

Adam Sędziwy*

Środowisko agentowe w syntaktycznym rozpoznawaniu obrazów

1. Wstęp

Grafowe gramatyki formalne są bardzo użytecznym formalizmem opisu i modelowania wielu typów systemów [2, 16], także informacyjnych. Metody syntaktyczne stanowią od dawna ważną grupę narzędzi w rozwiązywaniu problemów rozpoznawania obrazów [6, 11, 17, 18]. O ile zastosowanie ciągowych gramatyk formalnych [10] nie stwarza problemów obliczeniowych, o tyle gramatyki grafowe są w ogólności nieefektywne z uwagi na wysoką złożoność operacji parsingu czy rozstrzygania problemu przynależności. Projektant systemu rozpoznawania obrazu bazującego na gramatykach grafowych stoi między dwoma przeciwstawnymi wymaganiami: efektywnej (wielomianowej) złożoności obliczeniowej, z jednej strony i zadowalającej mocy opisowej gramatyki, z drugiej. Akceptowalnym kompromisem pomiędzy tymi żądaniami są gramatyki o złożoności kwadratowej, $O(N^2)$ takie jak ETPL(k) czy edNLC [4, 5, 9].

W przypadku problemów o dużym rozmiarze N , również kwadratowa złożoność może być zbyt wysoka z praktycznego punktu widzenia. Dobrym rozwiązaniem wydaje się wówczas zrównoleglenie obliczeń w oparciu o system wieloagentowy (por. [7]). W takim przypadku pojawiają się jednak trudności dotyczące zapewnienia spójności rozproszonego systemu agentów tak, aby ich sumaryczna wiedza zgodna była ze scentralizowanym opisem stanu systemu, a także trudności ze stworzeniem efektywnego sposobu wymiany informacji w takim środowisku. Dodatkowo stawiamy postulat dotyczący gramatyki modelującej system (zmiany w nim zachodzące). Od proponowanego formalizmu żądamy mianowicie, aby zbiór reguł transformacyjnych (produkcji) przygotowany dla opisu scentralizowanego mógł być stosowany przez poszczególnych agentów w ich lokalnych środowiskach, oraz aby ich zastosowanie nie prowadziło do zaburzenia spójności, o której mowa wyżej.

Platformą spełniającą postawione wymagania jest środowisko wieloagentowe GRADIS [8, 12]. Pozwala ono na transformację scentralizowanego modelu grafowego do modelu rozproszonego, zarządzanego przez zbiór agentów realizujących lokalnie wszelkie

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie

transformacje grafowe, w sposób zachowujący spójność opisu systemu. Platforma ta może być stosowana zarówno dla gramatyk w podejściu algorytmicznym, jak i algebraicznym [1, 2, 3]. Jednym z pierwszych procesów, jakie mają miejsce w nowo utworzonym systemie agentowym, jest jego przygotowanie do podjęcia konkretnego zadania obliczeniowego, w szczególności optymalna dystrybucja fragmentów grafu pomiędzy agentami. Pojęcie optymalności jest tu zależne od rodzaju problemu obliczeniowego [12]; w tym miejscu, mając na uwadze zagadnienia rozpoznawania obrazów, żądamy, aby agenci dysponowali jednakowej wielkości fragmentami głównego grafu. Dodatkowe założenie dotyczy stosowanej gramatyki grafowej. Będzie nią gramatyka ETPL(k) oparta o strukturę IE-grafów. Wybór tej klasy gramatyk umotywowany jest jej przydatnością w zadaniach rozpoznawania obrazów [4, 11].

Celem niniejszego artykułu jest przedstawienie środowiska agentowego spełniającego powyższe żądania i zdolnego do samoorganizacji pozwalającej na optymalny (równomierny) podział wiedzy, poprzedzający właściwe zadanie rozpoznawania obrazu.

W dalszej części artykułu przedstawione zostaną podstawowe pojęcia dotyczące struktur grafowych, których dotyczą niniejsze rozważania, koncepcja grafów częściowych, na której opiera się system GRADIS, przedstawiona zostanie semantyka operacji lokalnej produkcji w środowisku wieloagentowym i wreszcie, metoda optymalnej dekompozycji grafu centralnego, na zbiór statystycznie równych podgrafów.

2. IE grafy

IE grafy (*Indexed Edge-unambiguous*, zob. [15]) stanowią sprawdzony język modelowania obrazów (także wizyjnych), dla którego udało się:

- zdefiniować schemat konstrukcji jednoznacznej i unikalnej grafowej reprezentacji rozważanej klasy obrazów;
- scharakteryzować taką podklasę gramatyk, dla której możliwy jest parsing deterministyczny;
- skonstruować algorytm analizy syntaktycznej charakteryzujący się niewielką złożonością obliczeniową.

Poniżej przedstawiono formalną definicję IE-grafów.

Definicja 1. Indeksowanym, krawędziowo-jednoznacznym grafem, w skrócie IE-grafem, parametryzowanym przez S i G nazywamy piątkę $H = (V, D, \Sigma, \Gamma, \delta)$, gdzie:

- V – jest skończonym, niepustym zbiorem wierzchołków grafu, którym zostały przypisane w sposób jednoznaczny indeksy na bazie obiektów analizowanego obrazu,
- Σ – jest skończonym, niepustym zbiorem etykiet wierzchołkowych,

$\Gamma = \{\gamma_1, \dots, \gamma_n: \gamma_1 = \dots = \gamma_n\}$ – jest skończonym, niepustym zbiorem etykiet krawędziowych uporządkowanych przez relację prostego porządku,
 D – jest zbiorem krawędzi postaci (v, μ, w) gdzie $w, v \in V$
i $\mu \in \Gamma$
 $\delta: V \rightarrow \Gamma$ – jest funkcją etykietowania krawędzi. ■

Na tak zdefiniowanym zbiorze grafów oparta jest, przedstawiona w pracy, idea transformacji scentralizowanego modelu grafowego do modelu agentowego, bazującego na zbiorze tzw. grafów częściowych.

3. Grafy częściowe i system wieloagentowy

Zakładamy, iż spełnione są następujące postulaty [8] dotyczące środowiska agentowego:

- 1) Wiedza agenta składa się z dwóch, powiązanych ze sobą komponentów: (i) struktury grafowej (IE-grafu), reprezentującej fragment danego systemu, (ii) zbiór prawideł transformacyjnych (produkcji gramatyki $ETPL(k)$), modelujących zmiany i/lub interakcje zachodzące w tym systemie.
- 2) Fragmenty wiedzy będące w posiadaniu poszczególnych agentów są komplementarne, tzn. stanowią łącznie integralny opis całego systemu.
- 3) Lokalne transformacje grafów zachowują spójność globalnej wiedzy.

Reguły transformacyjne, o których mowa w punkcie 1-(ii) są dziedziczone z opisu scentralizowanego (z jakim mamy do czynienia przed rozproszeniem do systemu wieloagentowego) i są to gramatyki klasy $ETPL(k)$, z kolei strukturę grafową będącą drugim składnikiem wiedzy agenta definiuje się jako tzw. graf częściowy.

Definicja 2. Zbiór IE-grafów $G_i = (V_i, E_i, \delta_i)$ jest postacią komplementarną grafu G wtedy i tylko wtedy, gdy istnieje zbiór iniektywnych homomorfizmów $s_i: G_i \rightarrow G$, takich, że:

- 1) $\bigcup_{i=1..k} s_i(G_i) = G$,
- 2) $\forall i, j \in \{1, \dots, k\}: s_i(V_i) \cap s_j(V_j) = s_i(Border(G_i)) \cap s_j(Border(G_j))$,
- 3) $\forall w \in V_i \forall v \in V_j: \exists p = PathS(G, w, v) \Rightarrow \exists b \in Border(G_j): s_i(b) \in p$,
- 4) $\forall i \in \{1, \dots, k\}: v \in Border(G_i) \Leftrightarrow (\exists w \in G_i: w \text{ jest sąsiadem } v) \text{ lub } G_i = \{v\}$.

Gdzie $PathS(G, w, v)$ jest zbiorem wierzchołków należących do dowolnej acyklicznej ścieżki łączącej węzły w i v w grafie G , natomiast $Border(G_i)$ oznacza zbiór wszystkich tzw. węzłów brzegowych grafu G_i . Grafy G_i określać się będzie mianem grafów częściowych. ■

Powyższa definicja, niezbędna dla formalnego ujęcia problemu, jest nieużyteczna z praktycznego punktu widzenia, dlatego konieczne jest wprowadzenie algorytmów, które

w efektywny sposób pozwalałyby na tworzenie zbioru grafów częściowych i ponowną rekonstrukcję grafów częściowych do postaci scentralizowanej. Pierwsze z zadań możliwe jest przez rekurencyjne wywołanie algorytmu przedstawionego poniżej.

Algorytm 1. Załóżmy, że mamy dany graf G i pewien jego podgraf H . Tworzymy grafy częściowe H' i H'' w następujący sposób:

- 1) Początkowo $H' = H$ oraz $H'' = G - H$.
- 2) Każdy węzeł $v \in H$ taki, że istnieje węzeł $w \in G - H$, będący sąsiadem v w G , replikujemy (otrzymaną kopię v oznaczamy jako v') i:
 - a. Zatrzymujemy v w grafie H' i oznaczamy jako węzeł brzegowy.
 - b. Dołączamy do H'' węzeł v' wraz z wszystkimi krawędziami łączącymi v z $G - H$ i również oznaczamy v' jako wierzchołek brzegowy.
 - c. Dodajemy (uzupełniamy) odpowiednie referencje do węzłów v i v' , pozwalające na lokalizację wszystkich replik węzła v . W przypadku gdy v był już węzłem brzegowym, wówczas musimy uaktualnić informacje u wszystkich replik v co do pojawienia się wierzchołka v' .
- 3) Opcjonalnie, przeprowadzamy reindeksację węzłów w zbiorach $V(H')$ i $V(H'')$.

Wprowadza się następującą konwencję dotyczącą indeksacji węzłów w grafach częściowych. Wszystkie repliki danego węzła brzegowego noszą jednakową indeksację postaci $(-1, i)$, gdzie i numeruje węzeł brzegowy. Węzły niebędące brzegowymi mają indeksy postaci (k, i) , gdzie k jest numerem grafu częściowego, natomiast i stanowi unikalny w obszarze tego grafu numer wierzchołka. Rysunek 1 przedstawia przykładowy graf G (a) i jego postać komplementarną (b), indeksowaną zgodnie z przyjętą konwencją.

Algorytm odwrotny, rekonstrukcji grafu $G = (V, E, \delta)$ ze zbioru grafów częściowych $\{G_i\}$ ma następującą postać.

Algorytm 2. Rekonstrukcja grafu $G = (V, E, \delta)$. Inicjalnie: $V = \emptyset$, $E = \emptyset$.

- 1) Dla każdego węzła brzegowego $(-1, i)$, do zbioru V dodajemy tylko jedną jego replikę.
- 2) Pozostałe, zwykłe (niebrzegowe) węzły, dodajemy do V wraz z ich lokalnymi indeksacjami.
- 3) Krawędzie zbioru E są dziedziczone z lokalnych grafów, przy czym w sytuacji występowania krawędzi zduplikowanych (dana krawędź dziedziczona jest dwa lub więcej razy), dodajemy tylko pierwsze wystąpienie.
- 4) Funkcja δ , etykietująca krawędzie w grafie G , jest sumą δ_i po wszystkich grafach częściowych.

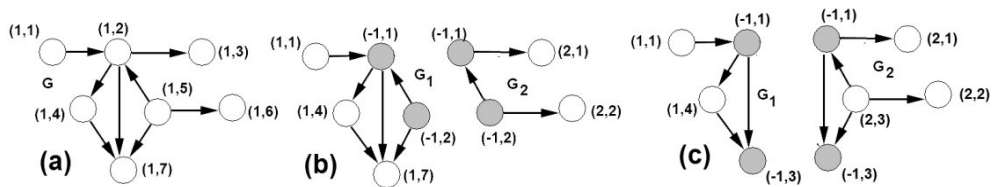
Skojarzenie agentów z komplementarną formą grafu G następuje z chwilą uruchomienia algorytmu 1. Początkowo system zawiera tylko jednego agenta, np. A1, związanego z grafem G ; następnie, po utworzeniu nowego grafu częściowego (H') A1 kreuje kolejnego agenta, A2, przypisanego do H' . A1 kontynuuje natomiast nadzór nad grafem H'' . Powstawanie dalszych agentów następuje w rezultacie rekurencyjnych wywołań algorytmu na kolejnych grafach częściowych.

Po przedstawionej wyżej transformacji opisu scentralizowanego (reprezentowanego przez G) do postaci rozproszonej (reprezentowanej przez $\{G_i\}$), z każdym z grafów G_i skojarzony jest agent odpowiedzialny za lokalne środowisko G_i , a w szczególności za lokalne wykonanie produkcji zadanej gramatyki grafowej.

Dla wykonania produkcji $P: L \rightarrow R \in \text{ETPL}(k)$ agent musi posiadać w trybie wyłączności wszystkie węzły grafu L , wraz z jego sąsiedztwem w G (aktualnie rozproszonym w zbiorze grafów częściowych). Wymóg ten wynika bezpośrednio z konieczności zapewnienia spójności grafu w trakcie transformacji osadzenia grafu R w $G-L$. Kluczowymi operacjami, niezbędnymi dla osiągnięcia tego celu są procedury $\text{INCORPORATE}(G_i, v)$ oraz $\text{NEIGHBORHOOD}(G, v, k)$.

$\text{INCORPORATE}(G_i, v)$ – dla zadanego węzła brzegowego $v \in V(G_i)$, agent zarządzający grafem G_i wykonuje, we współpracy z agentami posiadającymi repliki v (oznaczone jako v'), następujące kroki (formalny opis procedury znaleźć można w pracy [6]):

- 1) Dla każdego v' : replikuje sąsiadów v' i przyłącza te repliki do G_i wraz z krawędziami łączącymi je z v' . Sąsiedzi ci stają się węzłami brzegowymi, o ile już wcześniej nimi nie byli.
- 2) Usuwa wszystkie repliki v , zostawiając jedynie v , który staje się zwykłym węzłem.



Rys. 1. (a) Graf główny G . (b) Forma komplementarna grafu G złożona z dwóch grafów częściowych G_1 i G_2 . (c) Grafy częściowe po wykonaniu operacji $\text{INCORPORATE}(G_2, (-1,2))$ w zbiorze grafów (b)

Na rysunku 1 przedstawiono postać komplementarną danego grafu G po wywołaniu procedury $\text{INCORPORATE}(G_2, (-1,2))$.

$\text{NEIGHBORHOOD}(G, v, k)$ – dla $v \in V(G_i)$, funkcja zwraca graf B , taki, że dla dowolnego $w \in V(B)$ odległość z v do w jest nie większa niż k . W ogólnym przypadku, wykonanie tej procedury wymaga rekurencyjnej współpracy kilku agentów. Operacja ta istotna jest w procesie parsingu zachodzącym w trakcie syntaktycznego rozpoznawania obrazu.

Złożoność obliczeniowa (także w kontekście ilości komunikatów wysłanych do innych agentów) obu operacji jest złożonością wielomianową, co zostało wykazane w [14].

3.1. Wykonanie lokalnej produkcji gramatyki

Rozważamy produkcję $P: L \rightarrow R, P \in \text{ETPL}(k)$. Dla tej klasy gramatyk graf L , lewej strony produkcji, składa się z pojedynczego wierzchołka: $L = \{u\}$. Adaptując strategię opracowaną dla dowolnego grafu L , agent A_0 , musi wykonać trzy kroki (zob. [12]):

- 1) Wykryć u siebie wystąpienie węzła u ; gdy jest on wierzchołkiem brzegowym inkorporować go.
- 2) Inkorporować zbiór B zawierający wszystkich bezpośrednich sąsiadów u w grafie G : $B = \text{NEIGHBORHOOD}(G, u, 1) - \{u\}$.
- 3) Zastosować lokalnie produkcję P .

Ponieważ w ogólności B leży w obszarze kilku grafów częściowych, zbiór ten wyznaczany jest przez A_0 we współpracy z innymi agentami w następujący sposób. A_0 wysyła do odpowiednich agentów ($A_j, j = 1 \dots n$) żądanie zablokowania i dostarczenia wymaganych danych ($B_j, j = 1 \dots n$). Zbiór B jest wyznaczany jako suma zbiorów $B_j, j = 0 \dots n$.

Zgodnie z semantyką protokołu 2PC, w pierwszej fazie agent A_0 wysyła do agentów $A_j (j = 1 \dots n)$, żądanie zablokowania węzłów B_j . Agent A_j spełnia żądanie (*agreement* = *yes*) lub odrzuca je, gdy niektóre z węzłów B_j są już zablokowane (*agreement* = *noaccess*). Druga faza zależna jest od komunikatu otrzymanego przez agenta A_0 . Może ona przebiegać na dwa sposoby:

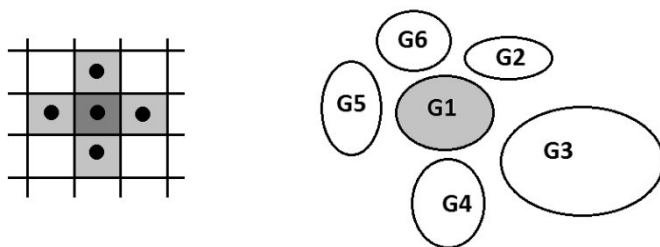
- 1) Gdy wszystkie odpowiedzi są *agreement* = *yes*, wówczas A_0 wysyła polecenie *commit* do wszystkich agentów uczestniczących w operacji, z żądaniem dostarczenia B_j .
- 2) Przynajmniej jedna odpowiedź jest *noaccess*, wówczas A_0 wysyła do wszystkich uczestniczących agentów komunikat *abort* i powtarza pierwszą fazę z pewnym losowym opóźnieniem.

W drugim kroku, po otrzymaniu wyłącznie odpowiedzi *agreement* = *yes*, A_0 inkorporuje wszystkie węzły B_j i przeprowadza lokalnie produkcję P (krok trzeci).

4. Przygotowanie środowiska agentowego

Celem, jaki stawiamy przed środowiskiem agentowym, jest jego samoorganizacja, w kierunku efektywnej realizacji zadania rozpoznawania obrazów. Nie dysponując żadną aprioryczną wiedzą na temat tego zadania, przyjmuje się, że optymalny stan systemu agentowego będzie polegał na równomiernym rozproszeniu wiedzy, a ściślej, jej komponentu grafowego (informacje odnośnie produkcji, jako dziedziczone z centralnego grafu G , są takie same dla wszystkich agentów). Żąda się, zatem, aby inicjalnie rozmiary wszystkich grafów częściowych były jednakowe i równe pewnej zadanej wartości N_{opt} . Próby uzyskania takiego środowiska były podejmowane wcześniej [13]. W podejściu tym kluczową rolę odgrywała funkcja kosztu, którą posiadał każdy z agentów, i na jej podstawie podejmował decyzję o przyłączeniu (inkorporacji) kolejnych węzłów.

W niniejszej pracy proponuje się zupełnie inne podejście, mające swoją inspirację w metodzie relaksacyjnej rozwiązywania równania Laplace'a $\Delta T = 0$, z ustalonymi warunkami brzegowymi. W metodzie tej obszar, dla którego poszukiwane jest rozwiązanie, dzieli się na siatkę, a następnie iteracyjnie oblicza się wartość rozwiązania w danym węźle, jako średnią arytmetyczną wartości w sąsiednich węzłach (rys. 2, po lewej).



Rys. 2. Metoda relaksacyjna. Schemat obliczania wartości dla równania Laplace'a (po lewej) oraz oparty o analogię sposób wyznaczania grafu częściowego, jako „średniej” z „sumy” dwóch grafów, $G1$ i wybranego sąsiada

W proponowanej metodzie relaksacyjnej zakłada się analogiczne zachowanie agentów, polegające na uśrednieniu rozmiaru. W zależności od rozmiaru swojego grafu częściowego agent A może podjąć trzy poniższe akcje:

- 1) W przypadku gdy rozmiar grafu częściowego agenta A mieści się w granicach $[N_{opt} - \epsilon, N_{opt} + \epsilon]$, wówczas nie jest podejmowana żadna akcja (ϵ jest wartością tolerancji rozmiaru grafu).
- 2) Jeśli agent A posiada graf częściowy o rozmiarze mniejszym niż $N_{opt} - \epsilon$, wówczas wybiera jednego z sąsiednich (tj. posiadającego wspólny węzeł brzegowy) agentów, np. B , scala swój graf częściowy z grafem agenta B , a następnie, o ile wynikowy graf ma rozmiar większy niż $N_{opt} + \epsilon$, rozdziela wynik scalenia na dwa podgrafy o równym rozmiarze, z których jeden zwracany jest agentowi B , natomiast drugi pozostaje pod nadzorem A . Wybór agenta B dokonywany jest pod kątem oceny rozmiaru wynikowego grafu częściowego agenta A : spośród sąsiadów wybierany jest ten, którego graf częściowy gwarantuje wynik zbliżony najbardziej do N_{opt} .
- 3) Agent A o grafie o rozmiarze powyżej $N_{opt} + \epsilon$, wykonuje jego podział na dwa równe podgrafy i do obsługi jednego z nich powołuje nowego agenta.

Należy zauważyć, że operacja scalenia (i podziału) dwóch grafów, o czym mowa powyżej, może być realizowana przez złożenie kilku operacji INCORPORATE.

4.1. Testy

Przeprowadzono następujące testy metody relaksacyjnej. Analizie poddano losowo wygenerowany graf centralny o rozmiarze $|V| = 1000$ i $|E| \approx 1414$. Przyjęto $N_{opt} = 30$, $\epsilon = 5$. Po wstępnej dekompozycji na grafy częściowe (ich liczba wyniosła ok. 700) o rozmiarze nieprzekraczającym 3 węzły, uruchomiono algorytm relaksacyjny. Kryterium stopu dla relaksacji był brak zmian średniego rozmiaru grafu częściowego. Eksperyment powtórzono 1000 razy, a otrzymane wyniki uśredniono. Badane były następujące parametry charakteryzujące zbiór grafów częściowych: rozmiar ich populacji (N_{PG}), średni rozmiar grafu

częściowego (S_{avg}) w populacji, ilość grafów mieszczących się w przyjętych granicach $[N_{opt} - \varepsilon, N_{opt} + \varepsilon]$ (N_{acc}), w relacji do rozmiaru populacji. Otrzymane wyniki przedstawiono w tabeli 1.

Tabela 1
Wyniki testów metody relaksacyjnej
(wartości parametrów są średnimi z 1000 przebiegów testu)

Parametr	Wartość
N_{PG}	41,3
S_{avg}	28,1
N_{acc}	41,2 (99,8%)

Dane w tabeli 1 pokazują, iż udało się osiągnąć podział, w którym praktycznie wszystkie grafy częściowe spełniają kryterium rozmiaru, a mianowicie zawierają 25–35 węzłów.

5. Wnioski

Przedstawiony powyżej model systemu agentowego, wyposażony w dwie podstawowe operacje, INCORPORATE i NEIGHBORHOOD, pozwala na efektywną realizację zadań rozpoznawania obrazu, opisanego za pomocą gramatyk klasy ETPL(k). Efektywność tego rozwiązania wynika zarówno z wykorzystania gramatyki o kwadratowej złożoności parsingu, jak i ze zrównoleglenia procesu przetwarzania oraz zapewnienia wielomianowej złożoności operacji wymiany komunikatów między agentami w trakcie realizacji dwóch wyżej wymienionych procedur.

Drugim czynnikiem wpływającym na oczekiwaną efektywność systemu jest podział grafu głównego na zbiór grafów częściowych o zbliżonych rozmiarach, co w przypadku zadania syntaktycznego rozpoznawania obrazu odpowiada podziałowi zadania na statystycznie równe podzadania. Bardzo wysoki poziom skuteczności algorytmu dekompozycji uzyskano dzięki oryginalnej metodzie relaksacyjnej, działającej w analogii do metody relaksacyjnej rozwiązywania równania Laplace’a.

Literatura

- [1] Corradini A., Montanari U., Rossi F., Ehrig H., Heckel R., Löwe M., *Algebraic approaches to graph transformation – part I: Basic concepts and double pushout approach*. Handbook of Graph Grammars and Computing by Graph Transformations, vol. 1: Foundations, World Scientific, 1997, 163–246.
- [2] Ehrig H., Engels G., Kreowski H.-J., Rozenberg G., *Handbook of Graph Grammars and Computing By Graph Transformation: Volume II, Applications, Languages, and Tools*. World Scientific Publishing Co., River Edge, NJ, 1999.

- [3] Ehrig H., Kreowski H.-J., Montanari U., Rozenberg G., *Handbook of Graph Grammars and Computing By Graph Transformation: Volume III, Concurrency, Parallelism, and Distribution*. World Scientific Publishing Co., River Edge, NJ, 1999.
- [4] Flasiński M., *On the Parsing of Deterministic Graph Languages for Syntactic Pattern Recognition*. Pattern Recognition, vol. 26, 1993, 16–93.
- [5] Flasiński M., *Power Properties of NCL Graph Grammars with a Polynomial Membership Problem*. Theoretical Computer Science, vol. 201, 1998, 189–231.
- [6] Flasiński M., *Distorted Pattern Analysis with the Help of Node Label Controlled Graph Languages*. Pattern Recognition, vol. 23, No. 7, 1990, 765–774.
- [7] Kotulski L., *Distributed Graphs Transformed by Multiagent System*. Artificial Intelligence and Soft Computing ICAISC 2008, LNAI 5097, 1234–1242.
- [8] Kotulski L., *GRADIS – multiagent environment supporting distributed graph transformations*. LNCS, vol. 51031/2008, 2008, 644–653.
- [9] Kotulski L., Nowak A., *Formalizing Software Refactoring in the Distributed Environment by aedNLC Graph Grammar*. SET06 IFIP Conference, Warszawa; Software Engineering Techniques: Design for Quality-Springer Series in Computer Science, ISBN: 10:0-387-39387-0, 2006, 349–360.
- [10] Ogiela M.R., Tadeusiewicz R., *Artificial Intelligence structural imaging techniques in visual pattern analysis and medical data understanding*. Pattern Recognition, vol. 36, 2003, 2441–2452.
- [11] Sędziwy A., *Distorted pattern recognition and analysis with the help of IE_f graph representation*. Computer Science (rocznik AGH), 2000.
- [12] Kotulski L., Sędziwy A., *GRADIS – the multiagent environment supported by graph transformations*. Simulation Modelling Practice and Theory, Elsevier 2010 (przyjęte do druku).
- [13] Kotulski L., Sędziwy A., *Agent Framework For Decomposing a Graph Into the Equally Sized Subgraphs*. WORLDCOMP’08 Conference, Foundations of Computer Science, 245–250.
- [14] Kotulski L., Sędziwy A., *Parallel Graph Transformation Supported by Replicated Complementary Graphs*. 5th International Conference on Graph Transformation ICGT 2010.
- [15] Kotulski L., Sędziwy A., *Stochastyczne metody generacji IE-grafów*. Automatyka (półrocznik AGH), t. 12, z. 3, 2008, 853–861.
- [16] Peng W., Krueger W., Grushin A., Carlos P., Manikonda V., Santos M., *Graph-based methods for the analysis of large-scale multiagent systems*. Proc. of the 8th International Conference on Autonomous Agents and Multiagent Systems, 2009, 545–552.
- [17] Liu H.-H., Fu K.-S., *An Application of Syntactic Pattern Recognition to Seismic Discrimination*. IEEE Transactions on Geoscience and Remote Sensing, vol. GE-21, Issue 2, 1983, 125–132.
- [18] Chiang, Y.T., Fu K.S., *Parallel Parsing Algorithms and VLSI Implementations for Syntactic Pattern Recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-6 Issue: 3, 1984, 302–314.