Abstract

With the advent of options and futures on financial instruments, investors have the opportunity to form complex strategies that meet their investment objectives. However, this opportunity gives rise to a difficult task for finding a desired strategy among a large amount of investment strategies. This paper describes an intelligent decision-support system for generating option-based investment strategies by using the notion of constraint satisfaction, which is widely applied to search problems. In this system, constraint satisfaction plays a role of navigation for automatically creating complex strategies through abstract matching with investors' profiles. Here, abstract matching can be viewed as a search method for producing qualitatively reasonable strategies, that describe a set of options to buy or sell without numerical information. Because this technique can be used as preprocessing to quantitative analysis such as linear programming in order to obtain an optimal solution, the proposed system provides a bridge for smooth transition between qualitative and quantitative analysis.

1 Introduction

Recent attempts to designing expert systems have focused on highly judgemental decision problems in business and financial domain. These attempts lay great emphasis on the inadequacy of the current expert system technologies in order to solve the problems. Hart et al., and Apte and Hong provided examples of exploiting new directions for financial application. Hart et al. introduced causal models in which they used qualitative information for designing a new decision support system[5]. Apte and Hong provided a mechanism for qualitative interpretation of arithmetic relations on the financial domain[1]. Following these attempts, the present paper focuses on these approaches to business and finance in applying AI methods into novel decision-support systems. Especially, we shall use the constraint-based approach to explore a financial decision support system in the area of option trading. This is based upon many recent advances in investment theory and practice so as to make a wide variety of investment strategies. That is, with the advent of options and futures on financial instruments, investors have the opportunity to form complex strategies to meet their own investment objectives. However, this opportunity gives birth to a difficult task for finding a desired strategy among a large amount of investment strategies.

In this paper, we propose an intelligent decision-support system for generating option-based investment strategies using constraint programming, which is one of the constraint-based methods[6]. In option theory, options can be arbitrary created by combining options once created in order to implement a variety of financial payoffs. This property indicates that the application of option theory gives rise to a new field, financial engineering, analogous to genetic engineering that allows biochemical constituents to be combined to replicate various biological entities[2]. Therefore, option is an interesting domain for applying a new AI technology to financial engineering problems as well as planning, design, scheduling and configuration tasks[10][8].

The proposed system is based on constraint satisfaction, that are widely applied to search problems. In this system, constraint satisfaction is used within two stages: qualitative analysis and quantitative analysis where qualitative analysis is to obtain possible qualitative solutions (or intuitive solutions), and quantitative analysis is to find an optimal solution using mathematical programming techniques such as Simplex Method in linear programming. This separation provides the following useful features:

- The separation is viewed as an abstract formulation for reducing search complexity, because qualitative analysis focuses on qualitative distinctions in solution space.
- The produced solutions in qualitative analysis are used as a set of constraints in quantitative analysis. This means that qualitative analysis plays a role of simplifying non-linear problems to linear problems that can be solved in quantitative analysis.
- The separation contributes to a bridge for smooth transition from qualitative to quantitative analysis in the decision-support system.

The system generates option strategies as shown in Figure 1. According to changes of investor's profiles
since qualitative analysis is more novel and promising paper, we will mainly focus on qualitative analysis, about market movement, the system repeats the cycle: qualitative analysis and quantitative analysis. In this paper, we will mainly focus on qualitative analysis, since qualitative analysis is more novel and promising than quantitative analysis.

2 Problem Description

We start by giving a brief introduction of option-based investment strategies. An option is the right to buy or sell a given amount of a security (e.g. stock) at a given price (called strike price) until a specific date. Here, the right to buy is called call option, and the right to sell put option. For example, ABC/JAN/50 call option gives the holder of the option the right to buy 100 shares of ABC stock at $50 a share until the January expiration.

The interesting feature of option-based investment is that we can tailor a number of option strategies by combining four primitive options (i.e. BuyCall, SellCall, BuyPut and SellPut). For example, the combination of SellCall and SellPut constructs a novel strategy in which we can get profit if the stock price does not change. However, this feature forces investors to find a desired strategy among option combinations. Suppose that the number of available options are $4^m$ where $m$ and $n$ indicate the number of striking prices and that of expiration dates. Since any strategies can be constructed from the list of options that are available for trading, we have $2^{4^m}$ option strategies. A large amount of option strategies give us the following questions: Which options should we select? How do we combine them? How do we find reasonable option combinations efficiently? The problem in generating option strategies is how to provide the way to find a desired strategy efficiently in order to meet investment objectives.

This problem involves two technical difficulties. The first is that most investors' profiles about market movement are imprecise. Since option theory consists of continuous variables that have probability distributions, it is impossible for an investor to give precise values of variables. This suggests a qualitative treatment for dealing with his profile. The treatment describes the profile in an abstract way. Suppose that the investor wants to restrict the loss of a stock option in decreasing stock price. The investor should represent his profile as the following qualitative statement:

The payoff is negative and constant when the stock price drops.

Obviously, the qualitative treatment leads to a heuristic approach, which is characterized by "if-then" rule. For example, the heuristic approach provides the rule "If the investor's attitude is bullish, the recommended strategy is BuyCall". However, this approach needs to pre-enumerate option strategies. Although it seems to be sufficient to prepare a large set of strategies, the investor must change his strategy and create a new strategy according to changes of his expectation about market movement. This second difficulty implies that a principled (not heuristic) approach is needed to generate option strategies.

3 The Constraint-based Approach

The role of constraint satisfaction in our approach is generating qualitatively reasonable strategies from a given investor's profile. Here, a qualitatively reasonable strategy does not contain quantitative information. In other words, this strategy can be specified by a set of partial ordering relations among variables. The investor's profile is also given as a qualitative description. This description can be characterized as a partially specified payoff function, which is a collection of qualitative states about the profile. Suppose that the profile is described as the following statements:

If the stock price is stable, get profit.
If the stock price is volatile, restrict loss.

The above statements can be translated to a qualitative description as shown below.

If the stock price is close to the stock price purchased, the payoff is positive.
If the stock price is close to 0 or 0, the payoff is negative and constant.

From the above qualitative description about the investor's profile, the following qualitatively reasonable strategy is generated:

Buy two call options whose strike prices are different ($K_1, K_3$), and sell two call options whose strike prices are the same ($K_2$) where $K_1 \leq K_2 \leq K_3$.

The distinct feature of this approach is to generate qualitatively reasonable strategies through constraint satisfaction. Unlike a heuristic approach, the approach uses qualitative calculus, that is based on qualitative version of continuity and mean value theorem. For example, the following simple rule is incorporated.
over the interval between the stock price \( X \) and \( Y \).

The qualitative value \( \text{qual} \) of the function \( f \) is defined as

\[
\text{qual} = + \quad \text{if the value of } f \text{ is greater than zero,}
\]

\[
\text{qual} = 0 \quad \text{if the value of } f \text{ is equal to zero,}
\]

\[
\text{qual} = - \quad \text{if the value of } f \text{ is less than zero.}
\]

The qualitative direction \( \text{qdir} \) of \( f \) is defined as

\[
\text{qdir} = \text{inc} \quad \text{if the derivative of } f \text{ is greater than zero,}
\]

\[
\text{qdir} = \text{std} \quad \text{if the derivative of } f \text{ is equal to zero,}
\]

\[
\text{qdir} = \text{dec} \quad \text{if the derivative of } f \text{ is less than zero.}
\]

\( QS(f) \) denotes the qualitative states \( < \text{qual}, \text{qdir} > \) of the function \( f \).

Using the above notations, a typical hedging pattern is described as follows:

\[
QS(PO(0)) = < - , \text{std} > \quad (1)
\]

\[
QS(PO(\infty)) = < + , \text{inc} > \quad (2)
\]

Clearly, the above specification about the payoff function is not complete; there exists unspecified portion of the payoff function. The generation procedure starts with such incomplete payoff function.

To create possible completed payoff functions, we use two types of constraints: universal (or domain-independent) constraints and domain-specific constraints. A universal constraint can be seen in mathematical theory, while a domain-specific constraint is derived from option theory.

Each constraint plays a different role. A universal constraint works as a generator, which makes the payoff function complete. To the contrary, a domain-specific constraint is used as a tester. It eliminates irrelevant payoff functions generated by using the universal constraints. Finally, the procedure synthesizes a combination of primitive options, which qualitatively matches the complete payoff function.

### 4.1 Interpolating unspecified portion of a payoff function

This section shows how to make a given payoff function complete by using a qualitative rule derived from the property of a continuous function. This rule corresponds to the simplified version of qualitative simulation[7]. The rule consists of two types of state transitions. These are From Point To Interval Rule and From Interval To Point Rule.

By using the transition rule, interpolating unspecified states has non-deterministic computation. This is due to qualitativness of the rule: the rule always does not infer the qualitative state of the payoff function. Therefore, there exists a number of payoff functions that satisfy only the continuity condition.

Now we show a simple example. Suppose that the payoff function is specified as (1) and (2). One of the completed payoff functions is produced as follows:

\[
QS(PO(0)) = < - , \text{std} > \quad (3)
\]

\[
QS(PO(0, BE)) = < - , \text{inc} > \quad (4)
\]

\[
QS(PO(BE)) = < 0 , \text{inc} > \quad (5)
\]

\[
QS(PO(\infty)) = < + , \text{inc} > \quad (2)
\]
where the symbol "BE" represents a break even point.

4.2 Constraints that eliminate irrelevant payoff functions

Unlike the transition rule presented in the previous section, this section shows constraints specific to option theory. These constraints play a role of eliminating irrelevant payoff functions.

We incorporate the following constraints:

1. Payoff functions that are steady in any interval does not exist.
2. Payoff functions that are always positive (or negative) does not exist.
3. Payoff functions that are positive (or negative) and increase (or decrease) at zero does not exist.

The first and second constraints are straightforwardly derived from option theory, while the third one indicates a rare case in practical option trading.

4.3 Synthesizing option strategies

Before synthesizing option strategies, we show the qualitative definition of an option strategy. A payoff function with respect to the option strategy \( s_t \) is defined as follows:

\[
PO^*(st, X) \quad \text{denotes the qualitative value of the payoff with respect to the strategy } st \text{ at the stock price } X.
\]

\[
PO^*(s_t, X, Y) \quad \text{denotes the qualitative value of the payoff with respect to the strategy } st \text{ over the interval between the stock price } X \text{ and } Y.
\]

An option strategy \( st \) is a qualitatively reasonable strategy for a payoff function \( PO \) if for all \( s_t \) and \( s_{t+1} \) such that \( 0 \leq s_t \leq s_{t+1} \), the condition

\[
QS(PO(s_t)) = QS(PO(s_{t+1}))
\]

is satisfied.

For example, a BuyCall strategy is a qualitatively reasonable strategy for the payoff function (3) – (7).

We now present the option synthesis procedure:

Let \( S = \{ \text{BuyCall, SellCall, BuyPut, SellPut} \} \). Given a completed payoff function \( PO \), qualitative reasonable strategies for \( PO \) are found by the following steps:

1. Generate a subset of \( S \).
2. Test whether the subset is qualitatively reasonable strategy or not.

5 System Architecture

The architecture of the decision-support system is shown in Figure 3. The inference part of the system is implemented in Quintus-Prolog on SUN-4, and a constraint logic programming language, Triton we developed[9]. Triton solves linear equations and inequations within constraint logic programming framework[6]. Because Triton provides a unified framework for handling both qualitative and quantitative information, the linkage between qualitative and quantitative analysis is straightforward.

The system provides spreadsheet interface, which is realized within the notion of incremental query[4]. This facility suggests a novel approach to solving optimum problems. In conventional approaches, the most simple way for solving the problems is to directly exploit libraries that implement Simplex method. Our approach is different; we use Triton for interactively finding reasonable solutions by using incremental query.

The knowledge base, which contains textbook strategies and associated default constraints, is mainly used for instructing an elementary course of option theory. For expert investors, the system produces all the possible strategies, that satisfy their profiles, by using constraint satisfaction mechanism. This facility indicates the advantage of the present approach in that expert investors empirically create their own strategies and therefore they may not cover all the option strategies.

6 Execution Example

An execution example of the system is shown in Figure 4. In this figure, an investor enters his own strategy in tabular form. The tabulation is used for specifying the value and the derivative of the payoff function. Here, the value is either +, 0, −, and
Enter your strategy:

val(0) = -1,
dir(0) = std.
val(inf) = +1,
dir(inf) = std.

{ The above strategy means that if the stock price is down, the result is the limited loss, and if the stock price is up, the result is the limited profit. }

There exists 1 qualitative pattern about the payoff function by combining 2 primitive options.

You can enter the following commands:

show, simple, complex, quit: complex.

{ Require complex patterns. }

There exists 4 qualitative patterns about the payoff function by combining 4 primitive options.

You can enter the following commands:

show, simple, complex, quit: show.

{ Show the patterns. }

Choose one of the patterns: 1.

There exists 6 qualitatively reasonable strategies.

Show or choose one of the strategy: show.

{ Show the strategy. }

1. Buy a call with the strike price $K_1$, sell a put with the strike price $K_2$, sell a call with the strike price $K_3$ and sell a call with the strike price $K_4$ where $K_1 \leq K_2 \leq K_3 \leq K_4$.

2. Buy a call with the strike price $K_1$,

Figure 4: Execution Example. (Underlines mean user's input. Comments in italics are not part of the actual interaction.)

<table>
<thead>
<tr>
<th>Program-1</th>
<th>Program-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>250</td>
<td>252</td>
</tr>
<tr>
<td>5133</td>
<td>67</td>
</tr>
<tr>
<td>98184</td>
<td>33</td>
</tr>
<tr>
<td>1885550</td>
<td>1885550</td>
</tr>
</tbody>
</table>

Table 1: Performance (in msec.)

7 Performance

The performance of the qualitative reasoning system is investigated by two inference parts: completing payoff functions and generating option combinations.

The efficiency of completing a given payoff function depends on user-supplied information. The worst case is to enter no information of the payoff function. In this case, computational complexity is exponential. However, such case can not be seen in realistic application.

The computation cost of qualitative reasoning is very low, because of the simplicity of qualitative calculus. For example, a number of payoff functions (even greater than 50 solutions) has been generated about one second.

Table 1 shows the performance of two programs for generating option combinations. The first program is based on generate-and-test style. The second program uses a qualitative version of option theory for efficiently finding desired combinations.

These programs were measured using Quintus-Prolog on SUN-4. The first one is simple, but exponential time algorithm. Therefore, this can not be applied to real financial problems. On the other hand, the second indicates that run time is proportional to the number of option combinations. This number corresponds to the computational complexity. The program just collects reasonable combinations and never search irrelevant candidates. This capability implies that completeness and consistency in searching solutions can be obtained without loss of efficiency by using a constraint-oriented approach based on qualitative reasoning.
8 Connection to Quantitative Analysis

The method based on qualitative reasoning can not answer the question:

How many options should we take?

Moreover, there exists a number of payoff functions varying quantitatively according to striking prices. Quantitative analysis is used for solving these problems.

As for quantitative analysis, we adopt linear programming techniques in order to get the optional solution. In option theory, option strategies are characterized by a set of position variables. For example, the position variable “Delta” indicates the expected rate of return. The variable “Premium” directly specifies the payment for buying options. These position variables reflect accurate investor’s attitude. Thus, the investor wants to constrain the values of the variables as intervals. Constraining the range of the position variables is specified as follows:

\[ \text{NetDelta} \in [\text{LowerDelta}, \text{UpperDelta}] \]

where \( \text{NetDelta} = \sum \text{Delta} \cdot \text{N} \), and \( \text{N} \) indicates the lot number of share for each option. So \( \text{N} > 0 \).

The objective function is of the form:

\[ \text{minimize} \left( \sum \text{Premium} \cdot \text{N} \right) \]

After solving this objective function, \( \text{N} \) is bound to real number.

After generating qualitative position in Figure 4, the problem is formulated as as follows:

- **Constraints**

\[
\begin{align*}
K_i & \in \{400, 420, 440, 460\} \\
K_i & \leq K_{i+1} \\
\text{NetDelta} & = D(\text{buy\_call}, K_i) \cdot N_i + \\
& D(\text{sell\_put}, K_2) \cdot N_2 + \\
& D(\text{sell\_call}, K_3) \cdot N_3 + \\
& D(\text{sell\_call}, K_4) \cdot N_4 \\
\text{NetDelta} & \in [\text{LowerDelta}, \text{UpperDelta}] \\
\end{align*}
\]

- **Objective function**

\[ \text{minimize} \left( P(\text{buy\_call}, K_1) \cdot N_1 + \\
P(\text{sell\_put}, K_2) \cdot N_2 + \\
P(\text{sell\_call}, K_3) \cdot N_3 + \\
P(\text{sell\_call}, K_4) \cdot N_4 \right) \]

where functions \( D(st, K_i) \) and \( P(st, K_i) \) return the delta value and the premium respectively for the strategy \( st \) with the strike price \( K_i \). The above formulation is non-linear model. However, the formulation is reduced to a linear programming problem, because the strike prices can be non-deterministically selected among the finite domain by using the computation mechanism in logic programming.

9 Concluding Remarks

We have described a constraint-based framework for generating option-based investment strategies. This framework provides both qualitative and quantitative analysis. The distinct feature of the framework is to introduce the principle technique using qualitative reasoning. Because qualitative analysis performs as preprocessing to quantitative analysis, the proposed framework provides a bridge between qualitative and quantitative analysis in the decision-support system.

The system has been developed within a three-year project (launched in 1989) in cooperation with a bank. The target of the project is currency option. At present, the system is not delivered yet to traders for the purpose of field test. This year, 1991, performance evaluation of the system will be reported.

References


