Agata SURÓWKA

Politechnika Rzeszowska, Wydział Zarządzania,
Katedra Metod Ilościowych
al. Powstańców Warszawy 8, 35-959 Rzeszów
agasur@prz.edu.pl

ALGORYTM FORDA – FULKERSONA I JEGO ZNACZENIE W ROZWIĄZYWANIU PROBLEMÓW TRANSPORTOWYCH

Streszczenie:

Artykuł zawiera autorski program Pgraf.exe, w którym zaimplementowano algorytm Forda - Fulkersona. Obliczenia wykonywane ręcznie wg tego algorytmu są czasochłonne, zatem opracowany program pozwala zaoszczędzić czas. Zapewnia również możliwość przechowywania w pamięci raz wprowadzonego grafu. Może mieć on zastosowanie w wielu dziedzinach, chociażby w logistyce.

Słowa kluczowe: metody ilościowe, algorytm Forda - Fulkersona, logistyka, programowanie w języku C.

WPROWADZENIE

Celem artykułu jest prezentacja algorytmu Forda - Fulkersona. Idea algorytmu została w nim wyjaśniona na przykładzie praktycznym dzięki sformułowaniu tzw. studium przypadku. Zwrócono uwagę na znaczenie tego algorytmu w rozwiązywaniu problemów transportowych, jak również zaprezentowano zaprojektowane autorskie oprogramowanie Algorytmu Forda - Fulkersona. Przedstawiony w artykule program Pgraf.exe napisany został z wykorzystaniem języka C++. Jeden z rozdziałów opisuje sposób pisania i etapy projektowania tego programu. Zaprezentowany własny program Pgraf.exe jest prostszy w obsłudze, bardziej wydajny oraz posiada bardziej przyjazny interfejs graficzny od istniejących na rynku rozwiązań.

1. ISTOTA ALGORYTMU FORDA FULKERSONA

Algorytm[1] Forda- Fulkersona opiera się na twierdzeniu o maksymalnym przepływie i minimalnym przekroju. Wynik ten został po raz pierwszy udowodniony przez Forda Fulkersona w 1955 roku. Twierdzenie to dowodzi, że wartość dowolnego przepływu maksymalnego jest równa przepustowości dowolnego przekroju minimalnego. Stosowanie go opiera się na poszukiwaniu takiego przekroju i przepływu, że wartość przepływu jest równa przepustowości przekroju.

[1] Algorytm w ujęciu ilościowym można najprościej określić jako pewnego rodzaju receptę, która pomoże rozwiązać pewien problem matematyczny. Składa się ze zbioru wskazówek, które mają być pomocne w rozwiązaniu danego zagadnienia. Każdy krok w takim algorytmie musi być określony w sposób precyzyjny i jednoznaczny, a proces ma zakończenie po skończonej liczbie kroków.

W literaturze dostępnych jest wiele sposobów zastosowania algorytmu Forda Fulkersona. Zaprezentowany poniżej jest jednym z wielu[2]. Przedstawiany on został w postaci następujących etapów (kroków):

Krok pierwszy

Należy znaleźć wyjściowe rozwiązanie dopuszczalne podstawiając $x_{v_\alpha, v_\beta} = 0$ dla $<v_\alpha, v_\beta \in U$[3] (gdzie x_{v_α, v_β} - maksymalne natężenie przepływu $<v_\alpha, v_\beta>$). Przepływ przez sieć jest równy 0. Utworzyć kolejkę L, w której w trakcie wykonywania kroku drugiego, przechowywane będą wierzchołki ocechowane i jeszcze nie zbadane (nie ocechowane). Zgodnie z regulaminem kolejki wierzchołki dołączane są do L od końca, a pobierane z początku. Podstawić L= Ø. Następnie nadajemy wierzchołkowi początkowemu (v_s) cechę stałą $(-,\infty)$. Znak „-" oznacza, że v_s nie jest poprzedzony przez żaden wierzchołek a „∞", że w źródle jest dostępna nieograniczona liczba jednostek towaru, tj. v_s nie ogranicza przepływu przez daną sieć.

Krok drugi - Badanie i cechowanie wierzchołków.

Na początku należy dołączyć „v_s " do L.

(a) Jeżeli L= Ø, to przechodzimy do kroku czwartego. W przeciwnym razie należy przejść do punktu (b) w celu utworzenia łańcucha powiększającego

(b) Pobrać wierzchołek z L (oznaczę go przez v_α). Badamy kolejno wierzchołki v_β osiągalne z v_α[4] aktualnie nie ocechowane. Stosujemy tu dwie reguły cechowania:

- Pierwsza – jeżeli poruszamy się zgodnie z orientacją łuku, wierzchołkowi v_β nadajemy cechę $(v_\alpha^+, d_{v_\beta})$, o ile łuk jest nienasycony (gdzie: v_α^+ - numer wierzchołka poprzedzającego węzeł (v_α) dla rozpatrywanego łańcucha, d_{v_β} - wartość przepływu do węzła v_β).
- Druga – jeżeli poruszamy się niezgodnie z orientacją łuku, wierzchołkowi v_β nadajemy cechę postaci $(v_\alpha^-, d_{v_\beta})$, o ile łuk $< v_\beta, v_\alpha >$ jest użyteczny. (gdzie: v_α^- - numer wierzchołka poprzedzającego węzeł (v_α) dla rozpatrywanego łańcucha)

  Następnie dołączamy v_β do L.

(c) jeżeli v_t (wierzchołek końcowy) należy do L, tzn. gdy udało się znaleźć łańcuch powiększający, to należy przejść do kroku trzeciego. W przeciwnym razie powrócić do punktu (a) kroku 2.

Krok trzeci - Powiększanie przepływu.

Startując od wierzchołka v_t i korzystając z pierwszych cech wierzchołków wyznaczyć łańcuch powiększający z v_s do v_t i zwiększyć przepływ po tym łańcuchu o d_{v_t}. Wierzchołki łańcucha wyznaczamy od końca. Cecha w wierzchołku v_t wskazuje na wierzchołek przedostatni w łańcuchu, z kolei jego cecha wskazuje na wierzchołek, który go poprzedza itd. (aż do osiągnięcia wierzchołka v_s). Zmiana przepływu jest wyznaczona przez znaki cech

[2] źródło: Ignasiak E., Badania operacyjne, Polskie Wydawnictwo Ekonomiczne, Warszawa 2001, s. 89 – 99.

[3] Z uwagi na ograniczenia dotyczące objętości artykułu podstawowe pojęcia używane w algorytmie nie zostały opisane. Zainteresowanych czytelników odsyłam do obszernej literatury z tego zakresu.

[4] Wierzchołek v_β jest osiągalny z v_α , jeżeli istnieje łuk $<v_\alpha, v_\beta>$ lub $< v_\beta, v_\alpha >$ tzn. jeżeli z v_α do v_β, możemy przejść zgodnie (istnieje łuk $< v_\alpha , v_\beta>$) bądź niezgodnie z orientacją łuku (istnieje łuk $< v_\beta, v_\alpha >$).

wierzchołków i ulega zwiększeniu o d_{v_t} na łukach o orientacji zgodnej oraz zmniejszeniu o d_{v_t} na łukach o orientacji przeciwnej do kierunku od v_s do v_t. Bieżący przepływ ulega zwiększeniu o d_{v_t}. Po zmianie przepływu zlikwidować wszystkie cechy wierzchołków z wyjątkiem wierzchołka v_s i powrócić do kroku drugiego omawianego algorytmu.

Krok czwarty - Koniec algorytmu.

Otrzymany przepływ jest maksymalny. Przekrój minimalny można otrzymać umieszczając w zbiorze P' wierzchołki ocechowane w kroku trzecim, a pozostałe – w zbiorze P''. Dla przekroju S określamy jego przepustowość jako sumą przepustowości łuków tworzących ten przekrój. Jeśli otrzymany przepływ równy jest przepustowości przekroju, to spełniony został warunek:

$$\Phi_{max} = S^* \quad (1)$$

gdzie: $\mathbf{\Phi_{max}}$ – maksymalny przepływ,
$\mathbf{S^*}$ – minimalny przekrój.

2. ZASTOSOWANIE ALGORYTMU FORDA FULKERSONA – OBLICZENIA WŁASNE NA PRZYKŁADZIE PRAKTYCZNYM

Algorytm Forda-Fulkersona znajduje zastosowanie w wielu dziedzinach. Jednym z przykładów spotykanych w literaturze[5] jest następujący, a dotyczący eksportu kawy z Meksyku. Kawa ta, jak podaje literatura znana jest z dobrej jakości i ma rynki zbytu na całym świecie. Towarzystwo „Corra Hermanos i spółka" posiada magazyny kawy w portach: Veracruz, Tampico, Tuxpan i Campeche. A importerzy to: Bordeaux, Saint-Nazaire, Hawar i Dunkierka. Do wyżej wymienionych portów kierowane są zamówienia od wymienionych importerów. Najważniejsze jest 100% wykorzystanie zapasów w magazynie Campeche. Wszystkie informacje potrzebne do skonstruowania sieci zostały zamieszczone w poniższych tabelach. (zob. tab. 1-3)

Tabela. 1. Zapasy przechowywane w magazynach portowych i zamówienia od poszczególnych importerów

A[6]	**Porty na grafie**	**Ilość zapasów**	**B**	**Importerzy na grafie**	**Ilość zamówionego towaru**
Veracruz	V_1	120	Dunkierka	V_5	100
Tampico	V_2	100	Bordeaux	V_6	80
Tuxpan	V_3	100	Saint-Nazaire	V_7	90
Campeche	V_4	100	Hawar	V_8	150

Źródło: Kaufman A., Faure R., Badania operacyjne na co dzień, Polskie Wydawnictwo Ekonomiczne, Warszawa 1998.

[5] Zob. Kaufmana A., Faure R., Badania operacyjne na co dzień, PWE, Warszawa 1998.
[6] W tabeli przyjęto następujące oznaczenia: A - porty, w których przechowywane są zapasy. B - importerzy od których są zamówienia.

Tabela 2. Ładowność na poszczególnych odcinkach.

→	Port docelowy	Dunkierka	Bordeaux	Saint-Nazaire	Hawar
Port wyjściowy	↓	V_5	V_6	V_7	V_8
Veracruz	V_1	70	30	20	0
Tampico	V_2	50	40	10	0
Tuxpan	V_3	0	20	40	80
Campeche	V_4	0	20	40	80

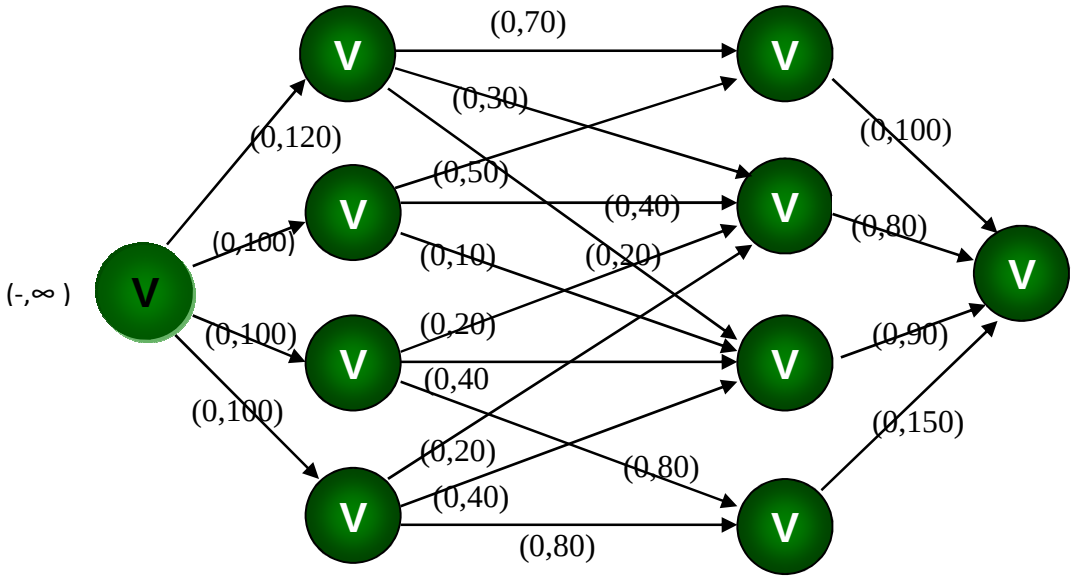
Źródło: Kaufman A., Faure R., Badania operacyjne na co dzień, Polskie Wydawnictwo Ekonomiczne, Warszawa 1998.

Tabela 3. Przepustowość poszczególnych łuków grafu.

Łuki na grafie	Wielkości ładunku	Łuki na grafie	Wielkości ładunku
$<V_0,V_1>$	120	$<V_2,V_7>$	10
$<V_0,V_2>$	100	$<V_3,V_6>$	20
$<V_0,V_3>$	100	$<V_3,V_7>$	40
$<V_0,V_4>$	100	$<V_3,V_8>$	80
$<V_1,V_5>$	70	$<V_4,V_6>$	20
$<V_1,V_6>$	30	$<V_4,V_7>$	40
$<V_1,V_7>$	20	$<V_4,V_8>$	80
$<V_2,V_5>$	50	$<V_5,V_9>$	100
$<V_2,V_6>$	40	$<V_6,V_9>$	80
$<V_7,V_9>$	90	$<V_8,V_9>$	150

Zob. Kaufman A., Faure R., Badania operacyjne na co dzień, Polskie Wydawnictwo Ekonomiczne, Warszawa 1998.

W oparciu o powyższe dane (uwzględniając przepustowość na poszczególnych łukach) skonstruowano graf, który w sposób graficzny został przedstawiony na poniższym rysunku (zob. rys.1).



Rys. 1. Operacje na grafach (proces cechowania)

Źródło: opracowanie i obliczenia własne.

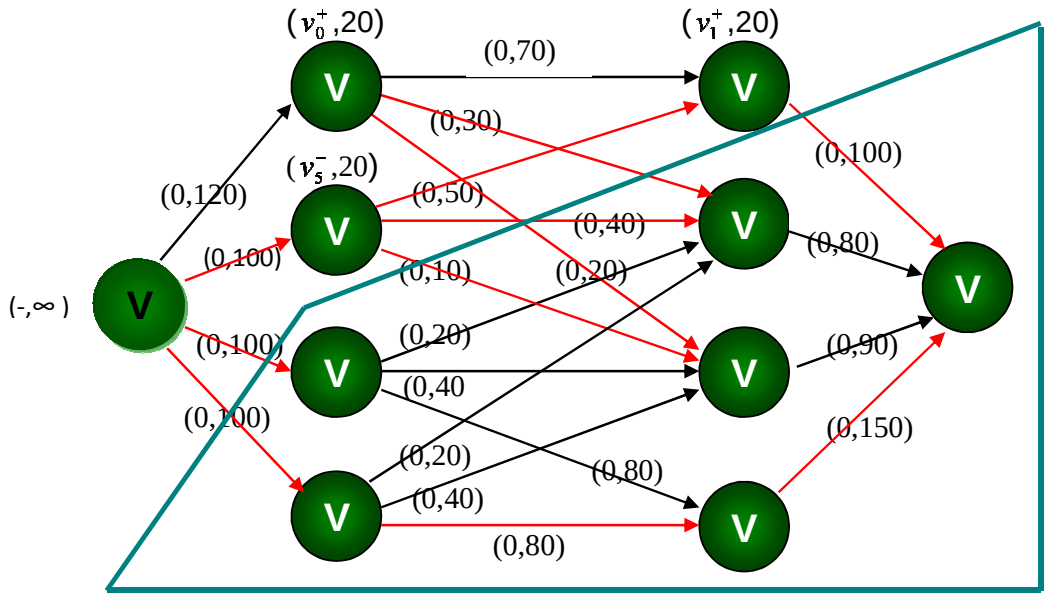
Na grafie tym wykonane zostały obliczenia z zastosowaniem algorytmu opisanego w poprzednim rozdziale. Wybrane wyniki obliczeń dotyczące poszczególnych etapów zostały zawarte w poniższej tabeli (zob. tab.4). Analiza tabeli pozwala zauważyć, iż wartość otrzymanego przepływu wynosi 400.

W sposób graficzny ostatni etap obliczeń został zaprezentowany na poniższym rysunku (zob. rys.2). Otrzymany przekrój został na nim wyznaczony przez ramkę.

Tabela 4. Łańcuchy powiększające i wartości przepływu.

Krok	Łańcuch powiększający	Wartość przepływu
1	$[v_0,v_4]$, $[v_4,v_8]$, $[v_8,v_9]$	80
2	$[v_0,v_4]$, $[v_4,v_7]$, $[v_7,v_9]$	20
3	$[v_0,v_3]$, $[v_3,v_8]$, $[v_8,v_9]$	70
4	$[v_0,v_3]$, $[v_3,v_7]$, $[v_7,v_9]$	30
5	$[v_0,v_2]$, $[v_2,v_7]$, $[v_7,v_9]$	10
6	$[v_0,v_2]$, $[v_2,v_6]$, $[v_6,v_9]$	40
7	$[v_0,v_2]$, $[v_2,v_5]$, $[v_5,v_9]$	50
8	$[v_0,v_1]$, $[v_1,v_7]$, $[v_7,v_9]$	20
9	$[v_0,v_1]$, $[v_1,v_6]$, $[v_6,v_9]$	30
10	$[v_0,v_1]$, $[v_1,v_5]$, $[v_5,v_9]$	50
11	Proces cechowania został zakończony, a węzeł końcowy nie został osiągnięty.	**400**

Źródło: opracowanie własne.



Rys. 2 Operacje na grafach (proces cechowania)

Źródło: opracowanie i obliczenia własne.

Analizując rysunek możemy zaobserwować, iż zbiór węzłów (oznaczony jako P) został podzielony na dwie części tj. P' i P''. Do zbioru P' dołączono te węzły, które zostały ocechowane w ostatnim etapie (P'=$\{v_0,v_1,v_2,v_5\}$). Natomiast do zbioru P'' włączono pozostałe węzły, czyli P''=$\{v_3,v_4,v_6,v_7,v_8,v_9\}$. Łatwo można zauważyć, że $v_0 \in$ P', $v_9 \in$ P'', a P''=P-P'. Zatem zbiory P' i P'' pozwalają na wyznaczenie przekroju, który ma postać

$$S= \{<v_0,v_4>, <v_0,v_3>, <v_2,v_7>, <v_2,v_6>, <v_1,v_7>, <v_1,v_6>, <v_5,v_9> \}.$$

Wartość przekroju została obliczona w następujący sposób:

$$\boldsymbol{S^*}= 100+100+10+40+20+30+100= \underline{\mathbf{400}}.$$

Spełniony został zatem warunek, że wartość przepływu jest identyczna jak wartość przekroju, co gwarantuje iż sposób podziału P na podzbiory P' i P'' jest przekrojem minimalnym.

3. PROGRAM PGRAF.EXE – SPOSÓB JEGO NAPISANIA I PROJEKTOWANIA

Etapy projektowania programu w języku C są tak ułożone, żeby przy zachowaniu ich kolejności wyeliminować możliwość powstawania licznych błędów wynikających najczęściej z faktu, że większość programistów rutynowo rzuca się w tok pisania programów bez wcześniejszego przygotowania do zadanego problemu. W ten sposób tracą oni niekiedy mnóstwo czasu na wprowadzenie licznych poprawek, gubią się w licznych szczegółach i poszukiwaniu niepotrzebnych rozwiązań do dodatkowych trudności. Dlatego rzeczą przydatną jest wyrobienie w sobie pewnych nawyków przy projektowaniu. Biorąc to pod uwagę projektowanie programu Pgraf.exe składało się z następujących etapów:

Etap 1 - Formułowanie założeń

1. Stworzenie algorytmu dokonującego obliczeń maksymalnego przepływu w grafie:
   - zdefiniowanie danych opisujących graf
   - napisanie funkcji przeszukującej graf
   - stworzenie algorytmu obliczającego maksymalny przepływ
2. Program powinien posiadać łatwy w obsłudze interfejs graficzny:
   - wizualne przedstawienie grafu
   - nieskomplikowany sposób wprowadzania danych opisujących graf
   - łatwy i szybki sposób otrzymania końcowego wyniku
3. Do napisania programu należy użyć języka, który pozwoli na łatwą realizację celu:
   - język powinien w sposób łatwy definiować graf
   - łatwość uzyskania algorytmów
   - nieskomplikowany sposób tworzenia graficznej reprezentacji danych

Etap 2 - Realizacja algorytmów według założeń

Założenie 1:

- Tablicowa prezentacja danych opisujących graf w pamięci komputera.
- Tablicowe przedstawienie przepływu.
- Prezentacja węzłów w postaci tablic.
- Tablica przeszukiwania grafu.
- Dane opisujące pierwszy i ostatni węzeł grafu.
- Podział programu na poszczególne bloki funkcyjne.
- Powiązanie danych z poszczególnymi blokami funkcjonalnymi.
- Wyodrębnienie poszczególnych elementarnych działań występujących w blokach funkcjonalnych.
- Zainicjalizowanie zmiennych początkowych.
- Prześledzenie działania algorytmu.
- Dokonanie ewentualnych poprawek.

Założenie 2:

- Wykorzystanie graficznych możliwości systemu operacyjnego Windows do wizualnej prezentacji grafu.

- Wykorzystanie graficznego interfejsu użytkownika do wprowadzania danych opisujących graf.
- Graficzne przedstawienie wyników działania programu.

Założenie 3:

- Wybór języka C do implementacji algorytmów.
- Wybór środowiska programistycznego C++ Builder ze względu na łatwość tworzenia grafiki, wspomaganie budowy interfejsu graficznego użytkownika.
- Wspomaganie uruchamiania i usuwania błędów w programie przez środowisko Buildera.

Etap 3 - Pisanie programu Pgraf.exe

Uruchomienie C++ Buildera, stworzenie interfejsu użytkownika poprzez rozmieszczenie następujących komponentów na formularzu programu:

1. TStringGrid –służy do wprowadzania danych przez użytkownika.
2. TSaveDialog – zapisuje dane do pliku na dysku.
3. TOpenDialog – wczytuje dane z pliku na dysku.
4. Button – dwa klawisze obsługujące zapis i odczyt danych do i z pliku. Klawisz uruchamiający wykonanie obliczeń przez program i kończący program po wykonaniu obliczeń. Przycisk kasujący dane i przygotowujący program do ponownych obliczeń, oraz klawisz wyświetlający informacje o autorze programu.
5. Label – służy do wyprowadzania napisów na ekran.

Etap 4 - Uruchamianie programu

Etap 5 - Testowanie prawidłowości wykonania obliczeń.

Opisane powyżej zagadnienia są tylko wybranymi z uwagi na ograniczone ramy opracowania

4. PROGRAM KOMPUTEROWY OBSŁUGUJĄCY ALGORYTM FORDA-FULKERSONA

Po uruchomieniu programu pojawia się okno główne (zob. rys. 3) na tle którego ukazuje się okno wyświetlające informacje o programie. W górnej części okna głównego znajdują się węzły opisujące graf, poniżej natomiast tabela umożliwiająca wprowadzanie danych określających wartości przepływu między węzłami. Wartość określającą krawędź wpisuje się do wiersza określającego węzeł, z którego krawędź wychodzi i kolumny, która określa węzeł docelowy. Przepływ jest opisywany przez liczby całkowite.

Na dolnym pasku okna głównego znajduje się pięć klawiszy:

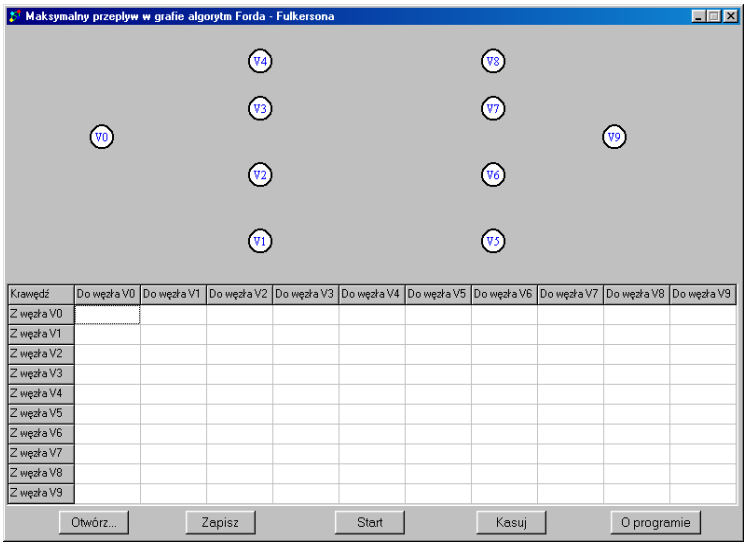
Otwórz – umożliwia wybranie pliku zawierającego dane opisujące krawędzie grafu.

Zapisz – umożliwia podanie nazwy pliku, w którym zostaną zapisane dane dotyczące krawędzi grafu.

Start – uruchamia algorytm obliczający maksymalny przepływ, a po otrzymaniu wyniku końcowego napis zmienia się na „koniec”.

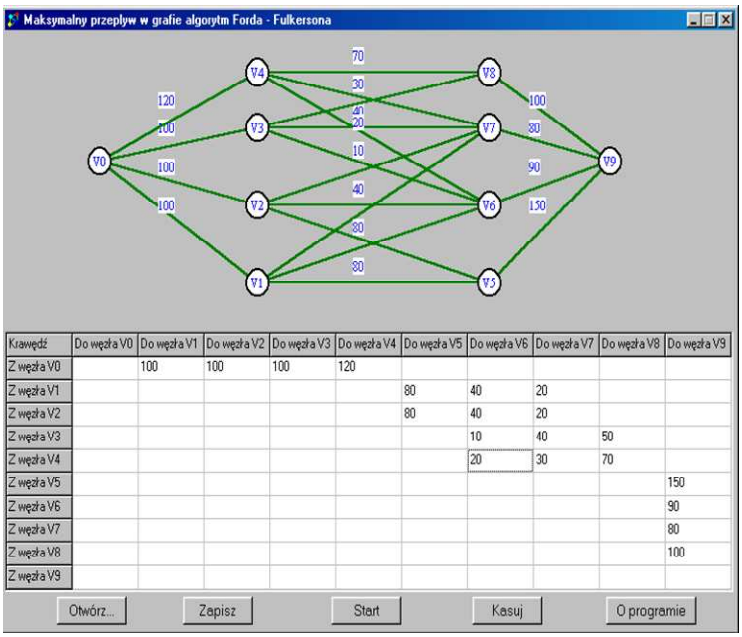
Kasuj – kasuje dane dotyczące obliczeń na grafie i jego graficzny obraz, przygotowuje program do ponownych obliczeń.

O programie – wyświetla okno zawierające informacje o programie.



Rys. 3. Wygląd okna głównego programu.

Źródło: Program Pgraf.exe.

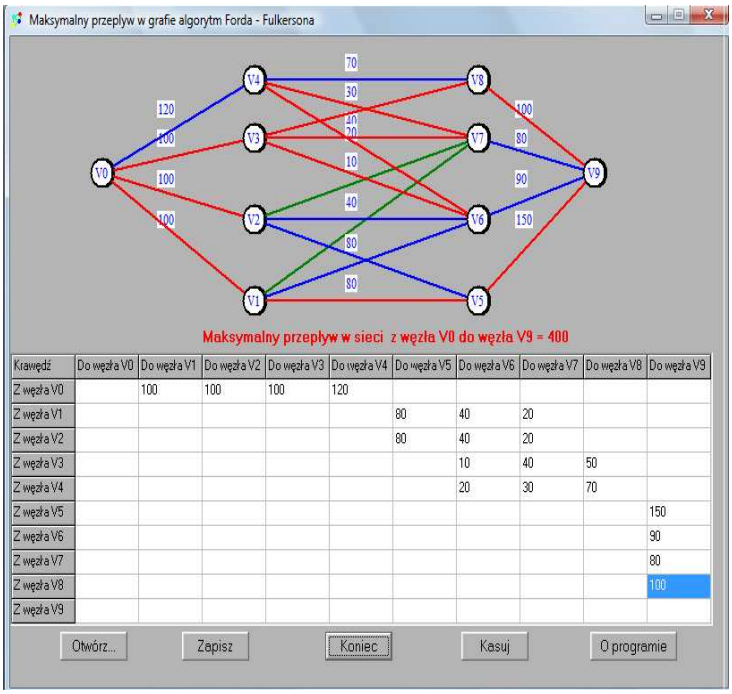


Rys. 4. Wygląd okna programu w trakcie wprowadzania danych.

Źródło: Program Pgraf.exe

W momencie wprowadzania danych pojawiają się krawędzie opisujące graf, obok krawędzi wyświetlana jest wartość przepływu wprowadzona przez użytkownika. Przykład podaje rysunek 4 pokazujący program w trakcie wprowadzania danych przez użytkownika programu.

Po wprowadzeniu wszystkich informacji należy nacisnąć przycisk start, w celu rozpoczęcia wykonywania obliczeń przez komputer. Wynikiem tej operacji będzie ukazanie się końcowego rozwiązania zadania. W sposób graficzny zostało to zaprezentowane na poniższym rysunku (zob. rys.5).



Krawędź	Do węzła V0	Do węzła V1	Do węzła V2	Do węzła V3	Do węzła V4	Do węzła V5	Do węzła V6	Do węzła V7	Do węzła V8	Do węzła V9
Z węzła V0		100	100	100	120					
Z węzła V1						80	40	20		
Z węzła V2						80	40	20		
Z węzła V3							10	40	50	
Z węzła V4							20	30	70	
Z węzła V5										150
Z węzła V6										90
Z węzła V7										80
Z węzła V8										100
Z węzła V9										

Rys. 5 Wygląd ekranu po dokonanych obliczeniach

Źródło: Program Pgraf.exe.

Analiza powyższego rysunku pozwala zaobserwować, iż program prezentuje graf po wykonaniu obliczeń zaznaczając krawędzie różnymi kolorami, które oznaczają odpowiednio:

- Kolorem czerwonym krawędzie nasycone.
- Kolorem niebieskim krawędzie nienasycone, ale biorące udział w obliczeniach.
- Kolorem zielonym krawędzie nie biorące udziału w obliczeniach.

Wartość maksymalnego przepływu zostaje podana przez program pod rysunkiem, w postaci komunikatu „Maksymalny przepływ w sieci z węzła v0 do węzła v9 wynosi …”.

Warto zauważyć, iż rysunek 5 przedstawia obliczenia wykonane na grafie dla przykładu opisanego w rozdziale 2. Czytelnik porównując końcowe wyniki obliczeń wykonywanych ręcznie (zob. rys. 2) z obliczeniami wykonanymi przez program Pgraf.exe (zob. rys. 5) może zaobserwować iż są one identyczne. Potwierdza to prawidłowe działanie programu Pgraf.exe.

Pisząc program autorka ograniczyła się do możliwości posiadania w grafie tylko 10 krawędzi, ale w przyszłości jej celem jest ulepszyć go o ich nieograniczoną ilość przy zachowaniu założeń dotyczących sieci Forda - Fulkersona. Jednak i w tej postaci program ma wiele zalet, chociażby to, iż wynik końcowy jest bardzo prosty i z grafu po obliczeniach można odczytać wiele informacji. Pomocne są też opcje zapisz i otwórz, które mają na celu zaoszczędzenie czasu dając możliwość przechowywania w pamięci raz wprowadzonego grafu. Znikoma jest również na rynku ilość programów komputerowych rozwiązujących zagadnienia transportowe.

5. PODSUMOWANIE

W związku z bardzo szybkim rozwojem informatyki w ostatnich latach stała się ona ważnym elementem w życiu każdego człowieka. Zaprezentowany w artykule program Pgraf.exe obsługujący algorytm Forda-Fulkersona stanowić może początek nowego programu, który po poszerzeniu jego możliwości może stać się w przyszłości nowym produktem na rynku. Wizualnie program jest przejrzysty, obsługa jego nieskomplikowana,

a obsługujący ma możliwość oglądania na ekranie wszystkich danych, które przenoszone są bezpośrednio na graf. Jego zastosowanie może mieć miejsce w wielu dziedzinach, chociażby w logistyce.

A w związku z tym, iż wykorzystanie metod ilościowych daje sposobność znalezienia rozwiązania, a zarazem i odpowiedzi na wiele pytań ich zastosowanie może przynieść wymierne oraz pozytywne efekty, dlatego też zachęcam czytelników do bliższego zapoznania się z nimi.

BIBLIOGRAFIA

[1] Bielecki J., Wprowadzenie do języka C, Wydawnictwo Naukowo – Techniczne, Warszawa 1998.
[2] Grudzewski W., Badania operacyjne w organizacji i zarządzaniu: praca zbiorowa, Polskie Wydawnictwo Naukowe, Warszawa 1995.
[3] Ignasiak E., Borucki W., Badania operacyjne: praca zbiorowa, Polskie Wydawnictwo Ekonomiczne, Warszawa 1996.
[4] Jędrzejczyk Z., Kukuła K., Badania operacyjne w przykładach i zadaniach, Wydawnictwo Naukowe PWN, Warszawa 2002.
[5] Kaufman A, Faure R, Badania operacyjne na co dzień, Polskie Wydawnictwo Ekonomiczne, Warszawa 1968.
[6] Majczak A., Od C do C++ Buildera w 48 godzin, Wydawnictwo Intersoftland, Warszawa 1999.
[7] Ragen A., Leksykon języka C, Wydawnictwo Naukowo –Techniczne, Warszawa 1990.
[8] Stasiewicz A., Wprowadzenie do C++ Buildera, Wydawnictwo Edition 2000, Kraków 2001.
[9] Stroustrup B., Programowanie. Teoria i praktyka z wykorzystaniem C++, Wydawnictwo Helion, Gliwice 2010.
[10] Trzaskalik T., Wprowadzenie do badań operacyjnych z komputerem, Polskie Wydawnictwo Ekonomiczne, Warszawa 2003.

APPLICATION OF C LANGUAGE IN SOLVING THE FORD-FULKERSON ALGORITHM

Abstract:

The article presents author's program Pgraf.exe which has been applied in the Ford-Fulkerson algorithm. Due to the fact that standard calculations by means of algorithm are time-consuming, the featured program allows to save up time, as well as gives an opportunity to store in memory once introduced graph. It can have lots of applications in many areas of research, e.g. in logistics.

In the article there has been described the Ford-Fulkerson algorithm, featured in individual stages. In another chapter on the practical example there has been explained the idea as well as the way of calculation concerning the preceding stages. Since the purpose of the article is to demonstrate the own software of the mentioned algorithm written in C language the third chapter describes the process of writing and designing the program. The last chapter presents author's own software of the Ford-Fulkerson algorithm. In the last chapter there have been presented the conclusions.

Key words: quantitative methods, the Ford-Fulkerson algorithm, logistics, programming in C language.