울산대학교 경영학 연구논문집 제3권 2호 pp. 1~20, 1996.
Journal of Management Vol. 3, No. 2, pp. 1~20, 1996
University of Ulsan.

객체지향기법을 이용한 파생금융상품의 가치평가 시스템 개발 전략

조희연
경영정보학과

I. 서 론

파생금융상품(Derivatives)이란 이름에서 알 수 있듯이 주식이나 채권, 환동의 기초증권(Primary Securities)이나 다른 파생금융상품을 기초로 해서 발행되는 2차 금융상품으로 투자가에게 위험을 관리하는 수단이나 투기의 대상으로 이용될 수 있다. 파생금융시장은 최근 수년간 국제 금융시장의 통합가속화, 정보통신 기술의 발달, 금융기관의 상품개발능력의 신장 등으로 인하여 급속하게 성장하여 왔다. 파생금융시장의 규모가 급팽창하면서 투자리스크도 점차 커지고 있으며 특히 최근에는 파생금융상품의 상품구조가 워낙 복잡하기 때문에 기업이나 금융기관들이 이에 대한 위험을 잘 알지 못하는 상황에서 거래를 함으로써 손해를 보는 경우가 많다. 파생금융상품에 대한 위험은 세계적 금융기관인 영국계의 베어링 부라더스 은행이 파생금융상품을 운용하던 펀드 메니저 한 사람의 실수로 문을 닫게 된 사건으로부터도 잘 알 수 있다. 파생금융상품시장의 본거지라고 할 수 있는 미국에서도 파생금융상품 거래확대와 더불어 이 거래로 손실을 입는 회사가 많이 늘어나자 파생금융상품 거래에 대한 근본적인 정책을 수정하는 기업이 늘어나고 있다. 과거에는 파생금융상품의 거래에서 손실이 발생하면 담당 부서 차원에서 전략을 수정하고 책임자를 징계하는 소극적인 관리를 해 왔으나 최근에는 담당직원에서 최고경영자에 이르는 전사적인 차원에서 위험관리 전략을 수립하고 있다. 우리 나라의 파생금융상품시장은 아직 초보적인 단계이지만 올해 개설된 주가지수 선물 시장을 비롯하여 내년에 도입 예정인 옵션시장등 향후 전개될 금융자유화는 우리 나라 파생금융시장을 급속히 발전시킬 것으로 기대된다. 이에 따라 국내 금융기관들은 파생금융상품에 대한 높은 관심을 보이면서 전문딜러 양성 및 파생금융상품에 대한 시스템 구축을 서두르고 있다. 파생금융상품은 고객의 다양한 욕구를 충족시키기 위하여 생성되는 금융상품이기 때문에 종류가 다양할 뿐만 아니라 그 수 또한 헤아릴 수 없을 정도로 많다 현재 선진 금융시장에서는 파생상품의 거래가 기초증권을 압도하고 있으며 향후 우리 나라에서도 이와 같은 현상이 발생될 것으로 기대된다. 따라서 국내 금융기관의 경쟁력은 새로운 파생금융상품의 개발 및 파생상품의 위험관리 능력에 달려 있다고 할 수 있다. 파생상품은 특성상 종류가 다양하고 그 수가 매우 많으며 상품구조가 복잡하기 때문에 파생금융상품에 대한 가치평가 및 위험관리에는 투자시스템의 이용이 불가피하다. 따라서 현재 투신을 비롯한 여러 증권회사에서 파생금

융 상품에 대한 시스템 개발이 활발히 이루어지고 있다.

이러한 추세에 따라서 본 연구에서는 객체 지향기법을 이용하여 대표적인 파생금융 상품인 옵션의 가격결정 방법에 대한 시스템 개발 모형을 제시하고자 한다. 객체지향 기법은 인공지능, 소프트웨어 공학, 시뮬레이션 등 여러 학문분야에서 개념과 구현방법이 도출되었다. 객체지향 기법은 객체위주의 문제파악 및 시스템 설계를 통하여 시스템의 모듈화를 추구하고 이로 인하여 소프트웨어의 재사용성, 확장성, 변경성이 용이하다는 장점을 갖고 있다.

시스템 구축 측면에서 볼 때 파생금융상품의 가격결정모형은 다음과 같은 특징을 갖는다. 첫째, 파생금융상품은 기초증권이나 다른 파생금융상품을 기반으로 발행되기 때문에 부모와 자식관계가 형성되고 이에 따라 특성이나 가격결정방법등의 상속(Inheritance)문제가 많이 발생한다. 둘째, 파생상품의 특성상 고객의 새로운 욕구가 발생할 때마다 새로운 파생상품이 개발되어야 하기 때문에 파생금융상품에 대한 시스템은 확장성이 뛰어 나야 한다. 셋째, 파생금융상품은 특성상 기초증권별 또는 가격결정방법에 따라 상호 관련된 상품별로 group화가 용이하며 이들은 계층을 형성한다.

이와 같은 파생금융상품의 특성은 객체지향기법의 효과적인 적용을 가능케 한다. 그러므로 객체지향기법을 이용한 파생금융상품 시스템 구축은 프로그래밍의 효율성을 높이는 동시에 새로운 파생상품의 추가나 변경 등 확장성이 좋게 된다. 본 연구에서는 이러한 객체지향기법의 장점을 이용하여 옵션모형개발을 위한 클래스 구조, 객체의 속성 및 멤버함수를 설계하고 이를 객체 지향언어인 Visual C++를 이용하여 구현함으로써 향후 활발히 이루어질 파생금융상품에 대한 시스템개발에 방향을 제시하고자 한다.

II. 옵션의 개념 및 가치 평가 기법

2.1 옵션의 개념 및 기본 옵션

옵션이란 특정상품을 일정가격으로 향후 일정기간 또는 그 기간 내에 사거나 팔 수 있는 권리를 말한다. 옵션은 권리의 종류에 따라서 살 수 있는 권리를 콜 옵션(Call Option), 팔 수 있는 권리를 풋 옵션(Put Option)으로 구분한다. 또한 옵션의 권리행사시점에 따라서 옵션의 만기일에만 권리를 행사할 수 있는 유럽식 옵션(European Option)과 만기일 이전에 어느 때라도 권리를 행사할 수 있는 미국식 옵션(American Option)으로 구분된다. 그러므로 기본적인 옵션의 종류는 4가지 형태가 된다. 그러나 이러한 단순옵션도 기초 증권으로 주식, 선물, 환, 지수 등 여러 금융자산을 가질 수 있기 때문에 다양한 형태가 될 수 있다.

단순 옵션의 가격 결정 요인은 Black & Sholes(1973)의 옵션 평가 모형으로부터 다음의 5가지로 요약 될 수 있다. 첫째는, 기초증권(S)과 옵션의 행사가격(EX)으로 콜옵션의 경우 손익구조가 MAX(S-EX, 0) 이므로 기초증권의 가격이 올라갈수록, 행사가격이 낮을 수록 가치가 커지게 된다. 풋옵션의 경우는 손익구조가 MAX(EX-S, 0)이므로 기초증권의 가격이 내려갈수록, 행사가격이 높을 수록 가치가 커지게 된다. 둘째는, 기초자산의 수익률의 분산(Sigma)이다. 기초자산의 수익률의 분산이 크다는 것은 향후 기초자산의 가격변동폭

이 크다는 것을 의미한다. 콜 옵션의 소유자는 기초증권의 가격하락 위험에는 방어되면서 동시에 가격상승의 이익은 모두 향유할 수 있기 때문에 기초증권의 가격 변동이 심할 수록 유리하게 된다. 풋옵션 소유자 역시 기초증권의 가격상승에서 발생되는 손실에는 방어되면서 기초증권의 가격하락에서 발생하는 이득은 모두 향유할 수 있기 때문에 기초증권의 가격 변동이 심할 수록 유리하게 된다. 그러므로 기초자산의 수익률의 분산이 클 수록 옵션의 가격은 증가한다. 셋째는 만기로써 만기가 길어질 수록 미국식 옵션의 경우는 권리를 행사할 수 있는 기회가 증가되기 때문에 옵션 가치가 증가된다. 네 번째 요인은 무위험 수익률로써 무위험 수익률이 옵션에 주는 영향은 옵션의 종류에 따라서 달리 나타난다. 콜옵션의 경우에는 무위험 수익률이 증가하면 옵션가치가 증가하는 반면 풋옵션의 경우는 가치가 감소한다. 다섯째로는 배당으로써 배당후 주가는 배당만큼 떨어지기 때문에 배당의 증가는 콜옵션에는 부정적 효과를 주며, 풋옵션에는 긍정적 효과를 준다.

2.2 복합 옵션(Exotic Option)

앞에서 언급한 바와 같이 파생금융상품은 고객의 욕구에 따라서 자유롭게 새로운 상품이 개발되기 때문에 옵션 역시 기본옵션 이외에 다양한 형태의 변형된 옵션이 존재하게 된다. 복합옵션의 경우 대부분이 시장의 특별한 욕구를 충족시키기 위해 금융기관에서 개발되며 표준화가 이루어져 있지 않기 때문에 점두시장에서 주로 거래된다. 복합옵션은 손익구조에 따라서 10가지 정도로 분류할 수 있다.(Rubinstein, 1991, 1992) 이 장에서는 본 연구가 다룰 이항옵션(Binary Option)에 대하여 설명하고자 한다. 이항옵션은 기본옵션과는 달리 불연속적인 손익구조를 갖는다는 특징을 갖고 있다. 즉 기초증권이 어떠한 조건을 만족시키느냐에 따라서 손익구조는 이항 형태를 갖게 된다. 대표적인 이항옵션들을 언급하면 다음과 같다.

2.2.1 Path-independent 이항 옵션

a. Cash-or-nothing calls : 가장 간단한 이항 옵션으로 옵션만기에 기초증권의 가격(S)이 행사가격(EX) 이하로 끝나는 경우에는 아무것도 지급치 않지만 기초증권의 가격이 행사가격 보다 높게 끝나는 경우에는 미리 결정된 금액(X)을 지급하는 옵션이다.

$$Payoff = \begin{cases} 0 & \text{if } S \le EX \\ X & \text{if } S > EX \end{cases}$$

b. Asset-or-nothing calls : 옵션만기에 기초증권의 가격(S)이 행사가격(EX) 이하로 끝나는 경우에는 아무것도 지급치 않지만 기초증권의 가격이 행사가격 보다 높게 끝나는 경우에는 기초증권의 가격만큼 지급하는 옵션이다.

$$Payoff = \begin{cases} 0 & \text{if } S \le EX \\ S & \text{if } S > EX \end{cases}$$

2.2.2 Path-dependent 이항 옵션

Path-dependent 이항 옵션은 이항옵션에 Barrier 옵션의 성격이 결합된 옵션으로 옵션의 손익구조는 옵션의 만기이내나 만기동안 기초증권의 가치가 일정한 값(Barrier)에 도달하느냐에 따라서 결정된다. 즉 옵션의 가치는 일정기간동안의 기초증권 가격변화의 Path에 의존하게 된다. 이러한 성격을 갖고 있는 옵션은 현재 개발된 것만 해도 수십 종류가 넘는다. 본 연구에서는 이중에서 28종류의 옵션에 대하여 평가 모형을 수립한다. 대표적인 몇 가지 옵션 형태를 설명하면 다음과 같다.

a. Down-and-in cash-(at hit)-or-nothing : 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하로 떨어지면 그 시점(at hit)에 미리 결정된 금액(X)을 지급하는 옵션이다.

$$Payoff = \begin{cases} X(at\ hit) & \text{if for } some\ z \le t,\ S(z) \le H,\ t\text{는 만기싯점} \\ 0 & \text{if for } all\ z \le t,\ S(z) > H,\ t\text{는 만기싯점} \end{cases}$$

b. Up-and-out cash-or-nothing : 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하를 계속 유지하면 만기시점에 미리 결정된 금액(X)을 지급하는 옵션이다. 만일 만기동안 기초증권의 가치가 Barrier인 H를 넘게 되면 그 시점에서 그 옵션은 사라져 버리게 된다.

$$Payoff = \begin{cases} X(at\ maturity) & \text{if for } all\ z \le t,\ S(z) < H,\ t\text{는 만기싯점} \\ 0 & \text{if for } some\ z \le t,\ S(z) \ge H,\ t\text{는 만기싯점} \end{cases}$$

c. Down-and-in cash-or-nothing call : 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하로 떨어진 적이 있고 만기시점에 기초증권의 가치가 행사가격(K)보다는 큰 경우 미리 결정된 금액(X)을 지급하는 옵션이다.

$$Payoff = \begin{cases} X(at\ maturity) & \text{if for } some\ z \le t,\ S(z) \le H,\ \text{and } S(t) > K \\ 0 & \text{if for } all\ z \le t,\ S(z) \ge H,\ or\ S(t) < K \end{cases}$$

d. Down-and-in asset-or-northing call : 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하로 떨어진 적이 있고 만기시점에 기초증권의 가치가 행사가격(K)보다는 큰 경우 기초증권의 가치를 지급하는 옵션이다.

$$Payoff = \begin{cases} S(t)(at\ maturity) & \text{if for } some\ z \le t,\ S(z) \le H,\ \text{and } S(t) > K \\ 0 & \text{if for } all\ z \le t,\ S(z) \ge H,\ or\ S(t) < K \end{cases}$$

e. Up-and-out cash-or-nothing call: 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하를 계속 유지하면서 동시에 만기시점에 기초증권의 가치가 행사가격(K)보다 큰 경

우 미리 결정된 금액(X)을 지급하는 옵션이다. 만일 만기동안 기초증권의 가치가 Barrier 인 H를 넘게되면 그 시점에서 그 옵션은 사라져(out) 버리게 된다.

$$Payoff = \begin{cases} X(at\ maturity) & \text{if for } all\ z \le t,\ S(z) < H,\ \text{and } S(t) > K \\ 0 & \text{if for } some\ z \le t,\ S(z) \ge H,\ or\ S(t) < K \end{cases}$$

2.3 옵션에 대한 평가기법

2.3.1 해석적 해(Closed Form Solution)

유럽식 기본옵션이나 기타 여러 복합옵션들은 해석적 해를 갖게 되는데 이러한 옵션에 대한 해석적 해는 Black-Sholes의 모형(1973,JPE)을 기반으로 유도된다. 옵션에 대한 해석적 해를 유도하기 위해서는 다음의 가정을 요구한다. 첫째로, 기초증권의 가격(S)은 일정한 분산(σ^2)을 갖는 확산모형(Diffusion Process)을 따른다. 둘째로 자본시장은 완전자본시장으로 모든 투자자는 비용 없이 모든 정보에 접근 가능하다. 셋째, 무위험 수익률(r)의 기간구조는 수평이고 확정적이다. 이러한 가정 하에서 재정이득이 존재하지 않는다는 조건(No Arbitrage Condition)과 개별 파생상품의 경계조건을 적용시키면 그 상품에 대한 가치를 구할 수 있다. 본 연구에서 사용되는 해석적 해를 유도하면 다음과 같다.

a. 유럽식 기본 옵션에 대한 해석적 해

$$Value = \Phi S e^{-dt} N(\Phi d_1) - \Phi E X e^{-rt} N(\Phi d_1 - \Phi \sigma \sqrt{t})$$

$$d_1 = \frac{\log(Se^{-dt}/EXe^{-rt}) + \frac{1}{2}\sigma^2 t}{\sigma\sqrt{t}}$$

S = 현재주가, EX = 행사가격, t = 만기,

r = 시중무위험이자율, d = 배당률, σ =주식수익률의 표준편차,

$N(x)$는 누적정규분포를 나타냄

위 식에서 Φ가 1이면 콜 옵션의 가치가 되고 Φ가 0이면 풋옵션의 가치가 된다.

b. 이항 옵션에 대한 해석적 해

본 연구에서 고려하는 32가지의 이항옵션은 유럽식 기본 옵션에 대한 공식 및 다음의 9가지 공식의 결합으로 구할 수 있다. 단 A는 이항 asset옵션의 가격결정을 위해 사용된다는 의미이고 C는 이항 cash옵션의 가격결정을 위해 사용된다는 의미이다.

$$1A = Se^{-dt}N(\Phi x)$$
$$1C = Xe^{-rt}N(\Phi x - \Phi\sigma\sqrt{t})$$
$$2A = Se^{-dt}N(\Phi x_{1)}$$
$$2C = Xe^{-rt}N(\Phi x_1 - \Phi\sigma\sqrt{t})$$
$$3A = Se^{-dt}(H/S)^{2\lambda}N(\pi y)$$
$$3C = Xe^{-rt}(H/S)^{2\lambda-2}N(\pi y - \pi\sigma\sqrt{t})$$
$$4A = Se^{-dt}(H/S)^{2\lambda}N(\pi y_{1)}$$
$$4C = Xe^{-rt}(H/S)^{2\lambda-2}N(\pi y_1 - \pi\sigma\sqrt{t})$$
$$6 = X[(H/S)^{a+b}N(\phi z) + (H/S)^{a-b}N(\pi z - 2\pi b\sigma\sqrt{t})$$
$$\mu = \log(\frac{r}{d}) - \frac{1}{2}\sigma^2$$
$$x = [\log(S/EX) + \sigma\sqrt{t}] + \lambda\sigma\sqrt{t}$$
$$x_1 = [\log(S/H) + \sigma\sqrt{t}] + \lambda\sigma\sqrt{t}$$
$$y = [\log(H^2/S*EX) + \sigma\sqrt{t}] + \lambda\sigma\sqrt{t}$$
$$y_1 = [\log(H/S) + \sigma\sqrt{t}] + \lambda\sigma\sqrt{t}$$
$$z = [\log(H/S) + \sigma\sqrt{t}] + b\sigma\sqrt{t}$$
$$\lambda = 1 + \mu/\sigma^2$$
$$a = \mu/\sigma^2$$
$$b = \sqrt{[\mu^2 + 2(\log r)\sigma^2]/\sigma^2}$$
$$\pi,\ \Phi = 1\ or\ -1$$

2.3.2 근사적 해법 (Approximation Method)

미국식 기본옵션이나 상당수의 복합옵션은 해석적 해가 존재하지 않기 때문에 근사적 해법을 이용하여 가치를 평가해야 한다. 근사적 해법에는 이항모형(Binomial Model), 유한차분법(Finite Difference Method), 몬테칼로 시뮬레이션(Monte-Carlo Simulation), 수치적 분방법(Numerical Integration)등이 있는데 투자 관리 분야에서 많이 사용되는 방법은 이항모형과 유한차분기법이다. 본 연구에서는 이중에서 이항모형을 이용한 옵션의 가격평가를 다룬다. 이항모형은 단위시간에 기초증권의 가격이 두개의 값만을 취할 수 있다는 가정 하에서 출발한다. 이러한 가정 하에서 재정이득이 존재하지 않는다는 조건을 적용시키면 옵션에 대한 가치를 구할 수 있다.

III. 객체 지향기법을 이용한 옵션 평가 모형 개발

객체지향기법은 인공지능분야나 소프트웨어공학, 시스템 시뮬레이션, 컴퓨터 그래픽스등 여러 학문분야에서 그 개념과 구현방법이 도출되었다. 인공지능분야에서는 지식표현의 기본 틀을 제공하는 프레임이론을 응용 구현하기 위한 도구로써 객체지향기법이 응용될 수 있다. 소프트웨어 공학에서는 특정 프로젝트에서 개발된 프로그램을 다른 프로젝트에서 재사용하도록 하는 소프트웨어 재사용 개념이 추구되어 왔는데 객체지향기법은 이에 대한 가능성을 제시하고 있다. 또한 객체지향기법은 모듈 화를 손쉽게 구현할 수 있는 방법이며, 프로그램의 사후관리비용을 절감시키는데 기여할 수 있다. 또한 프로그래밍 언어 분야에서도 객체지향기법은 복잡하고 추상적인 자료구조를 제공함으로써 데이터 추상화를 위한 사용자 정의의 자료구조를 제공에 기여할 수 있다

객체 지향기법은 객체 위주의 시각에서 시스템구축을 할 수 있으며, 또한 모듈화, 재사용, 확장성, 변경성이 용이하며 다형성과 동시작업이 가능하게 되는 장점을 갖고 있다. 이러한 객체 지향기법을 구현할 수 있는 객체 지향 프로그래밍언어는 Simula라는 시뮬레이션 언어에서 영향을 받아 1972년에 개발된 Smalltalk가 최초라고 할 수 있다 그후 C++, CLOS(Common Lisp Objective System), Flavor등 많은 객체지향 언어가 개발되었다. 본 연구에서는 이러한 객체지향기법의 장점을 이용하여 옵션모형의 개발을 위한 클래스 구조, 객체의 속성 및 멤버함수를 설계하고 이를 객체지향언인 Visual C++를 이용하여 구현하였다.

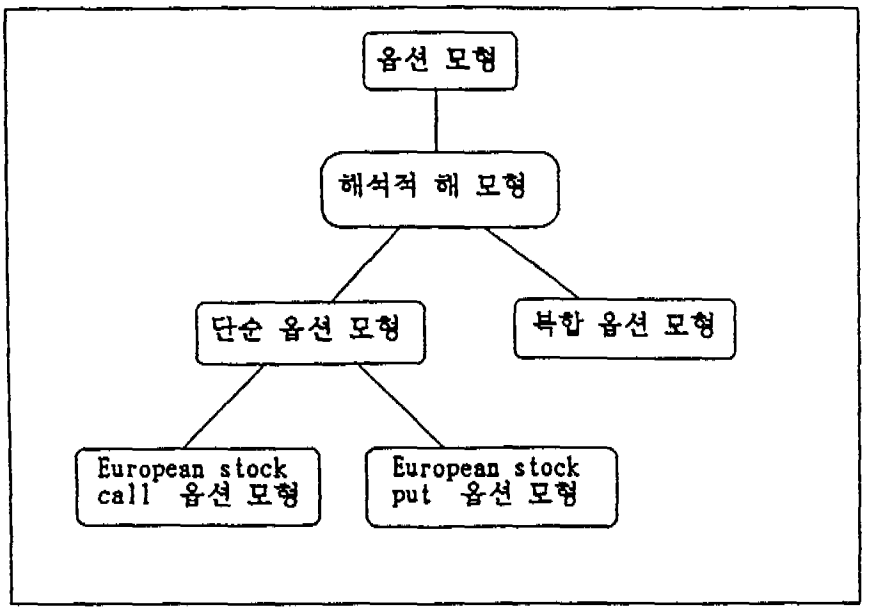
3.1 목표 분석

파생금융상품에 대한 정보시스템 구축에 있어서 가장 중요한 부분은 파생금융상품의 가치를 평가하는 부분이다. 옵션은 특성상 종류가 다양하고 그 수가 매우 많으며 상품구조가 복잡하기 때문에 옵션상품에 대한 가치평가는 복잡한 수학공식이나 근사적 기법이 이용된다. 본 시스템은 기본형태의 옵션 및 복합옵션중 이항옵션에 대한 모형적 가치를 제공하는 것을 목표로 한다.

3.2 해석적 해를 갖는 옵션의 가격결정 모형 개발

3.2.1 클래스 구조

해석적 해란 옵션의 가격 결정 공식(Closed Form Solution)을 도출할 수 있는 경우로써 해석적 해를 갖는 옵션은 상품구조의 복잡성에 따라서 단순옵션과 복합옵션으로 구분된다 <그림 1>에 객체 클래스의 계층적 구조가 나타나 있다.



<그림 1> 해석적 해를 갖는 옵션 모형의 클래스 구조

3.2.2 객체 속성 및 멤버 함수

(1) 옵션모형

옵션의 가치평가시 옵션의 가치에 공통적으로 영향을 주는 요인은 기초증권의 가격(S), 기초증권의 수익률 분산 (σ^2), 무위험 수익률(r), 옵션의 만기(t), 옵션의 행사가격(EX), 기초증권의 배당률(d)등이다. 그러므로 이러한 요인들은 최상위 클래스의 속성으로 정의하고 하위 클래스로 상속되게 된다. 최상위 클래스로서의 옵션모형 객체는 다음과 같이 표현할 수 있다.

```
class OptionModel {
public:
        //Attributs
        float t, sigma, S, EX, r, d;
        //Constructor
        OptionModel(float t float sigma, float S, float r, float, d, float EX)
                {t=t; sigma=sigma; S=S; r=r; d=d; EX=EX;};
};
```

(2) 해석적 해 모형

2단계 계층으로는 옵션의 평가기법에 의한 클래스 구분으로 해석적 해가 존재하는 옵션들의 집합과 근사적으로 해를 구해야 하는 옵션들의 집합이다. 해석적 해를 갖는 옵션 모형들은 다음의 두 가지 멤버함수를 갖는다.

. 누적 정규분포 계산 함수
. 옵션 평가 공식 중에서 x값 계산 함수

```
classClosedFormSolution : public OptionModel {
public:
        //Attributs
        float x,
        //Constructor
         ClosedFormSolution(float t float sigma, float S, float r, float, d, float EX):
      OptionModel(t, sigma, S, r, d, EX) {};

        // Member Functions
         virtual void calculate_x()
        {x=(log(S*exp(-d*t))/(EX*exp(-r*t))) + (0.5*sigma*sigma*t))/(sigma*sqrt(t));
      };
        virtual float calculate_normal(float x);
};
```

a. 단순 옵션 모형

단순옵션 클래스에 속하는 모든 옵션은 공통적인 가치평가 공식을 갖게 된다. 이 공식은 해석적해 모형 클래스의 멤버함수를 상속받아 이용하게 된다.

```
class ClosedFormSimpleOption :   public ClosedFormSolution {
public:
        // attributes
        float OP,
        //constructor
        ClosedFormSimpleOption(float t, float sigma,
        float S,    float r, float d, float EX)
        :   ClosedFormSolution(t, sigma,S,r, d, EX) {};

        // member functions
        virtual float calculate_option_value(int phi)
        {  calculate_x();
```

```
        OP - phi*S*exp(-d*t)*calculate_normal(phi*x) -
        phi*EX*exp(-r*t)*calculate_normal(phi*x-phi*sigma*sqrt(t));
        return OP, };
        virtual void display(ostream&);
};
```

a-1. European stock call 옵션 모형

단순 옵션 클래스의 옵션 평가 공식을 상속받아 이용한다. 다만 공식 중의 phi 값을 1로 설정하면 유럽식 콜 옵션 모형이 된다. 이 클래스로부터 유럽식 선물 콜 옵션, 지수 콜 옵션 및 환 콜 옵션 등을 응용하여 사용할 수 있게 된다.

```
class EuroStockCallOption : public  ClosedFormSimpleOption {
public:
        //constructor
          EuroStockCallOption(float t, float sigma, float S,
                  float r, float d, float EX)
           ClosedFormSimpleOption(t, sigma,S,r,d, EX) {};
         // member functions
          virtual void calculate_option_price()
           {   calculate_option_value(1); },
};
```

a-2. European stock 풋 옵션 모형

단순 옵션 클래스의 옵션 평가 공식을 상속받아 이용한다. 다만 공식 중의 phi 값을 -1로 설정하면 유럽식 풋 옵션 모형이 된다. 이 클래스로부터 유럽식 선물 풋 옵션, 지수 풋 옵션 및 환 풋 옵션 등을 응용하여 사용할 수 있게 된다.

```
class EuroStockPutOption : public  ClosedFormSimpleOption {
public:
        //constructor
        EuroStockPutOption(float t, float sigma, float S,
                  float r, float d, float EX)
        : ClosedFormSimpleOption(t, sigma,S,r,d, EX) {};
       // member functions
        virtual void calculate_option_price()
        {   calculate_option_value(-1); };
};
```

b. 이항옵션 모형

이항옵션의 경우는 상품구조가 복잡하여 단순옵션보다 훨씬 더 많은 계층을 갖게 된다. 이항 옵션 모형에 대한 계층구조는 <그림 2>와 <그림 3>에 나타나 있다. 본 연구에서는 특정 Path에 속하는 클래스만을 설명한다. 이항옵션 클래스에는 하위 계층에서 공통적으로 사용될 속성인 Barrier 수준(H)을 정의하고 나머지 옵션 특성들은 상위 클래스인 옵션 모형과 해석적 해 모형으로 부터 계승받는다. 또한 하위 계층에서 공통적으로 사용될 가치평가 공식들인 μ, λ, a, b, x, ,x_1, y, y_1, z를 위한 멤버 함수를 정의한다.

b-1. Cash 옵션 모형

이항옵션 모형의 다음 계층은 옵션의 조건 만족 시에 손익 지급이 미리 정해진 금액을 지급하는가 또는 기초증권의 가치를 지급하느냐에 따라서 Cash 옵션 모형과 Asset 옵션 모형으로 구분될 수 있다. Cash 옵션 모형에는 새로운 속성인 미리 정해진 금액인 X를 추가 정의해야 한다.

b-2. Path-dependent 옵션 모형

Cash 옵션 모형의 다음 계층은 옵션의 손익구조가 만기동안 기초증권의 가격 Path에 의존하는 가에 따라서 path-dependent 옵션 모형 과 path-independent 옵션 모형으로 구분된다.

b-3. Pay out at maturity 옵션 모형

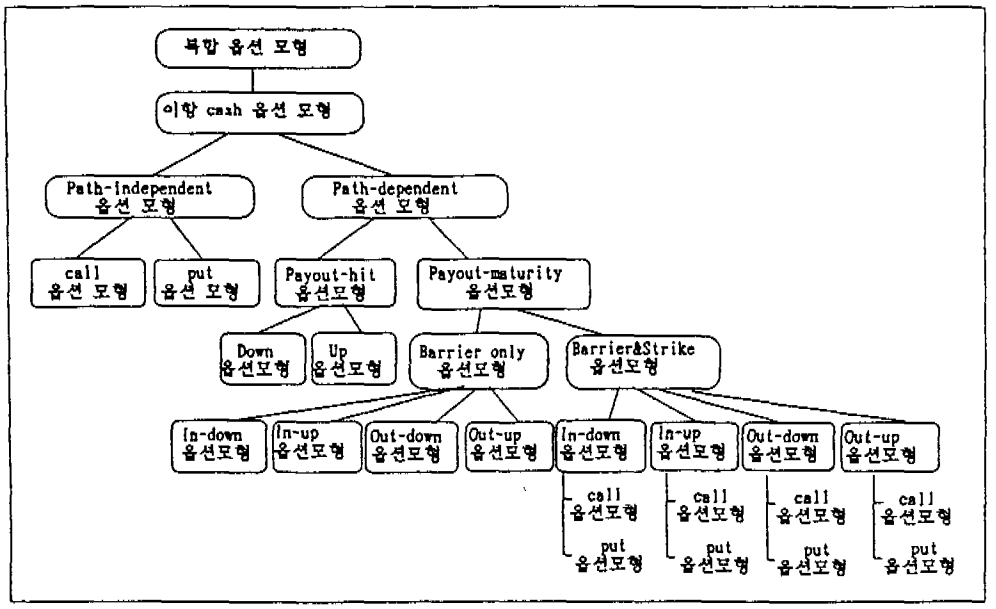
Path-dependent 옵션 모형의 다음 계층으로는 옵션의 조건이 만족되는 경우 손익지급이 만기중간에 이루어지느냐 또는 만기시점에서만 이루어지느냐에 따라 Payout at hit 옵션 모형과 payout at maturity 옵션 모형으로 구분된다 payout at maturity 옵션 모형에 속하는 옵션들은 II장에서 정의한 공식들중 1C, 2C, 3C, 4C가 공통적으로 이용되므로 payout at maturity 옵션 모형 클래스에 이들을 계산하는 멤버 함수를 정의한다.

b-4. Barrier only 옵션 모형

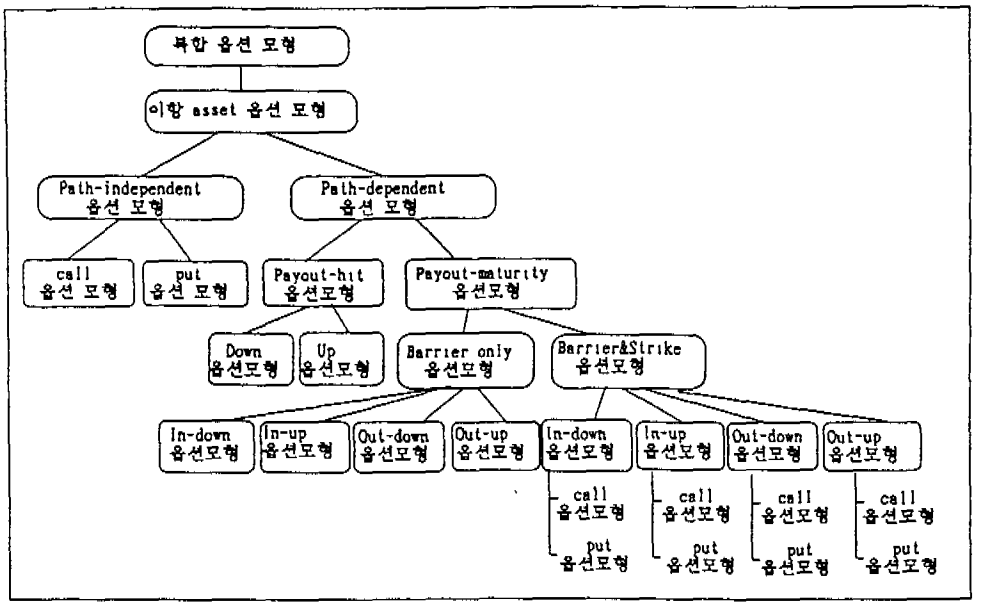
Pay out at maturity 옵션 모형은 손익 구조가 Barrier에만 의존하는가 또는 Barrier와 행사가격(strike price)에 모두 의존하는 가에 따라서 Barrier only 옵션 모형과 Barrier and strike 옵션 모형으로 구분된다 Barrier only 옵션 모형은 상위 계층인 옵션모형 클래스로부터 행사가격(strike price)을 계승받는다. Barrier only 옵션 모형에 속하는 모든 옵션은 공통적인 가격 결정 공식을 갖는다. 그러므로 이 공식을 Barrier only 옵션 모형 클래스의 멤버 함수로 정의한다 옵션 가치는 $-2C + \theta*4C$ 값을 갖는다.

b-5. In-Down 옵션 모형

옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하로 떨어진 적이 있는가 또는 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하를 계속 유지하는가에 따라 In, Out이 결정되고 또한 Barrier가 위쪽에 존재하느냐 또는 아래쪽에 존재하느냐에 따라서 Up과 Down 이 결정된다. In - Down 옵션 모형은 Barrier가 아래쪽에 존재하면서 손익 지급조건이 기초증권의 가치가 일정수준(H)이하로 떨어진 적이 있는가 하는 것이다. 즉 옵션의 만기동안에 기초증권의 가치가 일정수준(H)이하로 떨어진 적이 있으면 만기시점에 일정한 금액(X)을 지급하는 옵션이다. 이 옵션의 가치는 Barrier only 옵션 모형 클래스의 멤버 함수인 옵션 가치를 계승받아서 이용하게 된다. 단 In - Down 옵션 모형 경우는 인수 값이 $\theta = 1$, $\Phi = -1$, $\pi = 1$을 갖는다



<그림 2> 이항 Cash 옵션 모형의 클래스 구조

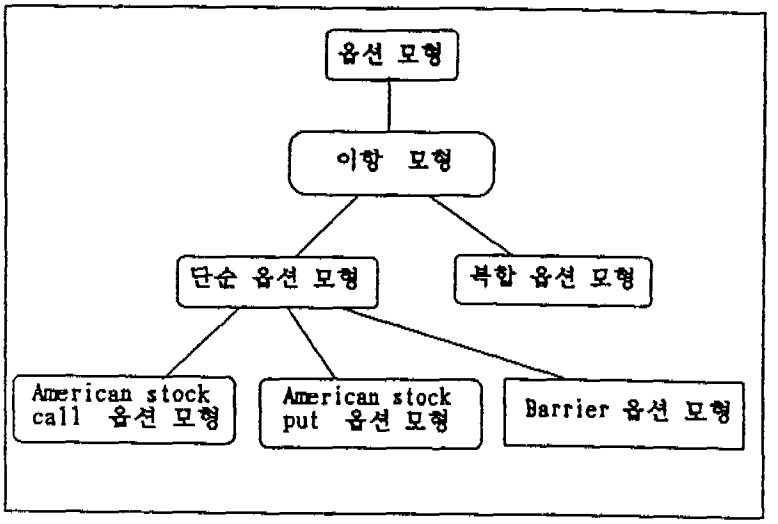


<그림 3> 이항 Asset 옵션 모형의 클래스 구조

3.3 이항 격자 모형(Binomial Lattice Model) 개발

3.3.1 클래스 구조

이항 모형은 상태 변수를 결정하는 방법에 따라서 단순 옵션과 복합옵션으로 구분된다. [그림 4]에 객체 클래스의 계층적 구조를 나타내었다.



<그림 4> 이항 옵션 모형의 클래스 구조

3.3.2 객체 속성 및 멤버 함수

[그림 4]의 클래스 구조하에서 각 클래스의 속성 및 멤버함수를 다음과 같이 설계하였다.

(1) 옵션 모형

이항 모형의 최상위 옵션 모형은 해석적 해를 갖는 옵션 모형의 최상위 클래스를 공유하게 된다.

(2) 이항 모형

이항 모형은 해석적 해가 존재하지 않는 옵션들에 대한 집합이므로 이들에 대한 해를 구하기 위해서는 해의 정확성을 결정하는 단위 기간의 크기(n) 를 결정해야 하며, 또한 단위 기간에 있어서의 기초 증권의 상승폭(u), 하락폭(d), 상승할 확률(p), 하락할 확률(1-p) 등을 구해야 한다. 이항 모형 클래스의 속성 및 멤버 함수는 다음과 같이 정의할 수 있다.

```
class BionomialModel : public OptionModel {
  public:
      // attributes
        float n, h,u,d,p;
     //constructor
        BionomialModel(float t, float sigma, float S,
                float r, float d, float EX, int n)
        : OptionModel(t, sigma,S,r, d, EX) {};
     // member functions
        virtual void cal_udp()
         {
                float  a;
                h = t/(float) n;
                u = exp(sigma*sqrt(h));
                d = 1.0/u;
                a = exp(r*h);
                p = (a-down)/(u-down), },
    };
```

a. 단순 옵션 모형

단순 옵션 모형은 경계 조건의 차이에 의해 미국식 콜 옵션, 미국식 풋 옵션, Barrier 옵션 등으로 나누어진다. 이러한 경계 조건은 하위 클래스에서 이용할 수 있도록 처리해

주고, 나머지 상태 변수 정의 방법 및 역대입 방법에 의한 가치 평가 방법이 동일하므로 이에 대한 처리는 단순 옵션 모형 클래스의 멤버 함수로 처리한다.

```
class BionomialSimpleOption : public BionomialModel{
 public:
        // attributes
           float OP;
        //constructor
           BionomialSimpleOption(float t, float sigma, float S,
                 float r, float d, float EX, int n)
           : BionomialModel(t, sigma,S,r, d, EX, n) {};
        // member functions
          virtual float pay_ftn(float s) { return 0;};
          virtual void calculate_option_price();
          virtual void display(ostream&);
};
void BionomialSimpleOption::calculate_tree()
{
          int i, j,
          float  c[370],payoff[370],n_payoff[370], s_price[370];
          calculate_udp();
          // Stock price at maturity

          for (i=0; i<=n; i++) {
               s_price[i] = S*pow(down,n-i)*pow(u,i);
               payoff[i] = pay_ftn(s_price[i]);
          }
          /* Backward computation */
          for (j=n-1; j >= 0; j--) {
                    for (i =0; i <= j; i++) {
                        n_payoff[i] = (p*payoff[i+1] +
                      (1-p)*payoff[i])*exp(-r*h);
                        s_price[i] = S*pow(down,n-i)*pow(u,i);
                        payoff[i] = pay_ftn(s_price[i]);
                     }
                    if (payoff[i] >= n_payoff[i])
                     c[i] = payoff[i];
                    else
                     c[i] = n_payoff[i];
                    for (i=0; i <= j; i++)
```

```
                payoff[i] = c[i];
            }
            OP = payoff[0];
}
```

a-1. American stock call 옵션 모형

옵션 가치 평가 방법에 이용되는 경계 조건(pay_ftn)이 달라지므로 이를 정의해주면 상위 클래스인 단순 옵션 모형의 동일 멤버 함수 pay_ftn 을 대체하게 된다. 이 클래스로부터 미국식 선물 콜 옵션, 지수 콜 옵션 및 환 콜 옵션 등을 응용하여 사용할 수 있게 된다.

```
class AmericanStockCallOption : public BionomialSimpleOption  {
public:
        //constructor
          AmericanStockCallOption(float t, float sigma,
                        float S,  float r, float d, float EX, int n)
          : BionomialSimpleOption(t, sigma,S,r,d, EX, n) {};
          // member functions
          virtual float pay_ftn(float s)
          {
             float  c;
             if (s >= EX)
                 c = s - EX;
             else
                 c = 0;
            return c;
          };
};
```

a-2. American stock put 옵션 모형

풋 옵션에 맞도록 상위 클래스인 단순 옵션 모형의 동일 멤버 함수 pay_ftn 을 대체하게 된다. 이 클래스로부터 미국식 선물 풋 옵션, 지수 풋 옵션 및 환 풋 옵션 등을 응용하여 사용할 수 있게 된다.

```
class AmericanStockPutOption : public BionomialSimpleOption  {
public:
        //constructor
          AmericanStockPutOption(float t, float sigma,
                  float S, float r, float d, float EX, int n)
```

```
        : BionomialSimpleOption(t, sigma,S,r,d, EX, n) {};
      // member functions
      virtual float pay_ftn(float s)
      {
        float  c;
        if (s >= EX)
            c = s - EX;
        else
            c = 0;
        return c;
      };
};
```

3.4 객체 지향에 의한 옵션 모형 수립 방법

옵션 모형 개발을 위해 필요한 주요 객체 클래스및 멤버 함수가 완성되고 나면, 사용자는 다양한 형태의 옵션 모형을 손쉽게 개발할 수 있게 된다. 또한 객체 지향 기법을 이용함으로써, 사용자가 변경하고자 하는 멤버함수를 쉽게 교체할 수 있으므로, 모형의 변경 및 확장이 용이하게 이루어지게 된다. 객체 지향에 의한 옵션 모형 수립 절차를 다음과 같이 정리할 수 있다.

(1) 객체의 생성 : 사용자는 각 클래스에 정의되어 있는 클래스의 생성자 함수를 이용하여 객체를 선언함으로써 구체적인 금융상품을 생성할 수 있다.

예를 들어, 사용자가 유럽식 주식 콜 옵션에 대한 가격을 계산하고자 하는 경우에는 EuroStockCallOption 클래스의 생성자를 이용하여 이 클래스의 새로운 사례(Instance) 객체를 생성하고, 관련되는 인자들을 선언해 준다.

```
EuroStockCallOption *NewOptionModel = new EuroStockCallOption(0.72, 0.2, 100, 0.1, 0, 90);
```

(2) 상속성 : 생성된 객체는 상위 클래스의 모든 관련 속성 및 멤버 함수를 이용할 수 있게 된다. 옵션 가격을 결정하는 멤버 함수를 호출하면 클래스간의 상호 작용에 의해 옵션 가격이 결정된다.

```
NewOptionModel->calculate_option_price();
```

(3) 함수의 중복 및 다형성 : 만일 사용자가 기 정의된 클래스에서 제공하는 멤버 함수를 변경하고 싶은 경우에는, 새로운 함수를 멤버 함수로 첨부하면 기존의 클래스에서 상속된 동일한 함수는 무효가 되고, 사용자가 정의한 새로운 멤버 함수로 대체하게 된다.

만일 사용자가 유럽식 콜 옵션에서 이용하는 옵션 가격 평가 함수를 대체하고 싶은 경우에는 다음과 같이 새로운 멤버 함수를 추가시키면 된다.

```
class NewOptionClass : public EuroStockCallOption  {
 public:
        //constructor
          NewOptionClass(float t, float sigma, float S,
                 float r, float d, float EX)
          : ClosedFormSimpleOption(t, sigma,S,r,d, EX) {};
        // member functions
        virtual void calculate_option_price()
        {     OP =  2*calculate_option_value(1) -
           calculate_option_value(-1); };
};

NewOptionClass* NewOptionModel   =  new NewOptionClass(0.72, 0.2,  100,  0.1, 0,
90);
NewOptionModel->calculate_option_price();
```

이 경우 EuroStockCallOption 클래스의 calculate_option_price() 멤버 함수는 무력화되고, 사용자가 정의한 동일한 이름의 멤버 함수가 적용하게 된다.

IV. 결 론

향후 금융시장이 발전하고 개방화가 가속화되면 옵션 및 선물, 스왑 등이 결합된 여러 복합금융상품들이 개발될 것이다. 이러한 복합금융상품들은 위험성이 매우 높기 때문에 효율적인 위험관리 시스템의 구축이 필수적이라 할 수 있다. 효율적인 위험관리 체계를 구축하기 위해서는 첫째로 파생금융상품 업무에 대한 조직차원의 표준과 절차가 명확히 수립되어야 한다. 즉 위험에 대한 허용 및 손실에 대한 한도설정, 보고의무사항 등에 대한 내부 규정이 수립되어야 한다. 둘째로, 각종 위험을 관리하고 성과를 측정하는데 필요한 모델 개발과 파생금융상품의 거래가 즉시 책임자에게 보고될 수 있는 거래시스템의 개발이다 셋째는 최고경영층의 파생금융상품에 대한 이해이다. 파생금융상품에 대한 업무가 전문성을 요구한다는 특성에 따라 소수 전문가들의 전유물이 된다면 파생금융상품에 대한 업무는 블랙박스가 되어 조직 차원에서 관리 감독이 어려워진다. 따라서 경영층은 개별 파생금융상품의 위험 및 전체 위험, 손익의 범위를 항상 파악하기 위해 파생금융상품에 대한 지식을 갖고 있어야 한다. 특히 자본시장의 개방을 앞두고 고도의 기술을 갖고 있는 외국투자가들에 대항하여 개방으로 인한 국부의 유출을 막기 위해서도 국내 투자가들의 과학적인 위험관리모형 및 시스템 개발이 시급히 요구된다고 할 수 있다.

이러한 관점에서 본 연구는 객체지향 기법을 이용하여 대표적인 파생금융상품인 옵션에

대한 가치평가 모형을 개발하고 C++언어를 이용하여 개발모형을 구현하였다. 잘 알려진 바와 같이 객체 지향기법은 인간위주의 시각에서 시스템 구축을 가능케 하며, 또한 모듈화, 재사용성, 확장성, 변경성이 용이하며 폴리몰피즘과 동시작업이 가능하게 되는 장점을 갖고 있다. 옵션의 가치평가 작업에는 클래스의 계층구조 및 상속개념, 모듈화의 개념이 많이 발생하기 때문에 이러한 객체지향기법의 이용은 전통적인 프로그래밍 기법보다 많은 장점을 갖게 된다.

첫째, 옵션은 기초증권이나 다른 파생금융상품을 기반으로 발행되기 때문에 각 금융상품간에 클래스의 계층적 구조가 형성되고 상위 클래스에서 하위 클래스로 특성이나 가격결정방법등의 상속(Inheritance)이 많이 발생한다. 이러한 특징은 프로그래밍의 효율성을 향상시킬 뿐만 이니라 모듈화로 인한 재사용성을 증가시킨다.

둘째, 옵션의 특성상 고객의 새로운 욕구가 발생할 때마다 새로운 파생상품이 개발되어야 하기 때문에 파생금융상품에 대한 시스템은 확장성이 뛰어 나야 한다. 객체지향 기법을 이용한 시스템 구축은 모듈화 및 캡슐화로 인하여 이러한 시스템의 확장성이 뛰어나게 된다.

본 연구에서 볼 수 있는 바와 같이 객체지향기법은 옵션의 가치평가 시스템 개발에 우수한 능력을 갖고 있다. 그러므로 향후 활발히 이루어질 파생금융상품에 대한 시스템구축에 객체 지향기법은 좋은 성과를 줄 것으로 기대된다.

참 고 문 헌

[1] Black, F., and M. Scholes. "The Pricing of Options and Corporate Liabilities." Journal of Political Economy, vol. 81, 1973, pp. 637-659.

[2] Brennan, M. J., and E. S. Schwartz. "Finite Difference Methods and Jump Processes Arising in the Pricing of Contingent Claims: A Synthesis." Journal of Financial and Quantitative Analysis, vol. 13, 1978, pp. 461-474.

[3] Cho, H. Y, and Lee, K Y, "An Extension of the Three Jump Process Model for Valuing contingent Claims," The Journal of Derivatives, Fall, 1995, pp. 102-108.

[4] Cho, H. Y, Kim, I. J, and Khun, S, W, "Duration of Default_Prone Bonds" The International Journal of Finance, Vol 8, No 2, 1996, pp 146-161.

[5] Cho, H. Y "An Algorithm for Pricing Geometric Average Options,"Forthcomming in The Journal of Financial Engineering, 1997.

[6] Cox, J.C., and S.A. Ross, and M. Rubinstein. Option Pricing: A Simplified Approach." Journal of Financial Economics, vol. 7, 1979, pp. 229-263.

[7] Cox, J, and M. Rubinstein, "Options Market," 1985, Prentice-Hall.

[8] Goldman, B., H. Sosin, M A. Gatto, "Path Dependent Options: Buy at the Low, Sell at the High," Journal of Finance, vol. 34, 1979, pp 1111-1127.

[9] Hull, J., and A. White, "Options Futures, and other Derivative Securities," Prentice-Hall, 1993.

[10] Lee, J. K, Suh, M. S., Oh, S. B., Kim. M. Y., Shin, J. C., and Song, Y. U.

"Development of UNIK for the Integration of Knowledge and Optimization Models," Technical Report N468-3509-7, KAIST, 1988.

[11] Morton, R "The Theory of Rational Option Pricing" Bell Journal of Economics and Management Science, vol. 4, 1973, pp. 141-183.

[12] Stulz, M.K. "Options on the Minimum or the Maximum of Two Risky Assets: Analysis and Applications." Journal of Financial Economics, vol. 10, 1982, pp. 161-185.

[13] Rubinstein, M "Unscrambling the Binary Code," RISK, October 1991.

[14] Stefik, M. and Bobrow, D.G, "Object Oriented Programming: Themes and Variations", The AI Magazine, 1985, pp

[15] Tumbull, S. "Pay-later option," Risk, April, 1992.

[16] Tumbull, S. M, and L.M. Wakeman, "A Quick Algorithm for Pricing European Average Options," Journal of Financial and Quantitative Analysis, 26, 1991, 377-378.