_q(S))$.

Proof. Let a sequence

(1) $R_q(S) = R_1, \ldots, R_n, \ldots, R_k$

be the direct q-reduction of the network S . Let us consider a direct reduct R_{n+1} of the network S for which

(2) $Q(n) \subsetneq L(i)$ and $\|Q(n)\| = 1$

The condition(2) is satisfied if the following condition

(3) $r(i) \neq 0$, $H(i) = \emptyset$, $g(i) = 1$

holds. From (3) it follows that

$$(4)\quad x \in L^{*}(i,0) \Rightarrow \left(\bigcup_{n=1}^{p-i} \left(L^{*}(i,n) - \bigcup_{j=1}^{n} \Gamma^{-1}(x) \right) \right) = \emptyset$$

Let us assume that the reduct R_{n+1} is the first (in the sense of the numeration of the reducts in q-reduction) from the reducts for which the condition (2) holds. Let us assume that for i-1 the following condition

$$(5)\quad f(i-1) > 1 \qquad r(i-1) \neq 0 \; , \; H(i-1) \neq \emptyset$$

holds. From (5) it follows that there exists such vertex $x_i \in F(i-1)$ that

$$(6)\quad F(i-1) \cap L^{*}(i-1,1) = \{ x_i \}$$

Which by definition of the set F (i-1) means that $x_i = s^{*}_{i-1}$.

We distinguish the following cases :

$$(7)\quad L^{*}(i,1) - \Gamma(x_i) = \emptyset$$

$$(8)\quad L^{*}(i,1) - \Gamma(x_i) \neq \emptyset$$

We shall show now, by leading to a contradiction that case (8) cannot hold. Suppose the contrary: the case (8) does hold.

From conditions (5) and (3) it follows that $\| L^{*}(i-1,1) \| = 2n$ $(n = 1,2,3,\ldots)$ and $L^{*}(i-1,1) = L^{*}(i,0) \cup \{x_i\}$. Since $x_i = s^{*}_{i-1}$, $\| L^{*}(i-1,1) \| = 2n$ and $L^{*}(i-1,1) = L^{*}(i,0) \cup \{x_i\}$ then in virtue of (5) and (4) we obtain finally that

$$(9)\quad L^{*}(i,1) - \Gamma(x_i) = \emptyset$$

In virtue (8) and (9) we get a contradiction, and thus in order to prove the theorem it suffices only to consider the case (7).

The question arises whether the singular vertex cannot be chosen by means of other rules in such a way as to obtain a reduct R_{n+1} for which $\| Q(n) \| = 2$ and for every $j<n$ there exists no such a reduct R_{j+1} in the sequence (1) that $\| Q(j) \| = 1$. We shall now show that the answer for the question given above is negative.

Let x_s denote an arbitrary but fixed vertex from the set $L^{*}(i,0)$. Let $F'(i-1)$ and $L'(i,0)$ be sets defined as follows:

$$(10)\quad \begin{aligned} F'(i-1) &= F(i-1) \cup \{x_s\} - \{x_i\} \\ L'(i,0) &= L^{*}(i,0) \cup \{x_i\} - \{x_s\} \end{aligned}$$

If the condition (7) is valid then by (4) we obtain that for every $x \in L'(i,0)$ we have $L^{*}(i,1) - \Gamma(x) = \emptyset$, which means that we cannot create any direct reduct containing simultaneously the vertices from the sets $L'(i,0)$ and $L^{*}(i,0)$, and thus we cannot create such a direct reduct R_{n+1} for which $\| Q(n) \| = 2$. This fact means that if we apply other rule for choosing singular vertices in such a way as to obtain the sets $F'(i-1)$ and $L'(i,0)$ determined according to (10), then in the sequence (1) appears before the reduct R_{n+1} such a reduct R_{j+1} (for $j<n$) that $\| Q(j) \| = 1$.

Repeating the reasoning given above to the succeeding reducts from the sequence (1) we come to the conclusion that there exists no such a direct reduction of the network S that its length is smaller than the length of the q-reduction of the network S .

The length of the reduction of the network S can be interpreted as the number of assignments of processors to the tasks of the computation which is represented by the network S .

From theorem 2 it follows that the q-reduction of the network S determines the smallest possible number of assignments of processors needed to perform the computation represented by the network S . In other words, the disjoint partition $\{Q(i)\}_{i \in I}$ of the set X of the network S determines such a sequencing of the tasks of the computation that the total execution time is minimized.

Theorem 3 .

If T is a function defined as

$$T(S) = \sum_{i=1}^{p} f(i)$$

then $d(R_q(S)) = T(S)$

Proof. By definition of the length of q-reduction we have

(1) $$d(R_q(S)) = \max_{x \in X} \{q(x)\}$$

On the other hand, from(1) by virtue of definition of the function q it follows that

$$d(R_q(S)) = \sum_{j=1}^{p} f(i)$$

Which means, by definition of the function T that

$$d(R_q(S)) = T(S)$$

What has been to be proved.

From theorem given above it follows that q-reduction of the network S is not needed to determine the dotal execution time of the computation which is represented by the network S , which means that the total execution time of the computation can be determined before completing the sequencing of the task of the computation being under consideration.

Let us summarize the results obtained in this section :

1. The partial ordering relation $\prec$ determines the optimal sequencing of the tasks of the computation, that is, such a sequencing that the total execution time is minimized.

2. For a given i , the tasks attached to the elementes of the set Q (i) can be concurrently executed.

3. For a given network S, the value of the function T determines the total execution time for optimal sequencing of the taskts of the computation which is represented by the network S .

6. An extension of the algorithm

In this section, we shall give an extension of the scheduling algoritthm for preemptive case by assumption that all tasks of the computation have the mutually commensurable execution times. We assume that is permited to interrupt any processor at stated times. It means that we admit to split any task in a limited way. Next, we shall consider an extension of the scheduling algoritm for preemptive case, provided that all tasks of the computation have the same execution times.

Let T_x denotes a task attached to the vertex x and t_x denotes execution time of the task T_x. Let t be the largest real number such that each task execution time can be expressed as an integral multiple of t.

Let $S = \langle X, \Gamma, x_o \rangle$ be a network with the mutually commensurable executution time of the tasks. We now associate with the network S a network $S^* = \langle X^*, \Gamma^*, x_o^* \rangle$ in the following way :

i) if $x \in X$ and $t_x = n \cdot t$ then $\{x_1, \ldots, x_n\} \subset X^*$ and for $i < n$ $\Gamma^*(x_i) = \Gamma x_{i+1}$

ii) if $x, y \in X$ and $y \in \Gamma(x)$ then $y_1 \in \Gamma^*(x_n)$

iii) if $x_i \in X^*$ then $t_{x_i} = t$

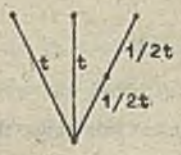
iiii the network S^* contains no other vertices as obtained by the above rules

It should be observed that $x_o^* = x_{on}$.

Let us assume that we have a system in which preemptions are allowed only at stated times $t, 2t, 3t, \ldots$. It is easy to verify that an optimal preemptive schedule for the network S^*, given by algorithm presented in this paper is an optimal preemptive schedule for the network S, provided that the case where $f(i) > 1$, $r(i) \neq 0$, $H(i) = \emptyset$ and $g(i) = 1$ does not hold see definition of the set $F(i)$. If the condition mentioned above holds, it means that the cardinality of the set $F(i)$ is an odd number and is greater than two and moreover $F(i) \subset L(j)$.

From the consideration given above it follows that the extension of the nonpreemptive algorithm for preemptive case by decomposing all tasks into chains of the tasks with the equal execution times not always yields optimal schedules. This remarks is of a general nature and concerns all preemptive algorithms obtained from nonpreemptive algorithms by assuption that all tasks of a computation have the mutually commensurable execution times. It is easy to verify that for every algorithm we can choose such a computation that on some stage of the computation we obtain the folowing network fragment

The execution time of this fragment is 2t. On the other hand, if we admit a split of one of the tasks into two equal parts as is pictured below



Then this fragment can be executed in 1 1/2 t. In this case, in order to obtain an optimal algorithm it is necessary to introduce additional rules for sequencing the tasks. For the algorithm considered in this paper it is relatively easy to introduce new rules, since this case appears in explicit way in the description of the algorithm.

Now we give an extension of the algorithm for the case where

$$f(i)>1\ ,\quad r(i)\neq 0\ ,\ H(i)=\emptyset \text{ and } g(i)=1$$

Obviously, this considerations are valid for extensions of nonpreemptive algorithm both for preemptive case by assumption that all tasks of the computation have the same execution times and for preemptive case where all tasks of the computation have the mutually commensurable execution times.

Let us assume that action of the nonpreemptive algorithm for the fixed network S^* is completed, that is, we are given a fixed sequence $F(1),\ldots,F(i)\ldots,F(p)$ of the sets for the network S^*. Let x_i^* be an arbitrary fixed vertex belonging to the set $F(i)$. With the vertex x_i^* we associate two vertices o_i and o_i^* such that $t_{o_i}=t_{o_i^*}=1/2\,t_{x_i^*}$. By this assumption we define a set $F^*(i)$ as follows:

$$F^*(i)=\begin{cases} F(i)\cup\{o_i,\ o_i^*\}-\{x_i^*\} & \text{if } f(i)>1\ ,\ r(i)\neq 0\ ,\ H(i)=\emptyset \text{ and } g(i)=1\\ F(i) & \text{otherwise}\end{cases}$$

Let the sequence $P^*(F^*(i))=P^*(1,i),\ldots,P^*(n,i),\ldots,P^*(k,i)$ be a fixed disjoint partition of the set $F^*(i)$ defined as folows:

1^0. if $F^*(i)=F(i)$ then $P^*(F^*(i))=P(F(i))$

2^0. if $F^*(i)\neq F(i)$ then

i) for $j\leq k$, $\|P^*(j,i)\|=2$

ii) $o_i\in P^*(1,i)$ and $o_i^*\in P^*(2,i)$

We are now in a position to redefine the funotion q and the disjoint partition $\{Q(i)\}_{i\in I}$ as follows:

$$q^*(x)=\begin{cases} \sum_{j=1}^{i} f(j)-f(i)+n & \text{for } x\in P^*(n,i) \text{ if } P^*(F^*(i)) \text{ exists}\\ \sum_{j=1}^{i} f(j) & \text{for } x\in F^*(i) \text{ if no such } P^*(F^*(i)) \text{ exists}\end{cases}$$

$$Q^*(i)=\left\{x\in\bigcup_{j=0}^{p} P^*(j) : q^*(i)=i\right\}$$

The vertices o_i and o_i^* can be interpreted as follows: Let $T_{x_i^*}$ be a task attached to the vertex x_i^*. We split the task $T_{x_i^*}$ into two subtasks T_1 and T_2 in such a way that they have the same execution times equal to $1/2\ t_{x_i^*}$. If execution of the task T_1 is completed then the task T_2 can be executed and moreover, the subtasks T_1 and T_2 cannot be executed by the same processor. The subtasks T_1 and T_2 are attached to the vertices o_i and x_i^* respectively.

If we assume such an interpretation of the vertices o_i and o_i^* as above, then the disjoint partition $\{Q^*(i)\}_{i \in I}$ determines optimal preemptive schedule for the computation represented by the network S. Moreover, the total execution time of the computation (schedule length) for this case is given by the following formula:

$$T^*(S) = \sum_{i=1}^{p} (f(i) - o/2) \cdot t$$

where $f(i)$ is the value of the function f for the network S^* and o is the number of occurrences the cases where $F^*(i) \neq F(i)$ for the network S^*.

The resolution of a problem of preemptive scheduling by reducing to a problem of nonpreemptive scheduling by assumption that all tasks hve the mutually commensurable execution times, aparat from the fact that not always this reduction is possible, cannot be find satisfactory. Obviously, the reduction based on decomposing all tasks into chains of tasks with the equal execution times yields problems which can be of size exponential in the size of the original problems. And thus, in some cases, scheduling algorithm can take exponential time complexity and the number of preemptions can be exponential. Moreover, some algorithms can fail to be optimal.

In the second part of this paper, we shall consider an optimal algorithm for preemptive scheduling of two-processor systems for computations consisting of dependent tasks with arbitrary execution times. And thus we have no constraints on execution time. Moreover, we assume that preemption times are completely unconstrained. The algorithm in question is an extension of the nonpreemptive algorithm considered in this paper. On the contrary to the extensions considered above, the extension of the algorithm in question is obtained by introducing and changing some rules for sequencing the tasks, but not by reducing a problem of preemptive scheduling to a problem of nonpreemptive scheduling.

References

[1] Coffman, E.G., Graham, R.L., Optimal scheduling for two-processor systems, Acta Informatica, vol. 1, no. 3 1972 , 200-213..

[2] Fujii, M., Kasami, T. and Ninomiya, K., Optimal sequencing of two equivalent processors, SIAM J. on Applied Mathematics, vol. 17, no. 4 1969 , 784-789.

[3] Fujii, M., Kasami, T. and Ninomiya, K., Erratum: Optimal sequencing of two equivalent processors, SIAM J. on Applied Mathematics, vol. 20, no. 4 1971 , 141.

[4] Coffman, E.G., Jr., Computer and job-shop scheduling theory, John Wiley and Sons, New York, London, Sydney, Toronto, 1976.

[5] Aho, A.V., Garey, M.R. and Ullman, J.D., The transitive reduction of a directed graph, SIAM J. on Computing, vol. 1, no. 2 1972 , 131-137.

[6] Muntz, R.R., Coffman, E.G., Jr., Optimal preemptive scheduling on two-processor systems, IEEE Transactions on Computers , vol. C-18, no. 11 1969 , 1014-1020.

[7] Rowicki, A., Realizacja maszynowa algorytmu planowania obliczeń dla systemów dwuprocesorowych, Prace Naukowo-Badawcze IMM, Warszawa 1980,83-99.

[8] Rowicki, A., Algorytm planowania obliczeń dwuprocesorowych, Prace IMM, vol. 17, no.2 975 , 65-82.

[9] Rowicki, A., A note on optimal scheduling for two-processor system, Information Processing Letters, vol. 4, no. 2 1975 , 27-30.

[10] Rowicki, A., On optimal scheduling for two-processor systems, Bull. Acad. Polon. Sci., Ser., Sci. Math. Astronom. Phys., vol., 24, no. 4 1976 , 287-293.

[11] Rowicki, A, Planowanie obliczeń zależnych podzielnych dla systemów dwuprocesorowych, Prace Naukowo-Badawcze IMM, Warszawa 1980, 100-117.

[12] Rowicki, A., A note on optimal preemptive scheduling for two-processor systems, Information Processing Letters, vol. 6, no. 1 1977 , 25-28.

[13] Rowicki, A., On optimal preemptive scheduling for two-processor systems, Bull. Acad. Polon. Sci., Ser., Sci. Math. Astronom. Phys., vol. 25, no. 7 1977 , 697-706.

[14] Rowicki, A., On optimal preemptive scheduling for multiprocessor systems, Bull. Acad. Polon. Sci., Ser., Sci. Math. Astronom. Phys., vol. 26, no. 7 1978 , 651-660.

Planowanie zadań bez wstępnego dzielenia zasobów dla systemów dwuprocesorowych

S t r e s z c z e n i e

W pracy rozważa się algorytm planowania obliczeń dla systemów dwuprocesorowych zawierających identyczne procesory. Zakłada się, że obliczenia zawierają zadanie niepodzielne o jednakowym czasie wykonania, nie zawierają pętli oraz mają jedno wyjście. Ponadto algorytm wyznacza również czas wykonania obliczenia.

W pracy pokazano, że rozważany algorytm jest optymalny tzn., że czas wykonania obliczenia jest minimalny.

Rozważono również rozszerzenie algorytmu planowania obliczeń na zadania podzielne, gdy zadania mają jednakowy czas wykonania oraz gdy zadania mają wspólną wielokrotność.

Планирование заданий без вытесняющего деления ресурсов для двухпроцессорных систем

К р а т к о е с о д е р ж а н и е

В работе рассматривается алгоритм планирования вычислений для двупроцессорных систем, содержащих одинковые процессоры.

Принято, что вычисления содержат неделимые задания с одинаковым временем выполнения, не содержат цепочек, а также имеют один выход. Кроме того алгоритмом задается также время выполнения вычисления.

В работе показано, что рассматриваемый алгоритм является оптимальным, т.е. имеет минимальное время выполнения вычисления.

Рассмотрено также расширение алгоритма планирования вычислений на делимые задания в случае, когда задания имеют одинаковые время выполнения, а также когда задания имеют общую многократность.

Informacja o cenach i warunkach prenumeraty na 1985 r.
- dla czasopism Instytutu Maszyn Matematycznych

● Cena prenumeraty rocznej

Techniki Komputerowe - Biuletyn Informacyjny	1560.-	dwum.
Przegląd Dokumentacyjny - Nauki i Techniki Komputerowe	1260.-	dwum.
Informacja Ekspresowa - Nauki i Techniki Komputerowe	2400.-	mies.
Prace naukowo-badawcze Instytutu Maszyn Matematycznych	660.-	3x w roku

● Warunki prenumeraty

1/ dla osób prawnych - instytucji i zakładów pracy:
- instytucje i zakłady pracy zlokalizowane w miastach wojewódzkich i pozostałych miastach, w których znajdują się siedziby oddziałów RSW "Prasa-Książka-Ruch" zamawiają prenumeratę w tych oddziałach;
- instytucje i zakłady pracy zlokalizowane w miejscowościach, gdzie nie ma oddziałów RSW "Prasa-Książka-Ruch" i na terenach wiejskich opłacają prenumeratę w urzędach pocztowych i u doręczycieli;

2/ dla osób fizycznych - prenumeratorów indywidualnych:
- osoby fizyczne zamieszkałe na wsi i w miejscowościach, gdzie nie ma oddziałów RSW "Prasa-Książka-Ruch" opłacają prenumeratę w urzędach pocztowych i u doręczycieli;
- osoby fizyczne zamieszkałe w miastach - siedzibach oddziałów RSW "Prasa-Książka-Ruch" opłacają prenumeratę wyłącznie w urzędach pocztowych nadawczo-oddawczych właściwych dla miejsca zamieszkania prenumeratora. Wpłaty dokonują używając "blankietu wpłaty" na rachunek bankowy miejscowego oddziału RSW "Prasa-Książka-Ruch";

3/ Prenumeratę ze zleceniem wysyłki za granicę przyjmuje RSW "Prasa-Książka-Ruch", Centrala Kolportażu Prasy i Wydawnictw, ul. Towarowa 28, 00-958 Warszawa, konto NBP XV Oddział w Warszawie nr 1153-201045-139-11. Prenumerata ze zleceniem wysyłki za granicę pocztą zwykłą jest droższa od prenumeraty krajowej o 50% dla zleceniodawców indywidualnych i o 100% dla zlecających instytucji i zakładów pracy.

● Terminy przyjmowania prenumeraty na kraj i za granicę:

- do dnia 10 listopada na I kwartał, I półrocze roku następnego oraz na cały rok następny,
- do dnia 1-każdego miesiąca poprzedzającego okres prenumeraty roku bieżącego.

Zamówienia na prenumeratę "Prac naukowo-badawczych Instytutu Maszyn Matematycznych przyjmuje Dział Sprzedaży Wysyłkowej Ośrodka Rozpowszechniania Wydawnictw Naukowych PAN, Warszawa, Pałac Kultury i Nauki, tel. tel.20-02-11 w.2516. Egzemplarze pojedyncze Prac są do nabycia w księgarni ORWN PAN, Warszawa, Pałac Kultury i Nauki, tel.20-02-11 w.2105.

Cena zł. 220.-

informacja ekspresowa

NAUKI I TECHNIKI KOMPUTEROWE

Instytut Maszyn Matematycznych zawiadamia, że od 1984 r., po dwuletniej przerwie, wznawia wydawanie miesięcznika "Informacja ekspresowa - Nauki i Techniki Komputerowe". W czasopiśmie zamieszczamy opisy bibliograficzne /wraz z krótkimi notatkami objaśniającymi/ dokumentów źródłowych, które znajdują się w bibliotece IMM - najlepiej zaopatrzonej w branży komputerowej.

Dokumentujemy ok. 600 pozycji książkowych rocznie /krajowych, i zagranicznych/ oraz 184 tytuły czasopism /około 2000 zeszytów/ w językach: polskim, angielskim, rosyjskim, niemieckim, czeskim; katalogi i in.

Informacja ekspresowa NiTK informuje o najnowszych publikacjach z zakresu branży komputerowej i dziedzin pokrewnych oraz nauk związanych z branżą /monografie, słowniki, podręczniki, materiały szkoleniowe, artykuły w czasopismach, przyczynki, któtkie notatki o najnowszych zdobyczach techniki komputerowej na świecie itp./ jest więc podstawowym i niezbędnym narzędziem pracy każdego pracownika naukowego, studenta, inżyniera - praktyka, projektanta i in.

Nasi Czytelnicy mogą zamawiać mikrofilmy i kserokopie dokumentów, których opisy znajdują się w Informacji ekspresowej.