PROTRADER: An Expert System for Program Trading

by K.C. Chen, Theodore Brix Professor of Finance, School of Business, California State University, Fresno, CA 93740-0007
and
Ting-peng Liang, Assistant Professor of Accounting, Department of Accountancy, University of Illinois at Urbana- Chamapign, Chamapign, IL 61820

Abstract

Recently, program trading has allowed arbitrageurs to take advantage of the discrepancies in the futures market and the stock market. The key that enables program trading is computer technology. This article presents the design of PROTRADER - an expert system prototype for program trading implemented in M.1. In particular, a learning mechanism that allows the system to adapt to the changes in the market is presented.

Introduction

In the past several years the introduction of stock index futures and options has created a new way of making money for institutional investors in the stock market. This new game on Wall Street is known as "program trading." It allows arbitrageurs to take advantage of the disparities between the futures and stock markets and provides opportunities to make risk-free profits.

Traditionally, investors buy or sell stocks for several fundamental reasons, including the prosperity of economy and the outlook of a company's earnings. Today, however, an increasing number of program traders have substantially changed the face of the stock market. By funneling billions of dollars in and out of stocks in concentrated doses, program trading has caused wild swings in the market. For example, program trading has been blamed for triggering an 87-point drop in the Dow Jones Industrial Average on September 11, 1986 and a 508-point drop on October 19, 1987.

The key factor that enables program trading is computer technology. Because of the volatile nature of the futures and stock markets, only computers can provide the power required for keeping track of the rapid changes in both markets, usually several times a minute. Designing computer software that executes transactions based on a predetermined return on investment is not very difficult. These kinds of traditional systems, however, provide only limited support to program traders. In order to take full advantage of program trading, an intelligent system that can adapt itself to dynamic markets is essential. In other words, the system must be able to learn changes in the markets and plan its investment strategy accordingly.

In this article, we will present an expert system for program trading - PROTRADER (PROgram TRADER) - and the learning mechanism adopted by this system. The system is implemented in M.1 expert system shell. It monitors the differences between two markets, determines the optimum strategy for investment, executes transactions when appropriate, and enhances its knowledge by learning. A key issue that makes an expert system useful is its ability to adapt to a dynamic environment. In other words, the system must have learning capabilities. The learning mechanism to be described is based on a parameter adjustment approach that adjusts several critical parameters based on the market situation. Since program traders have only a small set of possible actions and most of the decision rules are highly dynamic, program trading is different from traditional expert system domains such as medical diagnosis.

Program Trading: The Problem Domain

There are two markets on which stocks are traded: stock market and futures market. On the stock market, investors buy or sell stocks; on the futures market, however, investors buy or sell stock index contracts. Currently only selected stock indexes are available in the futures market, such as the Standard & Poor's 500 index (S&P 500) based on 400 industrial firms, 40 utilities, 20 transportation firms, and 40 financial institutions, and the Major Market Index (MMI) based on 20 blue-chip stocks. In general, the price of a futures contract (or the index futures) is higher than that of its underlying stocks (or the index) because an arbitrageur, in order to sell a futures contract safely, must purchase the underlying stocks and hold them for delivery on the expiration. The difference between the index and the price of the futures contract is called "premium" or "spread". On the third Friday of the maturing month, the futures contract will expire and the futures index and the stock index will have to be the same.

The idea of program trading is simple. Theoretically, the premium is equal to the costs for holding the stocks, i.e., the interest expenses less the dividends received from the stocks. In the real world, however, different demand and supply forces in two different markets sometimes drive the index futures far above or far below their "theoretical" fair value. In this case, an arbitrageur can trade the same amount of stocks in both markets and converts the premium into profits when the futures contract expires. If the premium is higher than the fair value, program traders buy stocks that match or mimic the index, usually in a "basket" of $6 million to $12 million, and at the same time hedge those stocks by selling short the overpriced stock-index futures contracts. This transaction is usually called a *buy program*. If the premium is sufficiently lower than the fair value, then program traders sell stocks in the stock market and at the same time buy an equivalent amount of futures contracts. This is called a *sell program*. The following example illustrates how the premium can be converted into a risk- free return (more examples can be found in (2, 3, 4)).

(Example 1) On October 10, 1986, the S&P stock index and a futures contract on the index expiring in December were selling for 235.48 and 236.87, respectively, i.e., the premium was 1.39 points. Suppose an arbitrageur invested $11.77 million to execute a buy program and successfully unwind the program on expiration (December 19, 1986) by selling the purchased stocks and settling the futures contracts sold on October 10. We also assume that the investor received $123,600 dividends from the purchased stocks (equivalent to an annual dividend yield of 4.2%). Since the program trader had no control over the stock market, the actual S&P 500 index on the expiration date could be higher or lower than the purchased index. In either situation, however, the investor had the same return on investment. This is why we call it risk-free.

Situation 1: The stock market was bullish and the index expired at 249.73.

Gain from selling the stocks. (bought at 235.48; sold at 249.73)	$712,500
Loss from settling futures contracts. (sold at 236.87; settled at 249.73)	(643,100)
Dividends from stocks.	123,600
Total profit.	193,100
Rate of return on $11.77 million.	1.64%
Transaction costs	0.50%
Annualized net rate of return.	8.21%
Treasury bill yield.	6.00%
Bonus risk-free rate	2.21%

Situation 2: The stock market was bearish and the index expired at 230.24.

Loss from selling the stocks. (bought at 235.48; sold at 230.24)	$ (262,000)
Gain from settling futures contracts. (sold at 236.87; settled at 230.24)	331,500
Dividends from stocks.	123,600
Total profit.	193,100
Rate of return on $11.77 million.	1.64%
Transaction costs	0.50%
Annualized net rate of return.	8.21%
Treasury bill yield.	6.00%
Bonus risk-free rate	2.21%

The major decisions involved in program trading are when and how much to invest. Although program trading guarantees a risk-free return, the volatility in both the stock and the futures markets makes it difficult to optimize these decisions. For example, when the premium is higher than its fair value, there is always a chance that the premium could be even higher. Therefore, the decision maker must decide whether to invest immediately when the premium is profitable or to wait for a higher return. The major factor that affects this decision is the probability that the premium will be higher. This probability may be affected by various economic and political factors and usually changes every day. In other words, knowledge and judgment are important in assessing the probability and an expert system is crucial to improving the performance of investment.

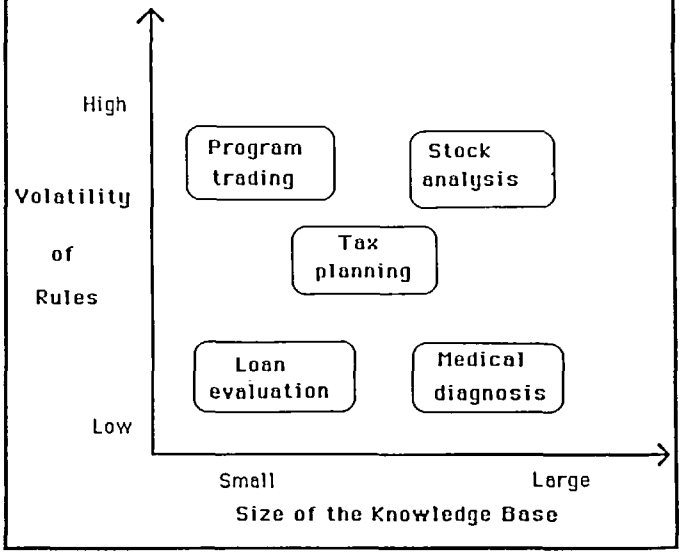
Design of PROTRADER

The return on investment of program trading is affected by two factors: the premium and the period an investor has to hold a buy or sell program. Since the premium varies from time to time, the major functions of PROTRADER include the following:

1. Monitor the premium in the market;
2. Determine the optimum investment strategy, including
   - When to execute a buy or sell program,
   - When to unwind a buy program, and
   - When to unwind a sell program.
3. Execute transactions when appropriate, and
4. Modify the knowledge base through a learning mechanism.

Although these functions indicate that PROTRADER falls into several generic categories of expert systems including monitoring and control (1,13), its design is different from many traditional applications. Figure 1 illustrates the differences among several typical domains in terms of the size of knowledge base and the volatility of decision rules.

Figure 1: Different Domains



In MYCIN, for example, there are many rules representing a substantial amount of possible actions in the problem space; each rule has an associated probability (9, 11, 14). Both the rules and the probability are reasonably stable. That is, they represent the best judgment of experts.

Designing PROTRADER, however, is facing a different situation.

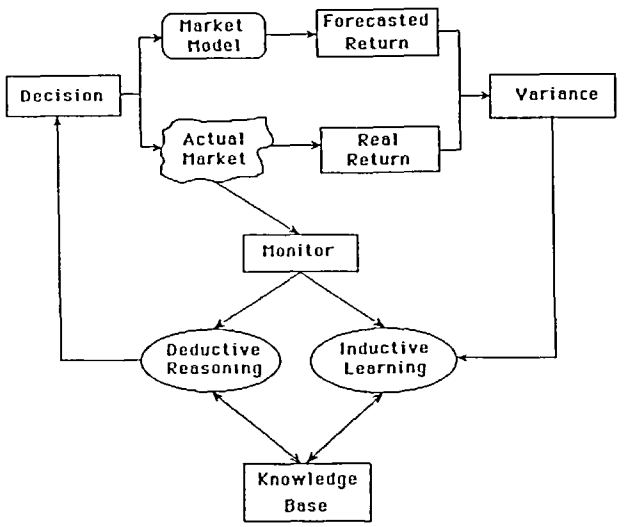
First, *its knowledge base includes only a small amount of decision rules*. Only two major indexes in the futures market are appropriate for program trading. For each of these indexes, there are five possible actions: execute a buy program, execute a sell program, unwind a previously executed buy program, unwind a previously executed sell program, and wait for better opportunities. Therefore, unlike MYCIN, PROTRADER needs to consider only a small set of possible actions.

Second, *the uncertainty associated with each rule is extremely dynamic*. Depending upon the market situation, the same rule may have different degrees of certainty at different time. For example, the probability that the premium will go higher than 1.39 point may be lower than 5% one day, but higher than 15% on another day. Different assessments on the probabilities may lead to different investment decisions. Therefore, rule probabilities are variables to be dynamically determined in PROTRADER. The following is a sample rule in the knowledge base (the capital letter 'N' represents a variable):

rule-51: if have money = yes and premium is positive
and return is profitable and confidence = N
then decision = execute buy program cf N.

With these two features, PROTRADER is composed of three main components: a knowledge base, a deductive reasoning mechanism, and an inductive learning mechanism. Figure 2 illustrates the relationships among these components. The monitor is a sensor that keeps track of the changes in the markets.

Figure 2: Architecture of the System



The knowledge base contains three categories of rules that determine when and how much to invest. First, it includes rules that determine the best strategy. In general, there are two strategies for program trading: one suggests the investors to hold a buy or sell program until expiration; whereas the other requires the investors to continuously monitor the market and to take actions based on a set of pre-determined rules. Because evidence indicates that the latter strategy usually outperforms the former, PROTRADER takes it as the primary strategy. In addition, the system also maintains a database that contains the resources on hand, including money, executed buy programs, and executed sell programs.

Second, it includes rules that convert a premium into return on investment. Although return and premium are highly related, decisions are made based on return. Because the conversion does not involve complex computation, the built-in arithmetic functions of M.1 are sufficient for this application.

Finally, it includes rules that provide suggestions and command transactions. Rule-101 illustrated in the following is an example

rule-101: if message-text (ID, LIST OF PARAMETERS)
= SUGGESTION and display (SUGGESTION)
then message (ID, LIST OF PARAMETERS).

These rules are then integrated by a capstone rule as follows:

rule-1: if resources is sought and market trend is
sought and return on investment is sought and printed
suggestions
then consultation is over.

Based on the rules in the knowledge base, the deductive reasoning mechanism allows the system to draw conclusions regarding the investment decision. Since PROTRADER is implemented in M.1, it takes advantage of the backward reasoning capabilities of the shell. The major reason for choosing a shell rather than developing a deductive reasoning mechanism in an AI language was the capability of quick prototyping.

The inductive learning mechanism of the system examines the variance between the real outcome and the forecasted outcome for each transaction and adjusts the probability associated with each decision rule. It plays a central role in the system. Because the stock market is highly volatile, human experts are involved in the learning process. More detail about the mechanisms will be discussed in the next section.

In the decision process, the system first examines the decision environment. It checks the available resources, examines important information in the real markets, such as the general outlook of the market, and builds a theoretical market model based on the rules in the knowledge base. Then, the monitor keeps track of the premiums, and the deductive reasoning mechanism works with the knowledge base to find opportunities for program trading. When a profitable opportunity is identified, the system commands a transaction. Since the market is extremely dynamic, the actual outcome of the transaction may be better or worse than the forecasted one. In either situation, the variance will activate the inductive learning mechanism to adjust the rules in the knowledge base.

Learning Mechanism in PROTRADER

Learning denotes changes in the system that are adaptive

in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time (12). It plays a key role in improving the performance of expert systems. Previous research in artificial intelligence has identified several learning strategies (5), such as:

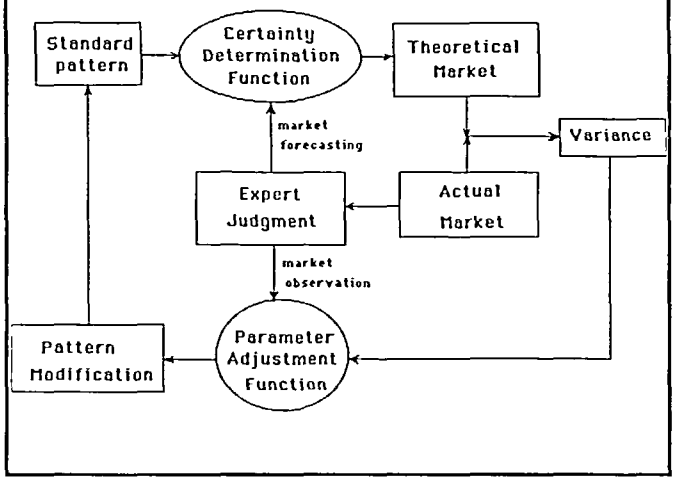
1. Rote learning: no inference or other transformation of knowledge is required;
2. Learning from instruction: the learner performs some inference, but the instructor presents the knowledge;
3. Learning by analogy: the learner acquires new facts or skills by transforming and argumenting existing knowledge that bears strong similarity to the desired new concept or skill that fits the new situation;
4. Learning from examples: the learner induces a general concept from a set of positive examples and counterexamples; and
5. Learning from observation and discovery: the learner acquires new concept in an unsupervised environment. It requires more inference than the previous four approaches.

Among these five approaches, two of them are appropriate for PROTRADER to adjust the probabilities associated with its decision rules. First, the system automatically identifies the market trend by analyzing historical data and then determines the probability that the premium will exceed a certain value. This is an application of learning by observation. Second, the system acquires the market trend from human experts and then determine the probability based on their judgment. This is an application of learning from instruction and is called "human-aided learning" in this article. Since the volatility of the stock market makes it almost impossible for a system to identify future market trends, human-aided learning is more feasible in this case.

The human-aided learning strategy implemented in PROTRADER is basically a parameter adjustment mechanism. The major parameter to be adjusted is the probability that the premium may exceed a particular value. This value determines the probability associated with a decision rule. In the example described in the first section, for instance, a premium of 1.39 results in a 2.21% bonus risk-free return. This is certainly a profitable opportunity. However, if the probability that the market will have a premium higher than 1.39 today is 60%, then the optimal strategy would be investing half of the money now and holding the rest for better opportunities, rather than committing all money right away.

How does the mechanism work? The major factor that affects the value of premium but cannot be controlled by the system is market trend. Therefore, human experts are responsible for forecasting and observing the market trend. By feeding the probability determination function with expert judgment, the system formulates a theoretical market model and provides suggestions accordingly. Since the theoretical market model only represents a forecast, frequent calibration is required. After the market is closing or when significant variance between the theoretical market model and the actual market is found, human experts are asked to provide their observation on the market. This information is then used to modify the standard patterns. Therefore, the major components in the mechanism are the formulation of the theoretical market model and pattern modification. Figure 3 illustrates this co-operation.

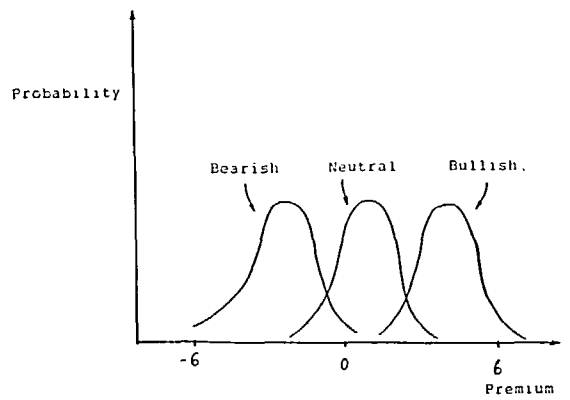
Figure 3: Human - aided Learning in PROTRADER



Formulation of the Theoretical Model

In order to determine the appropriate probabilities, the system maintains three standard patterns: bull market, neutral market, and bear market. Each pattern represents a probability distribution of premium in the corresponding market situation. They are determined every day based on the transaction data of the previous day. Figure 4 shows examples of these patterns.

Figure 4: Examples of Standard Patterns



Every day the experts forecast whether today's market will be bullish, neutral, or bearish. An uncertain judg-

ment such as 30% bullish and 50% neutral is allowed. Based on this information and the standard patterns, the system creates the market model by computing the probability that the premium will be higher than a certain value. This probability determines the probability associated with a decision rule.

The major issue in creating the theoretical model is how to convert standard patterns into a theoretical market model; or in other words, how a statement that "the market will be 50% bullish and 30% neutral" can be represented by a probability distribution of premiums. In designing PROTRADER, we use an approach based on a normality assumption on market patterns. Since the daily volume of transactions is very large, normal distribution is a reasonable assumption in this application.

By assuming that the premium distribution for each of the three market patterns is normal, we can represent each pattern by its mean and variance. The mean and variance of a theoretical market model can be calculated by the following two equations:

$$\text{Mean} = \frac{\sum_{i=1}^{3} W_i M_i}{\sum_{i=1}^{3} W_i} \quad (1)$$

$$\text{Variance} = \frac{\sum_{i=1}^{3} W_i V_i}{\sum_{i=1}^{3} W_i} \quad (2)$$

Where: W_i = the probability that the market will be an i type of market,

M_i = the mean of the premium distribution in the standard market pattern i,

V_i = the variance of the premium distribution in the standard market pattern i,

i = 1, 2 and 3 denote the bull, neutral, and bear market, respectively.

The following is a numerical example illustrating how the system formulates the theoretical market model.

(Example 2) Suppose the expert judgment on the market trend on April 1, 1987 is "50% bullish and 30% neutral" and the system has the following standard patterns:

Patterns	Mean	Variance
Bull	3.5	1.0
Neutral	0.5	0.09
Bear	-3.5	1.0

Following equations (1) and (2), the mean and variance of the premium distribution for the theoretical market model are:

$$\text{Mean} = \frac{0.5\,(3.5) + 0.3\,(0.5)}{0.5 + 0.3} = 2.375$$

$$\text{Variance} = \frac{0.5\,(1.0) + 0.3\,(0.09)}{0.5 + 0.3} = 0.65875$$

With these mean and variance, when the system detects a 2.3 points profitable premium, it will suggest the manager "to invest half of the money to execute a buy program and hold another half for other opportunities because the probability of having higher premium is higher than 50%."

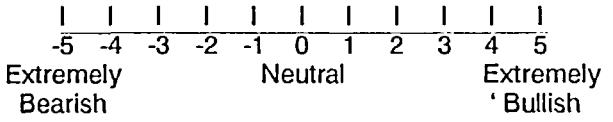
In summary, given the assumption of normal distribution, this approach has successfully simplified the calculation involved in the formulation of the theoretical market model and has made it possible to implement a prototype in M.1.

Pattern Modification

Standard patterns are the basis on which a market model can be developed. Therefore, it is very important to ensure that they are accurate. In the ideal case, the mean and variance recorded from the actual market are equal to those calculated from the theoretical model. In the real world, however, this rarely happens. After the market is closing or when significant variance between the theoretical and the actual market is found, human experts must be asked to provide the system with their observations on the actual market. This information enables the system to adapt itself to the changes in the market.

The method for calibrating standard patterns is similar to the approach previously described. The system first asks human experts to present their observation on a 5 to -5 scale, as follows:

What do you think would best describe today's market until now? Please indicate your observation on the following scale:



The value provided by human experts reflects a posterior observation which is likely to be more accurate than the prior forecast used to formulate the theoretical model. By modifying equations (1) and (2), the mean and variance of the modified standard pattern can be calculated. Since the system has only one observation, some of the patterns must remain unchanged in the calibration process.

In selecting the pattern to be modified, the general rule is that the neutral case remains unchanged unless the observation happens to be neutral (i.e., value = 0). In other words, if the value is higher than zero, then the system will modify the pattern of bull market. If the value is less than zero, then the system will modify the pattern of bear market. Equations (3) and (4) are used to adjust the means and variances of the bull and bear market patterns. If the value

is equal to zero, then the system uses equations (5) and (6) to reassign the neutral pattern.

$$M_i = \frac{5M_a - (5 - c)M_o}{c} \quad (3)$$

$$V_i = \frac{5V_a - (5 - c)V_o}{c} \quad (4)$$

$$M_o = M_a \quad (5)$$

$$V_o = V_a \quad (6)$$

Where:

- M_i = mean of standard pattern i
- V_i = variance of standard pattern i
- M_a = mean recorded from the actual market
- V_a = variance recorded from the actual market
- M_0 = mean of the neutral market
- V_0 = variance of the neutral market
- b = value provided by human experts; -55
- i = bull, if $b > 0$
  = bear, if $b < 0$.
- c = b, if $b > 0$;
  = -b, if $b < 0$.

(Example 3) This example continues the analysis shown in example 2. If at the closing of the market the system has recorded a mean of 2.1 and a variance of 0.7 and the expert observation on the market is 2 which means "more or less bullish", then following equations (3) and (4), the new pattern for the bull market can be created. Next time the system is used this pattern will replace the old standard for generating the theoretical market model.

$$\text{Mean} = \frac{5(2.1) - 3(0.5)}{2} = 4.50$$

$$\text{Variance} = \frac{5(0.7) - 3(0.09)}{2} = 1.615$$

Concluding Remarks

The rapid progress of computer technology has enabled program trading which allows arbitrageurs to take advantage of the disparities in the futures market and the stock market to make risk-free profits. Because of the volatile nature of both markets, the probability associated with each decision rule needs to be dynamically determined every time the system is used. This makes the development of an expert system for program trading different from traditional applications such as MYCIN.

In this article, we have presented a prototype system for program trading and the issues involved in designing this system. In particular, we have described a learning mechanism that integrates human observations and the knowledge of the system to improve the performance of the system. Although the work has been focused on program trading, the techniques described here can be generalised to any domain of high degree of volatility.

References

(1) Basden, A. (1983), "On the Application of Expert Systems," *International Journal of Man-Machine Studies*, 19, pp. 461-477.

(2) Business Week (1985), A New Breed of Investor is Whipsawing Wall Street, September 23, pp. 84-86.

(3) Business Week (1986), Those Big Swings on Wall Street, April 7, pp. 32-36.

(4) Business Week (1986), How Chicago Zaps Wall Street, September 29, pp. 32-36.

(5) Carbonell, J.G., Michalski, R.S. and Mitchell, T.M. (1983), "An Overview of Machine Learning," in Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds), *Machine Learning: An Artificial Intelligence Approach*, Vol. I, Los Altos, CA: Morgan Kaufmann Publishers, Inc., pp. 1-24.

(6) Coombs, M. and Alty, J. (1984), "Expert Systems: An Alternative Paradigm," *International Journal of Man-Machine Studies*, 20, pp. 21-43.

(7) Finnerty, J.E. and Park, H.Y. (1987), "Empirical Evidence on Stock Index Arbitrage: The Case of Program Trading," BEBR Faculty Working Paper 1321, College of Commerce and Business Administration, University of Illinois at Urbana-Chamapign.

(8) Forsyth, R. (1984), "Fuzzy Reasoning Systems," in Forsyth, R. (ed.), *Expert Systems: Principles and Case Studies*, London, U.K.: Chapman and Hall, pp. 51-62.

(9) Hayes-Roth, F., Waterman, D.A., and Lenat, D.B. (1983), *Building Expert Systems*, Reading, MA: Addison-Wesley Publishing Co.

(10) Schmucker, K.J. (1984), *Fuzzy Sets, Natural Language Computations, and Risk Analysis*, Rockville, MD: Computer Science Press.

(11) Shortliffe, E.H. (1976), *Computer-based Medical Consultations: MYCIN*, Amsterdam, The Netherland: Elsevier Scientific Publishing Co.

(12) Simon, H.A. (1983), "Why should Machines Learn," in Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol. I, Los Altos, CA: Morgan Kaufmann Publishers, Inc., pp. 25-37.

(13) Stefik, M., et al (1982), "The Organization of Expert Systems: A Tutorial," *Artificial Intelligence*, 18:2, pp. 135-174.

(14) Waterman, D.A. (1986), *A Guide to Expert Systems*, Reading, MA: Addison-Wesley Publishing Co.

(15) Zualkernan, I., Tsai, W.T. and Volovik, D. (1986), "Expert Systems and Software Engineering: Ready for Marriage," *IEEE Expert*, 1:4, pp. 24-31.

(16) Zadeh, L.A. (1973), "Outline of a New Approach to the Analysis of Complex Systems and Design Processes," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 3, pp. 28-45.