ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996

Seria: INFORMATYKA z. 30 Nr kol. 1315

Jarosław FRANCIK
Przemysław SZMAL

# WIELOSTANOWISKOWA WIZUALIZACJA ALGORYTMÓW W SIECIACH KOMPUTEROWYCH

Streszczenie. W artykule omówiono możliwe konfiguracje wielostanowiskowych systemów wizualizacji algorytmów. W takich systemach potencjalnie może występować wiele stanowisk generujących dane i stanowisk projekcyjnych. Pokazano, jak można skonstruować system wielostanowiskowy na bazie struktury systemu jednostanowiskowego. Wskazano możliwe protokoły do komunikowania się składowych systemu. Przedstawiono sposoby synchronizowania projekcji algorytmów.

## MULTI-STATION ALGORITHM VISUALIZATION IN COMPUTER NETWORKS

Summary. In the paper possible configurations of multi-station systems for algorithm visualization are discussed. The systems may potentially comprise many data producers and many projection stands. One shows how a multistation system can be constructed on a basis of a single-station system structure. Possible communication protocols for distributed system components are cosidered. Methods of projection synchronization are presented.

## MULTI-POSTE VISUALISATION DES ALGORITHMES DANS LES RESEAUX DES ORDINATEURS

Résumé. Dans l'article on discute des configurations possibles de systèmes multi-postes de la visualisation des algorithmes. Ces systèmes peuvent poten-

tiellement comprendre plusieurs générateurs de données et plusieurs postes de projection. On montre comment un système multi-poste peut être construit à la base de la structure d'un système à poste singulier. On indique des protocoles possibles pour la communication entre les composants distribués du systeme. On présente des methôdes de la synchronisation de projections.

# 1. Wstęp

W ciągu ostatnich kilkunastu lat można zaobserwować znaczne zainteresowanie problematyką wizualizacji algorytmów i silnie z tym związanej wizualizacji programów. Wizualizacja algorytmów polega na odwzorowaniu określonych cech algorytmów za pomocą odpowiednio dobranej reprezentacji graficznej. Przedstawienie takiej reprezentacji w formie animowanego obrazu generowanego przez system wizualizacji nazywamy projekcją.

Opracowano wiele systemów programowych umożliwiających wizualizację algorytmów. Najbardziej znany spośród nich jest system *BALSA* [1, 2] oraz systemy nowszej generacji — *TANGO* [9], *XTANGO* [10] i *ANIMUS* [3, 4]. Podstawową koncepcję wizualizacji, przyjętą w systemie *BALSA*, zastosowano w systemie *SANAL* [5, 6].

W chwili, gdy dysponujemy narzędziem umożliwiającym obserwację działania pojedynczego algorytmu na pojedynczym stanowisku projekcyjnym, w sposób naturalny pojawia się problem stworzenia warunków do porównania zachowania się kilku algorytmów, pojedynczego algorytmu z różnymi zestawami danych wejściowych, czy wreszcie algorytmu realizowanego w sposób rozproszony w sieci komputerowej. Odmiennym, acz ściśle z tym związanym problemem jest możliwość realizacji projekcji rozproszonej, przedstawianej na wielu stanowiskach roboczych jednocześnie. Projekcje takie tworzyć można z myślą o odbiorcy grupowym bądź też o indywidualnym obserwatorze, mającym jednoczesny dostęp do bogactwa środków i reprezentacji graficznych niemożliwych do uzyskania na pojedynczym, choćby bardzo wyrafinowanym stanowisku.

Możliwość rozwiązania niektórych przytoczonych tu problemów oferują jednostanowiskowe systemy z możliwością quasi-równoległych prezentacji synchronicznych. Dotyczy to obserwacji kilku algorytmów lub pojedynczego algorytmu z różnymi zestawami danych wejściowych. Jednym z takich systemów jest *SANAL* [5].

W wielu przypadkach nieodzowne jest jednak rozproszenie systemu wizualizacji na wiele stanowisk połączonych w sieci komputerowej. Dotyczy to wizualizacji algorytmów wykonywanych w sposób rozproszony w środowisku sieciowym, jak również wizualizacji wykorzystujących dwa i więcej graficznych stanowisk roboczych, na których realizowana jest projekcja.

# 2. Konfiguracje wielostanowiskowej wizualizacji algorytmów

Wielostanowiskowe systemy wizualizacji algorytmów w środowisku sieci komputerowych można konfigurować różnie. Podstawą klasyfikacji przyjętej w niniejszym opracowaniu jest liczba stanowisk, na których uruchomiony jest wizualizowany program (liczba źródeł danych), oraz liczba stanowisk, na których obserwowany jest przebieg wizualizacji (liczba obserwatorów).

## 2.1. Systemy z pojedynczym źródłem danych i pojedynczym stanowiskiem projekcyjnym

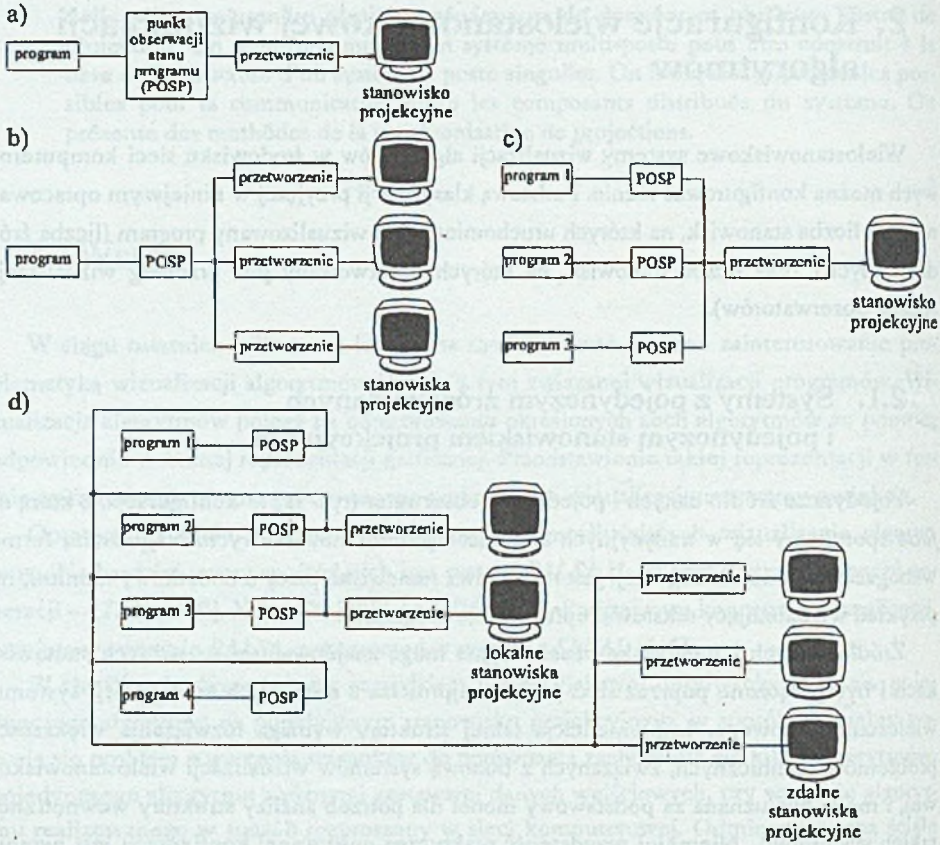
Pojedyncze źródło danych i pojedynczy obserwator (rys. 1a) to konfiguracja, z którą *de facto* spotykamy się w tradycyjnych systemach jednostanowiskowych. Najprostszą formą wzbogacenia takiej konfiguracji jest rozbudowa stanowiska pracy o dodatkowy monitor, na przykład wizualizujący tekstową reprezentację programu.

Źródło danych i stanowisko obserwacyjne mogą znajdować się na różnych stanowiskach i być połączone poprzez sieć. Jest to najprostsza z możliwych konfiguracji systemu wielostanowiskowego. Implementacja takiej struktury wymaga rozwiązania większości problemów technicznych, związanych z budową systemów wizualizacji wielostanowiskowej, i może być uznana za podstawowy model dla potrzeb analizy struktury wewnętrznej takich systemów[1]. Niemniej przydatność praktyczna opisywanej konfiguracji jest ograniczona do tych przypadków, w których przeprowadzenie wizualizacji na tym samym stanowisku, na którym realizowany jest program, jest niepożądane lub niemożliwe, jak na przykład przy wizualizacji systemów czasu rzeczywistego (napotykamy wówczas problemy związane z synchronizacją wizualizacji, por. p. 4.1). Opisywany model przydatny jest też w przypadku systemów, w których zastosowanie środków graficznych niezbędnych do wizualizacji jest trudne lub niemożliwe — jak na przykład w sterownikach przemysłowych.

## 2.2. Systemy ze zwielokrotnionymi stanowiskami projekcyjnymi

W systemach ze zwielokrotnionymi stanowiskami projekcyjnymi (rys. 1b) przebieg algorytmu wykonywanego na jednym stanowisku można obserwować na wielu stanowiskach. Każde ze stanowisk projekcyjnych może być obsługiwane przez indywidualnego ob-

[1] Analizę taką przeprowadzimy w punkcie 3 niniejszego opracowania.




Rys. 1. Konfiguracje wielostanowiskowych systemów wizualizacji algorytmów
Fig. 1. Configurations of multi-station algorithm animation systems

serwatora, który może mieć możliwość indywidualnego strojenia własnej projekcji. Możliwość taka może być ograniczona, ewentualnie zablokowana przez osobę nadzorującą projekcję. Opisywana konfiguracja systemu nadaje się przede wszystkim do przeprowadzania (być może wyreżyserowanych) prezentacji działania algorytmów zarówno w celach dydaktycznych, jak i w ramach prezentacji naukowo-technicznych lub komercyjnych.

Innym zastosowaniem opisywanej konfiguracji systemu jest prowadzenie wizualizacji wysoce skomplikowanych algorytmów na wielu stanowiskach. Zastosowanie rozwiązania rozproszonego sprowadza się tu do zwielokrotnienia dostępnej powierzchni ekranu. Na każdym stanowisku należy tak skonfigurować projekcję, by obrazowany był odmienny aspekt obserwowanego procesu. Metoda taka nadawać się może szczególnie do obserwacji skomplikowanych algorytmów współbieżnych. Zwróćmy uwagę na fakt, że obserwacja zwielokrotnionych stanowisk projekcyjnych może być dla pojedynczego obserwatora na

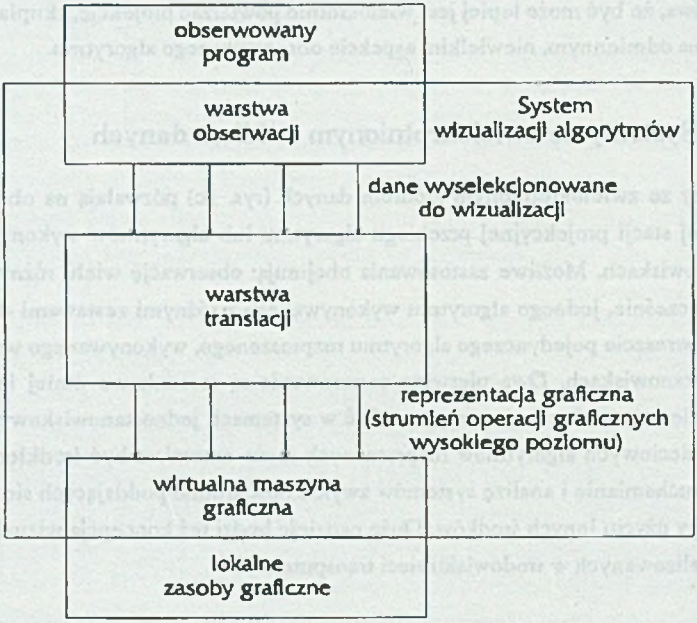
tyle uciążliwa, że być może lepiej jest wielokrotnie powtarzać projekcję, skupiając się każdorazowo na odmiennym, niewielkim aspekcie obserwowanego algorytmu.

## 2.3. Systemy ze zwielokrotnionym źródłem danych

Systemy ze zwielokrotnionym źródłem danych (rys. 1c) pozwalają na obserwację na pojedynczej stacji projekcyjnej przebiegu algorytmu lub algorytmów wykonywanych na wielu stanowiskach. Możliwe zastosowania obejmują: obserwację wielu różnych algorytmów jednocześnie, jednego algorytmu wykonywanego z różnymi zestawami danych wejściowych i wreszcie pojedynczego algorytmu rozproszonego, wykonywanego współbieżnie na wielu stanowiskach. Dwa pierwsze zastosowania są stosunkowo mniej interesujące, gdyż dają się z powodzeniem przeprowadzić w systemach jednostanowiskowych [5]. Wizualizacja sieciowych algorytmów rozproszonych może natomiast być środkiem wspomagającym uruchamianie i analizę systemów zwykle dość trudno poddających się skutecznej analizie przy użyciu innych środków. Duże nadzieje budzi też koncepcja wizualizacji algorytmów realizowanych w środowisku sieci transputerów.

## 2.4. Systemy ze zwielokrotnionym źródłem danych i ze zwielokrotnionymi stanowiskami projekcyjnymi

Model systemu ze zwielokrotnionym źródłem danych i ze zwielokrotnionymi stanowiskami projekcyjnym (rys. 1d) stanowi model uogólniony, którego przypadkami szczególnymi są struktury opisane w poprzednich punktach. Zakłada się, że na stanowiskach, na których uruchomione są wizualizowane algorytmy lub ich wątki, może się odbywać ich wizualizacja w trybie lokalnym. Jednocześnie na jednym lub wielu zdalnych stanowiskach realizować można wizualizację zdalną, odnoszącą się do procesu wykonywanego na pojedynczym stanowisku lub stanowiącą syntetyczny obraz algorytmów lub wątków realizowanych na wielu stanowiskach. W zasadzie systemy o tak uogólnionej strukturze łączą możliwości i zalety systemów o strukturach będących przypadkami szczególnymi, opisanymi poprzednio; należy się jednak spodziewać, że w większości rozwiązań praktycznych najbardziej adekwatne będzie któreś z rozwiązań szczegółowych. Systemy o uogólnionej strukturze, dające się jednak skonfigurować w sposób zgodny z koncepcją dowolnie wybraną przez użytkownika spośród opisanych wcześniej architektur szczegółowych, to systemy zapewniające maksymalną elastyczność i dogodne dla wszystkich potencjalnych grup użytkowników. Oczywiście ich realizacja może być dużo trudniejsza od implementacji pozostałych typów systemów.




Rys. 2. Warstwowa struktura systemów wizualizacji algorytmów
Fig. 2. Layer structure of algorithm visualization systems

# 3. Struktura systemów wizualizacji algorytmów z możliwością wizualizacji wielostanowiskowej

## 3.1. Warstwowy model struktury systemów wizualizacji algorytmów

Wiodącym modelem budowy systemów wizualizacji algorytmów jest model struktury warstwowej [2, 5, 11, 12] (rys. 2). Wynika on wprost ze specyfiki zadania wykonywanego przez te systemy.

Wizualizacja algorytmów polega na odwzorowaniu określonych cech algorytmów na odpowiednio dobraną reprezentację graficzną. W typowych systemach proces ten można podzielić na trzy etapy. Przyjmujemy, że kolejne etapy realizowane są przez poszczególne warstwy systemu.

W etapie pierwszym następuje ekstrakcja cech algorytmu. W typowych przypadkach oznacza to przejęcie zawartości struktur danych przetwarzanych przez wizualizowany algo-

rytm. Realizująca ten etap warstwa obserwacji dokonuje wyboru struktur danych istotnych dla konkretnej wizualizacji, wykrywa zmiany zachodzące w tych strukturach oraz zapewnia dostęp do tych danych głębiej położonym warstwom systemu. Efektem działania systemu na tym etapie jest strumień wyselekcjonowanych, lecz nie przetworzonych (surowych) danych przeznaczonych do wizualizacji.

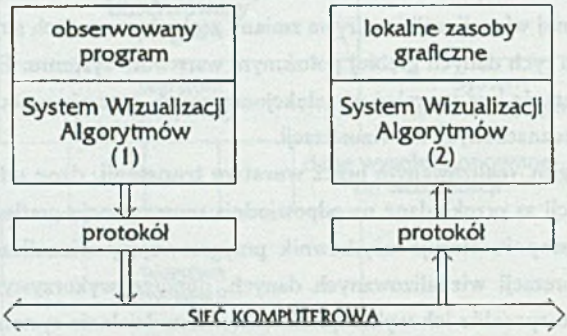
W etapie drugim, realizowanym przez warstwę translacji, dane udostępniane przez warstwę obserwacji są przekładane na odpowiednią reprezentację graficzną. Obróbką danych w dużym stopniu steruje użytkownik przygotowujący wizualizację — decyduje o sposobie interpretacji wizualizowanych danych, doborze wykorzystywanych środków graficznych oraz o sposobie ich wykorzystania. Efektem działania systemu na tym etapie jest strumień danych definiujących ciąg chwilowych reprezentacji graficznych wizualizowanego algorytmu. Najczęściej dane te to sekwencje operacji graficznych zapisanych w odpowiednio dobranym języku. Aby zapewnić przejrzystą strukturę warstwy translacji oraz przenośność systemu na tym poziomie, zastosowany język operacji graficznych powinien być językiem wysokiego poziomu, niezależnym od platformy sprzętowej i systemowej oraz zawierającym zestaw operacji zorientowanych na wykorzystanie w systemach wizualizacji.

Ostatni, trzeci etap jest realizowany przez warstwę systemu nazywaną wirtualną maszyną graficzną. Maszyna ta interpretuje strumień danych graficznych spływający z warstwy translacji i wykonuje operacje graficzne na podstawie dostępnych usług systemu operacyjnego i/lub bibliotek graficznych ogólnego przeznaczenia.

## 3.2. Możliwości podziału systemów na wiele stanowisk

Opisany powyżej warstwowy model systemu wizualizacji algorytmów dotyczy w ogólnym przypadku systemów jednostanowiskowych. W przypadku systemu dwustanowiskowego obserwowany program znajduje się na jednym stanowisku, podczas gdy wizualizację wykonuje się przy wykorzystaniu zasobów graficznych drugiego stanowiska. System wizualizacji algorytmów jest podzielony na dwa moduły, rezydujące na obu stanowiskach, przy czym komunikację pomiędzy tymi modułami zapewnia sieć komputerowa w oparciu o odpowiedni protokół komunikacyjny (rys. 3). Przypadek systemu wielostanowiskowego jest uogólnieniem poprzedniego, przy założeniu, że dokonano zwielokrotnienia jednego lub obu modułów.

Z technicznego punktu widzenia wskazane jest, by oba moduły wielostanowiskowego systemu wizualizacji algorytmów wykazywały dużą spoistość, umożliwiającą użytkownikowi zrozumienie zasady działania sytemu, co jest warunkiem pełnego wykorzystania róż-




Rys. 3. Struktura wielostanowiskowego systemu wizualizacji algorytmów
Fig. 3. Structure of a multi-station algorithm visualization system

nych konfiguracji sieciowych, oraz luźne powiązania z otoczeniem, co zapewnia prostotę zastosowanych protokołów sieciowych i minimalizuje przepływ informacji przez nośnik, a tym samym i obciążenie sieci. W rozpowszechnionej klasyfikacji (por. [7, 8]) odpowiadałoby to spoistości (mocy) funkcjonalnej oraz słabej więzi.

W tej sytuacji naturalne wydaje się, by linia podziału na moduły systemu wielostanowiskowego była poprowadzona na granicy warstw, z możliwością zlokalizowania warstwy translacji bądź w jednym, bądź w drugim module.

Inne podziały, choć teoretycznie możliwe, są zdecydowanie mniej korzystne. Podział przebiegający wewnątrz głównych wartstw struktury systemu jest niewskazany, gdyż trudno wskazać podwarstwy o mocy funkcjonalnej i dostatecznie słabej więzi. Podział przebiegający pomiędzy obserwowanym programem a warstwą obserwacji jest prawie niemożliwy do realizacji — przedmiotem działania warstwy obserwacji jest *de facto* cała przestrzeń adresowa obserwowanego programu. Wreszcie podział przebiegający pomiędzy maszyną graficzną a systemem graficznym, jakkolwiek technicznie do przyjęcia, nie wnosi nic nowego w stosunku do podziału pomiędzy warstwami translacji i graficzną. Poważnie natomiast utrudnia konfigurację prezentacji na docelowej stacji, a także wymusza zależność protokołu transmisji od platformy sprzętowej i systemowej, praktycznie uniemożliwiając wizualizację w środowisku sieci heterogenicznych.

## 3.3. Architektury wielostanowiskowych systemów wizualizacji algorytmów

Zgodnie z analizą zawartą w punkcie poprzednim, dokonamy obecnie klasyfikacji wielostanowiskowych systemów wizualizacji algorytmów na podstawie ich architektury. Określimy też przydatność poszczególnych struktur do trzech głównych zastosowań systemów wizualizacji: projektowo-badawczego, dydaktycznego oraz do celów prezentacji naukowo-technicznych i komercyjnych.

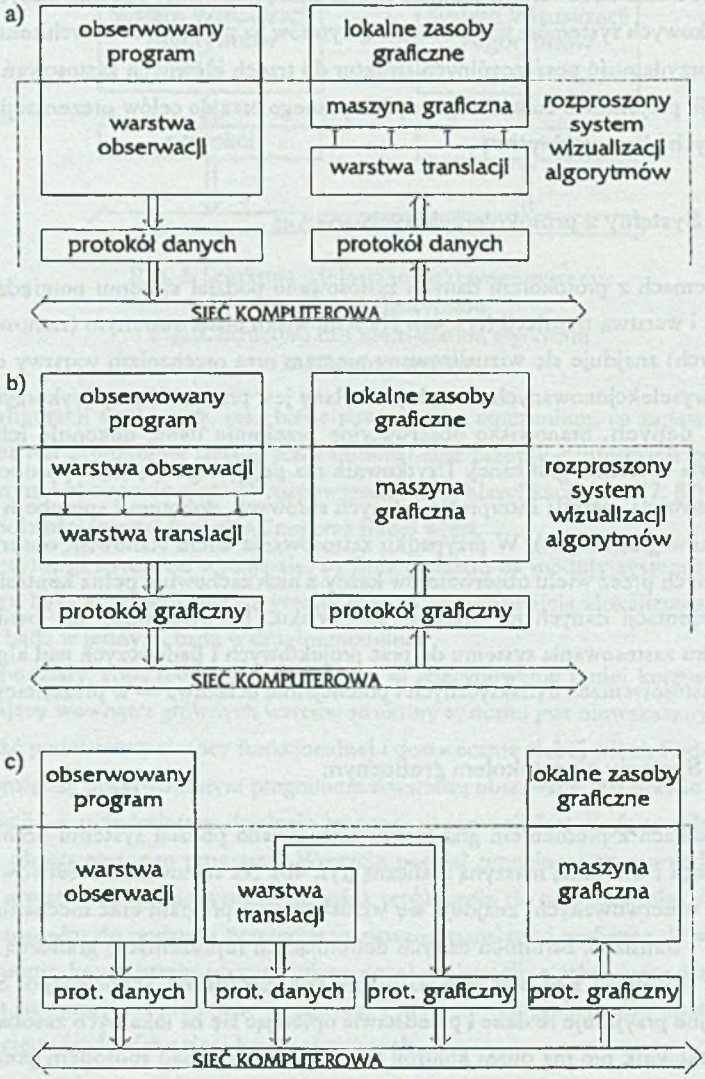
### 3.3.1. Systemy z protokołem danych

W systemach z protokołem danych zastosowano podział systemu pomiędzy warstwą obserwacji i warstwą translacji (rys. 4a). Na stanowisku obserwowanym (stanowiskach obserwowanych) znajduje się wizualizowany program oraz mechanizm warstwy obserwacji. Strumień wyselekcjonowanych danych przesyłany jest przez sieć przy wykorzystaniu tzw. protokołu danych. Stanowisko obserwacyjne przejmuje dane, dokonuje ich translacji i przedstawia w formie graficznej. Użytkownik ma pełną kontrolę nad sposobem prezentacji (zastosowaną metodą interpretacji danych surowych, doborem i sposobem wykorzystania środków graficznych). W przypadku zastosowania wielu stanowisk obserwacyjnych obsługiwanych przez wielu obserwatorów każdy z nich zachowuje pełną kontrolę nad sposobem prezentacji danych na własnym stanowisku. Ta właściwość jest bardzo istotna w przypadku zastosowania systemu do prac projektowych i badawczych nad algorytmami, cenna w zastosowaniach dydaktycznych i potencjalnie uciążliwa — w prezentacyjnych.

### 3.3.2. Systemy z protokołem graficznym

W systemach z protokołem graficznym zastosowano podział systemu pomiędzy warstwą translacji i wirtualną maszyną graficzną (rys. 4b). Na stanowisku obserwowanym (stanowiskach obserwowanych) znajduje się wizualizowany program oraz mechanizm warstw obserwacji i translacji. Strumień danych definiujących reprezentację graficzną algorytmu przesyłany jest przez sieć przy wykorzystaniu tzw. protokołu graficznego. Stanowisko obserwacyjne przyjmuje te dane i przedstawia opierając się na lokalnych zasobach graficznych. Użytkownik nie ma dużej kontroli (innej niż zdalna) nad sposobem prezentacji — w jego gestii może leżeć, co najwyżej, filtrowanie napływających danych, skalowanie obrazu itp. W przypadku grupy użytkowników obserwujących ten sam algorytm można łatwo i sprawnie zarządzać przebiegiem wizualizacji, określając sposób prezentacji (zastosowaną metodę interpretacji danych surowych, dobór i sposób wykorzystania środków graficznych)

jednocześnie dla wszystkich obserwatorów. Systemy o tej architekturze doskonale pasują do zastosowań prezentacyjnych i dydaktycznych. W zastosowaniach projektowo-badawczych są nieco mniej przydatne, choć mogą być wygodne przy pewnych typach projektów grupowych.




Rys. 4. Architektury wielostanowiskowych systemów wizualizacji algorytmów
Fig. 4. Schemes of structure of multi-station algorithm visualization systems

### 3.3.3. Systemy z protokołem mieszanym

W systemach z protokołem mieszanym zastosowano połączenie koncepcji wykorzystanych w systemach z protokołem danych oraz z protokołem graficznym (rys. 4c). Na stanowisku obserwowanym (stanowiskach obserwowanych) znajduje się wizualizowany program oraz mechanizm warstwy obserwacji. Strumień wyselekcjonowanych danych przesyłany jest przez sieć przy wykorzystaniu protokołu danych. Na innym stanowisku dane są przejmowane, dokonuje się ich translacji i ponownie przesyła, tym razem opierając się na protokole sieciowym. Wreszcie na stanowisku obserwacyjnym dane są przyjmowane i przedstawiane przy użyciu lokalnych zasobów graficznych. Powyższy opis oddaje potencjalną możliwość konstrukcji systemu: wydzielenie warstwy translacji do oddzielnego stanowiska najczęściej nie miałoby sensu. W praktycznych zastosowaniach obydwa połączenia sieciowe można zastąpić odpowiednimi połączeniami wewnątrz systemu, uzyskując uniwersalną architekturę łączącą zalety systemów z protokołem danych i z protokołem graficznym. Użytkownik może w sposób elastyczny, naprzemienny lub równoczesny, korzystać z obydwu typów protokołów.

Tabela 1

Zastosowania wielostanowiskowych systemów wizualizacji algorytmów o różnych architekturach

| zastosowanie | systemy z protokołem danych | systemy z protokołem graficznym | systemy z protokołem mieszanym |
|---|---|---|---|
| projekty i badania | ++ | + | ++ |
| dydaktyka | + | ++ | ++ |
| prezentacje | | ++ | ++ |

### 3.3.4. Systemy z rozdzieloną warstwą translacji i zdalna konfiguracja prezentacji

Możliwość budowy wielostanowiskowego systemu wizualizacji algorytmów z rozdzieloną którąkolwiek z warstw (z wykorzystaniem protokołu sieciowego przenoszącego jakąś formę wewnętrznych danych tej warstwy) w zasadzie odrzuciliśmy ze względu na trudność wyróżnienia podwarstw o mocy funkcjonalnej i dostatecznie słabej więzi (por. p. 3.2). Jednak w przypadku rozdzielenia warstwy translacji architektura taka, mimo trudności koncepcyjnych, umożliwiłaby częściowe sprecyzowanie sposobu prezentacji na stanowisku obserwowanym, ostateczne zaś — na stanowisku obserwacyjnym. Byłaby to dość interesująca alternatywa dla wcześniej opisanych koncepcji budowy systemu i znalazłaby zastosowanie do celów dydaktycznych. Kierujący wizualizacją nauczyciel narzucałby ogólne zasady prezentacji (jak w koncepcji z protokołem graficznym), podczas gdy poszczególni obserwatorzy zachowywaliby w znacznym stopniu kontrolę nad lokalnymi systemami (jak w koncepcji z protokołem danych).

Rozwiązaniem równie korzystnym z punktu widzenia użytkownika, a jednocześnie rodzącym dużo mniej problemów technicznych, byłoby jednak zastosowanie jednej z trzech podstawowych architektur systemowych i wyposażenie systemu w sprawny mechanizm zdalnego konfigurowania wizualizacji.

# 4. Synchronizacja prezentacji wielostanowiskowej

## 4.1. Synchronizacja w przypadku prezentacji jednostanowiskowej

W klasycznych, jednostanowiskowych systemach wizualizacji algorytmów o tempie prezentacji decyduje użytkownik — obserwator. Tempo to powinno być tak dobrane, by z jednej strony zapewnić możliwość dostrzeżenia krótkotrwałych i ulotnych zdarzeń zachodzących w toku realizacji algorytmu, z drugiej zaś — nie powodować nadmiernego wydłużenia czasu projekcji, nie powodując zmęczenia i zniecierpliwienia użytkownika. Najczęściej użytkownik ma też możliwość wyboru — od tempa powolnego, pozwalającego dojrzeć wszystkie szczegóły, do tempa szybkiego, czasem ograniczonego tylko efektywnością realizowanych operacji graficznych i mimo to pozwalającego dostrzec pewien ogólny, makroskopowy obraz przebiegu algorytmu. Zwykle istnieje też możliwość pracy krokowej.

Bez względu na wybór tempa pracy systemu w zdecydowanej większości przypadków wizualizacja przebiega w tempie wielokrotnie wolniejszym niż normalny, nie wizualizowany przebieg algorytmu.

Najczęściej przebieg wizualizowanego algorytmu jest spowalniany tak, by zsynchronizować tempo wykonania programu i tempo jego wizualizacji. Najprostszym środkiem synchronizacji jest zatrzymywanie wykonania programu na czas wykonywania operacji graficznych. Rozwiązaniem alternatywnym jest zapisywanie tak zwanego śladu wykonania programu w specjalnym pliku. Jest to sekwencja danych definiująca przebieg projekcji algorytmu[2]. Operacje zapisu śladu trwają o wiele krócej od operacji graficznych, choć i tu nie znika problem zsynchronizowania (opóźnienia) obserwowanego algorytmu do tempa pracy procesu zapisującego ślad. Wizualizacja algorytmu jest przy zastosowaniu tej metody

[2] Dokonując analizy podobnej do tej, jaką przeprowadzono w p. 3.2, można zauważyć, że śladem wykonania programu powinna być sekwencja danych odpowiadająca strumieniowi danych przepływającemu od warstwy obserwacji do warstwy translacji (ślad danych) lub od warstwy translacji do wirtualnej maszyny graficznej (ślad operacji graficznych). Innymi słowy, ślad zawiera te dane, które w systemach wielostanowiskowych nadają się do przesłania przez sieć.

w istocie wizualizacją śladu wykonania programu i może być z nim zupełnie rozłączna w czasie. Wadą tego rozwiązania jest niemożność pracy interaktywnej z obserwowanym programem w trakcie jego wizualizacji, zaletą zaś — wprowadzanie jedynie niewielkich opóźnień wykonania obserwowanego programu, co jest istotne w przypadku wizualizacji systemów czasu rzeczywistego (o ile otoczenie nie bazuje na treści wyświetlanych danych), programów zorientowanych na interakcję z użytkownikiem lub wizualizacji połączonej z pomiarem czasu wykonania programu lub jego części.

Osobnym problemem jest wizualizacja, która powinna zachodzić jednocześnie z wykonaniem wizualizowanego programu i zachowaniem jedynie ograniczonej możliwości opóźniania obserwowanego programu. Przykładem zastosowania jest monitoring systemów czasu rzeczywistego. Ze względu na tempo wykonania użytkownik nie ma możliwości dostrzeżenia szczegółów algorytmu, a jedynie może obserwować pewne jego aspekty makroskopowe, na przykład obciążenie systemu. Wizualizowany program nie jest wstrzymywany dla celów synchronizacji, ewentualne jego spowolnienie wiąże się jedynie z koniecznością akwizycji danych dla wizualizacji (zupełnie nie zaburzony przebieg jest możliwy tylko w przypadku dedykowanych rozwiązań sprzętowych). System wizualizacji pobiera próbki danych tak często, jak to możliwe, nie ma jednak możliwości reagowania na zachodzące zmiany. Zauważmy też, że umieszczenie systemu wizualizacji na tym samym procesorze, co obserwowanego procesu, siłą rzeczy spowoduje zmniejszenie efektywności tego procesu. Wskazana jest zatem zdalna wizualizacja poprzez sieć.

## 4.2. Synchronizacja w systemach wielostanowiskowych dla potrzeb jednego użytkownika

Wielostanowiskowe systemy wizualizacji algorytmów, w których algorytm bądź algorytmy są wykonywane na jednym lub na wielu stanowiskach i obserwowane na jednym lub na wielu stanowiskach przez jednego obserwatora, mogą korzystać z tych samych sposobów synchronizacji, co systemy jednostanowiskowe — zmianie ulec muszą jedynie środki realizacji technicznej.

W przypadku systemów dostosowujących tempo wykonywania programu do tempa wizualizacji odpowiednie mechanizmy synchronizujące powinny być wbudowane w protokół sieciowy (odpowiednio protokół danych i/lub protokół graficzny). W wielu systemach adekwatnym środkiem technicznym do zapewnienia takiej synchronizacji są semafory. Protokół powinien być sporządzony w sposób gwarantujący, że przypadkowa utrata danych podczas transmisji nie spowoduje nieprawidłowości w działaniu mechanizmów synchronizacji. Dla przykładu, odebranie ramki danych powodującej wykonanie operacji graficznej

powinno wymagać potwierdzenia. W razie zagubienia informacji transmisję powtarza się. Bez tego mechanizmu strona nadawcza mogłaby oczekiwać na zakończenie operacji graficznej, która być może nigdy nie doszła do skutku, gdyż zagubiona została informująca o niej ramka.

Systemy zachowujące ślad wykonania programu bardzo dobrze nadają się do realizacji wielostanowiskowej. Ponieważ dane składające się na ślad mogą (i najczęściej są) ścisłymi odpowiednikami danych przesyłanych w systemach poprzez sieć, w skrajnym przypadku można w ogóle zrezygnować z protokołów sieciowych. W przypadku trywialnym wystarczy wykonać wizualizację na podstawie śladu wykonania programu, przy czym oczywiście nie będzie miało najmniejszego znaczenia, czy wizualizację wykonuje się na tym samym stanowisku, na którym wykonano program, czy na innym, czy wizualizację wykonuje się na jednym, czy na wielu stanowiskach, a nawet to, czy stanowiska te są w ogóle połączone w sieć.

## 4.3. Synchronizacja w systemach wielostanowiskowych dla potrzeb wielu użytkowników

Technika dostosowania tempa wykonywania programu do tempa wizualizacji natrafia na poważne problemy w przypadku wizualizacji wielostanowiskowej przeznaczonej dla potrzeb wielu użytkowników. Wynika to z faktu, że tempo wizualizacji może być różne dla różnych użytkowników. Program będzie wówczas synchronizowany do najwolniej realizowanej wizualizacji. Taki układ jest akceptowalny co najwyżej podczas projekcji dydaktycznych lub dla celów demonstracyjnych, gdy nauczyciel lub prelegent może sam ustalić tempo demonstracji, na które nie ma już wpływu żaden z obserwatorów.

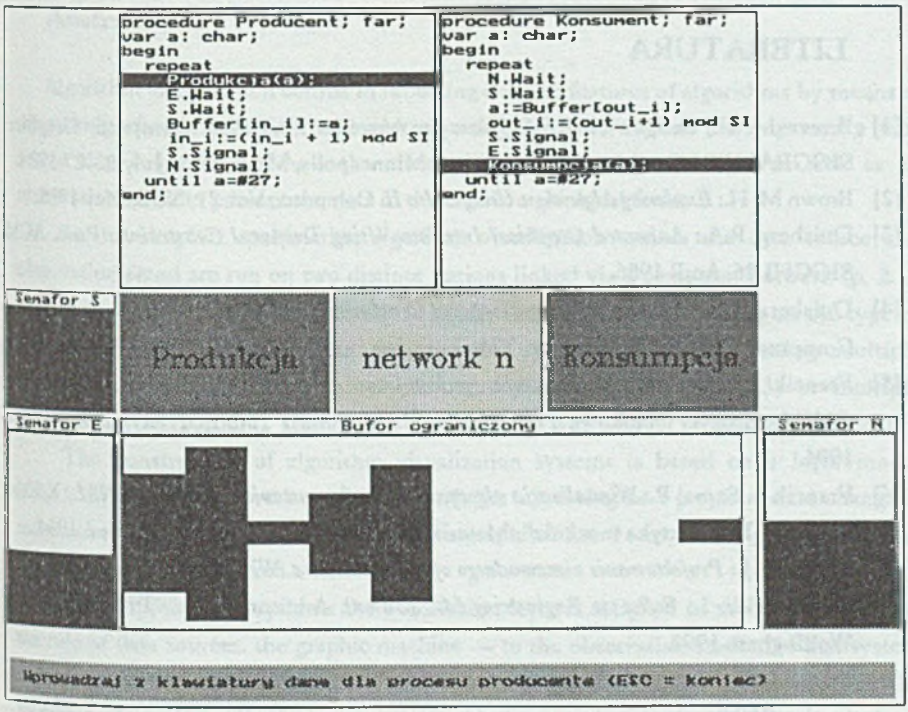
Głównym argumentem za dostosowaniem tempa wykonania programu do tempa jego wizualizacji jest możliwość interakcji pomiędzy obserwatorem a obserwowanym algorytmem w czasie trwania wizualizacji. Zauważmy, że w przypadku wizualizacji prowadzonej dla wielu użytkowników jest to argument co najmniej dyskusyjny.

Zasadniczą metodą wizualizacji algorytmów dla wielu obserwatorów jest wizualizacja z tworzeniem śladu wykonania programu.

Rozwiązaniem alternatywnym jest współbieżne wykonywanie wielu kopii obserwowanego programu, po jednej dla każdego użytkownika. Każdy użytkownik może dowolnie wpływać na tempo wykonania własnej kopii programu, a także dokonywać interakcji z programem podczas wizualizacji. Technicznie rzecz biorąc, jest to sprowadzenie problemu do zagadnienia wizualizacji przeznaczonej dla jednego użytkownika. Niekoniecznie jednak tak musi to wyglądać z punktu widzenia użytkownika, który może mieć wrażenie, że wizualizacja tego samego algorytmu jest prowadzona dla całej grupy obserwatorów.

# 5. Realizacja praktyczna

Eksperymentalną wersję wielostanowiskowego systemu wizualizacji algorytmów przygotowano wykorzystując realizowany w Zakładzie Oprogramowania Instytutu Informatyki Politechniki Śląskiej w latach 1989-1993 system animacji algorytmów *SANAL* (obecnie w wersji 3.1) [5, 6]. System ten wykorzystuje tzw. zgłoszenia zdarzeń specjalnych, wprowadzane jako wywołania podprogramów systemowych wprost do tekstu źródłowego wizualizowanego programu. Zgłoszenia te odpowiadają operacjom graficznym wysokiego poziomu. W systemie *SANAL* nie ma wyodrębnionej warstwy translacji, przez co nie ma wyboru pomiędzy architekturą systemu wielostanowiskowego opartego na protokole danych i na protokole graficznym. Zastosowany protokół zdarzeń specjalnych funkcjonalnie odpowiada protokołowi graficznemu i stanowi prosty mechanizm zdalnego wywoływania procedur obsługujących zdarzenia specjalne.




Rys. 5. Przykładowa scena z wielostanowiskowej projekcji ilustrującej problem producenta i konsumenta

Fig. 5. A sample scene from a multi-station projection that illustrates the problem of the producer and the consumer.

Rysunek 5 przedstawia scenę z projekcji ilustrującej problem producenta i konsumenta. Zarówno proces producenta, jak i konsumenta są wykonywane na osobnych, zdalnych stanowiskach, z wykorzystaniem wspólnego, cyklicznego bufora zorganizowanego w pliku z dostępem swobodnym. Plik jest umieszczony na wspólnym dysku sieciowym. Sama wizualizacja realizowana jest na trzecim stanowisku.

System *SANAL* zawiera mechanizm zapisu śladu projekcji, który mógłby być wykorzystany do realizacji wizualizacji wielostanowiskowej. Z uwagi jednak na studialny, wstępny charakter prac implementacyjnych w systemie *SANAL* koncepcji tej nie rozwinięto.

Wersja docelowa wielostanowiskowego systemu wizualizacji algorytmów zostanie zaimplementowana jako rozszerzenie systemu animacji algorytmów o roboczej nazwie „*SANAL* for Windows". Zgodnie z założeniami ma to być system z protokołem mieszanym, o elastycznej strukturze umożliwiającej tworzenie różnorakich konfiguracji sieciowych.

## LITERATURA


[1] Brown M. H., Sedgewick R.: *A System for Algorithm Animation*. Computer Graphics: SIGGRAPH'84 Conference Proceedings. Minneapolis, Minn. 18(3) July 23-27 1984.

[2] Brown M. H.: *Exploring Algorithms Using Balsa II*. Computer, Vol. 21, No 5, May 1988.

[3] Duisberg R.A.: *Animated Graphical Interfaces Using Temporal Constraints*. Proc. ACM SIGCHI 86. April 1986.

[4] Duisberg R. A.: *Animation Using Temporal Constraints, an Overview of the Animus System*. Computer Research Laboratory, Tektronix, Inc.

[5] Francik J., Szmal P.: *Wizualizacja synchroniczna algorytmów — realizacja w systemie SANAL*. Zeszyty Naukowe Politechniki Śląskiej, seria Informatyka z. 27, Gliwice 1994.

[6] Francik J., Szmal P.: *Wizualizacja algorytmów z wykorzystaniem systemu SANAL*. X Konferencja „Informatyka w szkole". Materiały konferencyjne. Toruń, wrzesień 1994.

[7] Myers G. J.: *Projektowanie niezawodnego oprogramowania*. WNT, Warszawa, 1980.

[8] Sommerville I.: *Software Engineering (4th edition)*. Addison-Wesley Publ. Company, Wokingham 1992.

[9] Stasko J. T.: Tango: *A Framework and System for Algorithm Animation*. Computer, September 1990.

[10] Stasko J.: *Animating algorithms with XTANGO*. SIGACT News, Vol: 23, Iss: 2 p. 67-71.

[11] Szmal P., Francik J., Tomaszewska A., Wójcik K.: *Wizualizacja dynamicznych struktur danych w języku Smalltalk-80*. Raport końcowy z realizowanych badań statutowych. Instytut Informatyki Politechniki Śląskiej, Gliwice 1994.

[12] Szmal P., Dobosz K., Francik J., Gonera P., Migas A.: *Metody wizualizacji sterowanej danymi w środowisku Smalltalk-80*. Raport końcowy z realizowanych badań statutowych. Instytut Informatyki Politechniki Śląskiej, Gliwice 1995.

Recenzent: Dr hab. inż. Tadeusz Czachórski

Wpłynęło do Redakcji 27 grudnia 1995 r.

## Abstract

Algorithm visualization consist in reflecting definite features of algorithms by means of adequately chosen graphical representation. Visualization system encompasses a data source that informs it of how the state of algorithm/program change, as well as an observation stand to present the data to the user in a graphical form.

In the simplest multistation algorithm visualization system the data source and observation stand are run on two distinct stations linked via a computer network (p. 2.1). More developed hardware configurations can be used (Fig. 1). Depending on the type of intended visualizations they can encompass a single data source feeding multiple observation stands (p. 2.2), or multiple sources feeding single (p. 2.3) or multiple observation stands (p. 2.4). The stations communicate with respect to adequate protocol.

The construction of algorithm visualization systems is based on a layer model [2,5,11,12] (Fig. 2). It covers an observation layer (observing how program state changes), translation layer (translating source data to a graphical form) and virtual graphic machine (interpreting graphic data to a visible form).

In all multistation systems the observation layer is assigned to the stations playing the role of data sources, the graphic machine — to the observation stations. Final system architecture and the kind of data protocol used depend on the assignment of the translation layer (Fig. 4). Data protocol, graphical protocol and mixed-mode protocol are used, respectively, when the layer is assigned to data sources (p. 3.3.1), to observation stands (p. 3.3.2) or when it is run on independent stations. The architecture affects the systems' usefulness in typical applications. Conclusions are shown in Tab. 1.

Problems of synchronization of partial processes run on individual stations are discussed in p. 4.

In p. 5 main features of multistation visualization implementation in the *SANAL* system [5,6] are presented. A scene from a sample algorithm visualization with 3 stations involved is shown in Fig. 5.