

JAVA를 이용한 Web-기반 유동 해석 프로그램의 개발

김경성¹, 박종천²

DEVELOPMENT OF A WEB-BASED FLOW ANALYSIS PROGRAM USING OBJECT-ORIENTED LANGUAGE JAVA

K.-S. Kim¹ and J.-C. Park²

객체지향 언어인 JAVA를 이용하여 Web-기반의 유동 해석 프로그램을 개발하였다. 지금까지 유동해석에 관한 대부분의 프로그램은 Fortran, C, C++와 같은 언어로 이루어져 있으며, 이 경우 계산 속도는 빠르지만, 각 언어의 Compiler와 Builder를 필요로 한다. 따라서 사용된 각각의 언어에 대한 Compiler 및 Builder가 사용자의 개인용 컴퓨터 상에서만 설치 및 구동될 수 있으며, 그로 인해 사용자는 계산된 Data형 결과물만을 얻을 수 있었다. 본 연구에서는 이를 개선하고자 객체지향형 언어인 JAVA를 이용하여 유동해석 프로그램을 구성할 수 있는 기반을 구축하였으며, Web과 연동하여 시간적·공간적 제약을 극복하고 사용자의 의견 개입을 가능하게 하였다. 일반적으로 JAVA 언어는 연산속도가 느려서 수치해석용으로는 부적합하다는 평이 지배적이었으나, 컴퓨터의 성능 발달로 이는 개선이 될 수 있으며, 이는 사용자가 시간적·공간적 제약을 받지 않고 사용 가능하다는 점에서 극복되어 질 수 있다.

Key Words : 유동해석(Flow analysis), 웹기반(Web-based), JAVA 언어(JAVA language)

1. 서 론

개인용 컴퓨터의 발달과 수치해석 기법의 발달로 인해 공학에서의 해를 구함에 있어서 수치 해석은 중요한 위치를 차지하게 되었다.

이전까지의 수치해석은 포트란(Fortran), C, C++의 프로그래밍 언어와 매트랩(Matlab)과 같은 수치해석적 툴(tool)을 사용함으로써 공학적 해를 구해 왔다. 이러한 형식의 수치해석은 계산 속도가 빠르다는 장점을 가지고 있는 대신, 사용되는 언어 또는 툴의 컴파일러(Compiler)와 빌더(Builder)가 없으면 사용이 불가피 하고, 이와 관련된 프로그램을 구입하여 개인용 컴퓨터 또는 연산을 수행하고자 하는 컴퓨터에 설치 후 수치해석을 수행하여야 하는 제약이 따르게 된다.

이에 대해, 자바는 컴파일 프로그램과 빌더가 무료로 공개되어 있어, 웹(Web)에 접속할 수 있다면 시간적·공간적 제약 없이 다운로드하여 프로그램의 개발 및 실행 할 수 있다. 또

한 완전한 버전(Full Version)의 컴파일러와 빌더가 아닌 실행 환경 프로그램(Runtime Environment Program)만 설치 한 후 간단히 수치해석을 실행 할 수도 있다.

자바는 웹상에서 사용될 수 있는 객체지향 언어이며, 수치해석 프로그램을 외부기억장치 등의 도움 없이 웹상에서 즉시 적용·실행하여 사용자 의견이 반영된 프로그램 개발을 비교적 용이하게 할 수 있다.

한편, 단점으로는 연산속도가 느려 수치해석용으로는 부적합하다는 평이 지배적이었으나 지속적인 컴퓨터의 성능 발달로 극복되어 질 수 있으며, 상술한 장점들로 인하여 보완이 가능하다[1].

본 연구에서는 객체지향의 웹기반 언어인 자바를 이용하여, 대류항의 수치적인 이산화 과정을 통해 공간 이산화에 각종 수치도식(scheme)을 적용하고, 이를 후처리 하는 과정을 개발하기로 한다. 개발된 프로그램은 웹에서 사용가능한 애플릿(Applet) 프로그램으로 작성함으로써 향후 웹기반의 유동 해석을 위한 기반을 마련하기로 한다.

1 학생회원, 부산대학교 조선해양공학과

2 정회원, 부산대학교 조선해양공학과

* Corresponding author, E-mail: jcpark@pusan.ac.kr

2. 수치해석 알고리즘 구현

2.1 문제 설정

식 (1)의 테스트 문제로는 Fig. 1과 같이 ϕ 가 계단함수로 표현되는 경우를 택하였다. 함수 ϕ 의 초기 값으로는 $x=4 \sim 7$ 의 영역에 $\phi=1$ 의 값을, 나머지 영역에서는 $\phi=0$ 의 값을 주었다. 격자간격 Δx 는 0.05 이며, 시간간격 Δt 를 조정하게 하여 Courant 수를 변경 가능하게 하였다.

2.2 알고리즘

2.2.1 애플릿(Applet)

자바 언어로 작성된 소프트웨어 중 별도의 애플릿 형식을 뜻한다. 크기가 작아서 네트워크에서의 전송에 적합하고, 웹 상에서 배포가 가능하다. 웹에서 사용하는 표준 데이터 형식인 하이퍼텍스트 생성 언어(HTML)로 작성한 문서에 애플릿이라는 태그를 사용하여 자바 애플릿을 지정한다. 자바 애플릿을 동작시키는 데는 자바 가상 머신을 내장한 웹 브라우저가 필요하다. 브라우저는 불러내 온 문서 속에 애플릿이라는 태그가 있으면, 지정된 자바 애플릿을 웹 서버로부터 다운로드하여 실행한다(Fig. 2).

2.2.2 데이터 처리 방식

Fig. 3은 수치해석 및 데이터 처리 과정을 보여준다. 먼저 연산 클래스에서 연산을 한 뒤, 데이터 클래스에서 각 데이터를 수집 및 분류를 한다. 그 뒤 데이터 클래스에서 수집 및 분류되어진 값과 그리기 클래스의 설정을 스레드(Thread)가 참조하여 수집 및 분류되어진 데이터를 그리기를 한다. 이 일련의 과정을 애플릿에서 구동하게 하여 웹과 연동되어지는 수치해석 프로그램이 만들어 진다[2].

2.3 수치 도식

1차 상류차분(1st Up-wind Scheme)과 중심차분, CIP(Cubic-interpolation pseudoparticle) Scheme, QUICK (Quadratic Upstream Interpolation for Convective Kinematics) Scheme 그리고 Lax-Wendroff(이하, L-W) 도식[3]을 자바 언어를 사용하여 보존형으로 표현되는 식 (1)의 1차원 파동(대류) 방정식을 수치 시뮬레이션 하였다.

$$\frac{\partial \phi}{\partial t} = -U \frac{\partial \phi}{\partial x} \quad (1)$$

단, U 는 x -방향에서의 이류속도를 나타낸다.

물리량 ϕ_i 를 격자중심에 배치하는 정규격자를 채용하여 식 (1)을 유한체적법으로 이산화 한다. 시간적분은 1차 정확

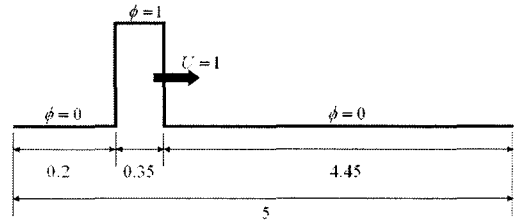


Fig. 1. 1-D test model for solving wave equation.

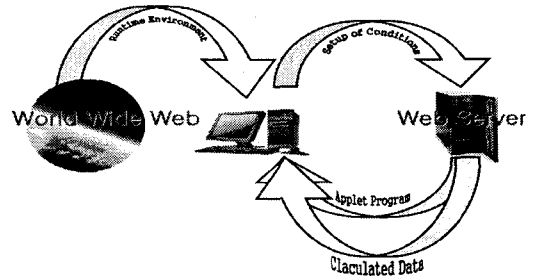


Fig. 2 System of program

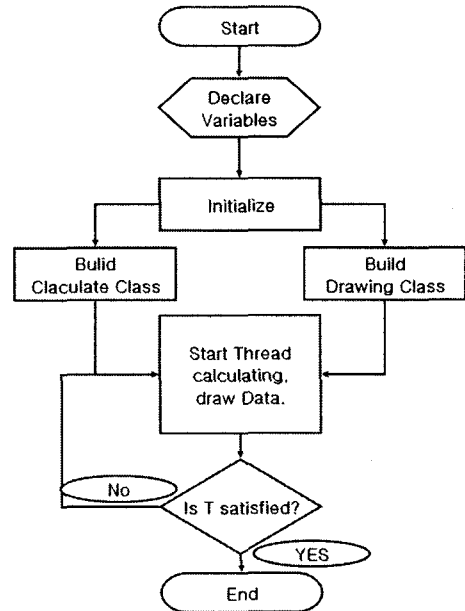


Fig. 3 Algorithm for JAVA program

도의 오일러(Euler) 도식을 적용하면,

$$\phi_i^{n+1} = \phi_i^n - \Delta t \frac{F_{i+1/2}^n - F_{i-1/2}^n}{V_i} \quad (2)$$

여기서 V_i 는 격자 i 의 체적, $F_{i+1/2}$ 은 격자경계 $i+1/2$ 의 운동량 Flux 이다. 수치해석에서는 앞서 말한 5종류의 수치도식이 적용되었다.

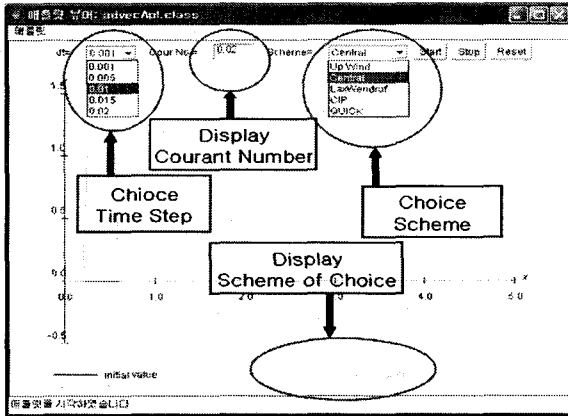


Fig. 4 Sample of Applet program.

2.4 수치 프로그램 실행

2.4.1 초기 조건

Fig. 4 에서 보이는 바와 같이, 본 프로그램에서는 시간간격과 수치도식을 사용자가 선택할 수 있으며, Δx 의 값은 프로그램 내에서 지정된 0.05를 쓰고, 변화된 시간간격에 따라 Courant 수가 자동적으로 표시된다. 또한 수치도식을 선택하면 좌측 하단에 초기 값과는 다른 색으로 선택된 수치도식이 표시된다. 좌측 상단에는 Start, Stop, Reset의 세 개의 버튼을 선택할 수 있고, Start는 연산 및 그리기 시작을, Stop은 연산 및 그리기 멈춤을, 그리고 Reset은 초기 형태로의 복원을 각각 실행하는 버튼이다. 연산 및 그리기는 실시간으로 데이터를 받아서 매 시간간격마다 계산된 값을 그리도록 되어 있다.

2.4.2 실행 결과

Fig. 5는 시간 간격 0.001에서 격자간격 0.05인 조건 하에서 계단 함수의 시간적 전파 과정에 따른 공간적 분포를 보인다. 상류차분 도식의 경우에는 심한 수치적 확산과 소산에 의해 원래 형태의 함수를 알아보기가 힘들고, 중심차분 도식에서는 심한 수치적 진동을 보인다. 그 외의 수치도식의 경우에 대해서도 이전의 선행되었던 결과들과[4]과 비슷한 결과를 나타내었다.

다음으로 시간간격의 변화에 대해서도 실험을 행하였다. 대상 수치도식은 수치도식들 중에서 시간간격의 변화에 민감하게 반응했던 QUICK 도식을 사용하여 시간간격 $\Delta t=0.001, 0.005, 0.01, 0.015, 0.02$ 에 대하여 계산을 수행하였다. Fig. 6에서 보이는 바와 같이, 시간간격이 증가함에 따라 최대값이 초기값과는 큰 차이를 보였으며, 수치진동역시 증가함을 확인할 수 있었다.

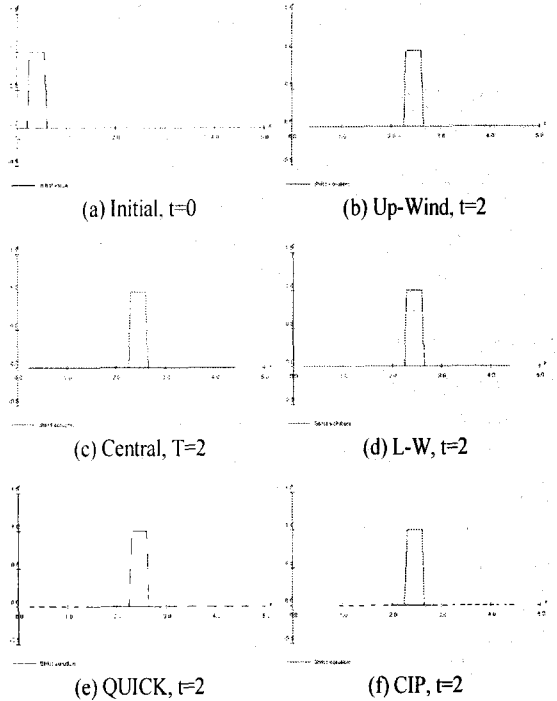


Fig. 5. Results of simulation, $\Delta t=0.001, \Delta x=0.05$.

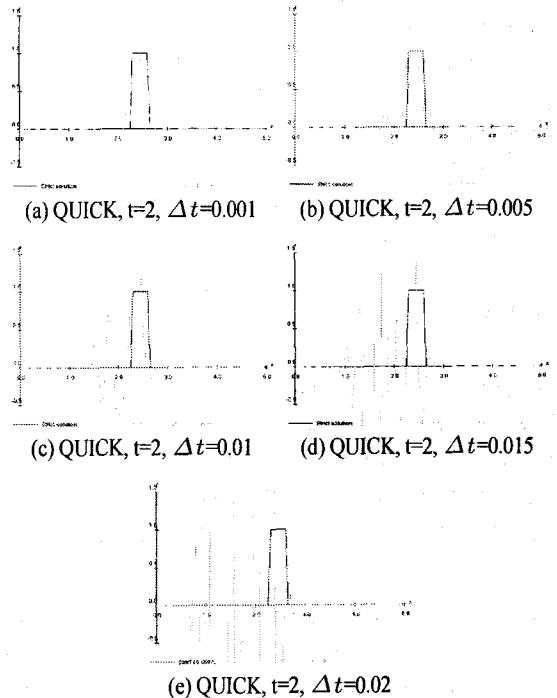


Fig. 6. Case of QUICK scheme.

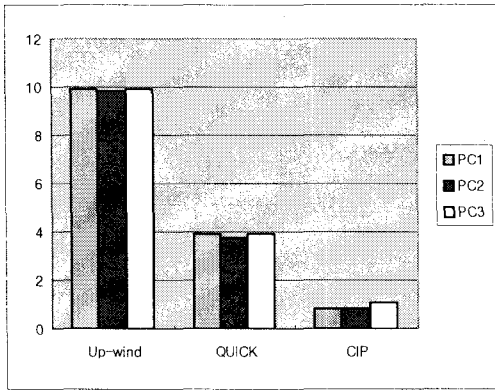


Fig. 7 Comparison of CPU time.

2.5 계산시간 비교

계산된 CPU 시간을 비교하기 위해, 제안된 수치도식들 중에서 Up-wind, QUICK 그리고 CIP 도식에 대해서 각기 다른 컴퓨터를 사용하여 계산을 실행하였다. 단, 무차원 계산 시간이 $t=2.0$ 일 때까지 계산을 수행하였다. Table 1에는 계산에 사용되어진 컴퓨터의 사양을 표시한다. Fig. 7에 보이는 것과 같이 동일한 수치도식에 동일한 시간간격을 설정하더라도 컴퓨터의 사양에 따라 CPU 시간에 큰 차이가 있음을 확인할 수 있다. 이로써 JAVA 언어의 연산속도가 느리다고 하는 단점은 컴퓨터의 발달에 의해서 보완되어짐을 확인할 수 있었다.

마지막으로, 제안된 수치도식 중 Up-wind, 중심차분, 그리고 QUICK 도식에 대해서 PC2를 사용하여 포트란 언어와 자바 언어, 그리고 그래픽 처리가 가미된 자바 언어와의 CPU 시간을 비교해 보았다. Table 2에서 보는 것과 단순 연산에

대해서 포트란과 자바의 연산속도는 차이를 보이지 않았다. 한편, 그래픽이 처리가 가미된 경우는 자바가 현저히 낮은 경향을 보였으나 복잡한 그래픽 처리를 동시수행 한다는 관점에서는 타당한 결과라고 볼 수 있다. 이때 그래픽 처리의 점유율은 전체 CPU 시간의 96%에 해당한다.

3. 결론

본 연구에서는, 자바를 이용하여 웹과 연동 가능하며 사용자의 실정이 가능한 애플릿 프로그램을 개발하였으며, 유동장 해석의 기본이 될 수 있는 비정상 이류 문제에 적용하였다. 수치해석된 결과는 자바의 그리기 기능을 이용하여 후 처리 하도록 하였다. 자바의 단점인 연산시간에 대하여, 각종 수치 도식, 컴퓨터, 그리고 프로그래밍 언어별 비교를 통하여 CPU 시간의 차이를 확인 하였다. 그래픽 처리 과정이 배제된 경우, 연산속도가 늦다는 JAVA의 단점은 컴퓨터의 발달로 인해 보완될 수 있다는 점을 확인할 수 있었다.

참고문헌

- [1] Walter Savitch, 2006, "Absolute JAVA," Addison Wesley, University of California, San Diego.
- [2] 峯材 吉泰, 2006, "JAVAで学ぶ シミュレーションの基礎," 森北出版株式会社, 東京.
- [3] 越塚越一, 2004, "數值流體力學," 培風館 東京.
- [4] 박종천, 이병혁, 김정후, 2006, "자유표면 유동 시뮬레이션을 위한 고정확도 수치도식의 검토," 한국해양공학회지, 제20권 제4호, pp.31-36.

Table 1 Spec of computers used in benchmark tests.

	CPU	RAM	VGA
PC1	Intel Pentium(R) M 1.4GHz	251MHz, 256MB	Nvidia GeForce4 440 Go
PC2	Intel Pentium(R) 4 3.4GHz	3.41GHz, 2.0 GB	Nvidia GeForce 6600
PC3	Intel Core(TM) 2 CPU 6600, 2.4GHz	2.4GHz, 4.0GB	Nvidia Geforce 7300 LE

Table 2 Comparison of CPU time in PC2.

	JAVA	JAVA W/ Graphics	Fortran
Up-wind, $\Delta t=0.001$	0.0160s	3.72s	0.0156s
Central, $\Delta t=0.001$	0.0160s	3.72s	0.0156s
L-W, $\Delta t=0.001$	0.0155s	3.51s	0.0155s