

Kiểm thử chức năng

Sinh viên : Nguyễn Tiến Hoàng - 21020123

1 Mô tả bài toán

Bài toán được em triển khai trong báo cáo lần này mang chủ đề Sử dụng Giải thuật Di truyền (Genetic Algorithms) trong việc Tối ưu hóa dự đoán giá cổ phiếu trong Thị trường chứng khoán .

1.1 Tổng quan

Chủ thể trong bài toán cố gắng dự đoán giá sắp tới của cổ phiếu sao cho gần với giá thực tế nhất , thông qua việc "hỏi những hàng xóm xung quanh" và đưa ra quyết định mua hay bán để tối ưu hóa lợi nhuận nhất có thể .

Ta xem xét giá cổ phiếu ban đầu là $P = 1$ và tăng hay giảm bao nhiêu chỉ phụ thuộc vào số lượng người mua và người bán , được tính theo công thức của thị trường chứng khoán :

$$bt = \frac{\text{buyers agents} - \text{sellers agents}}{\text{Total amount of agents}}$$

Formula 2.: bt variable

$$1 + \frac{e^{bt} - e^{-bt}}{e^{bt} + e^{-bt}}$$

Formula 3.: Correction coeficient

Chủ thể chỉ có thể hỏi một vài người , và những người đó đã đi hỏi những người khác và có những câu trả lời , tạo thành một cộng đồng nhỏ trong quần thể bài toán .

Từ lượng thông tin đã có từ cộng đồng nhỏ , có thể dự đoán được giá của cổ phiếu .

Ví dụ : Quần thể bài toán xem xét với 40 người , 21 người mua và 19 người bán , từ công thức trên ta có thể tính được Giá thực tế của cổ phiếu sẽ tăng lên 1.05

Trong cộng đồng nhỏ của chủ thể có 10 người , 3 người mua và 7 người bán , Giá dự đoán của cổ phiếu sẽ giảm xuống 0.62 , lệch đến 0.43 so với Giá thực tế .

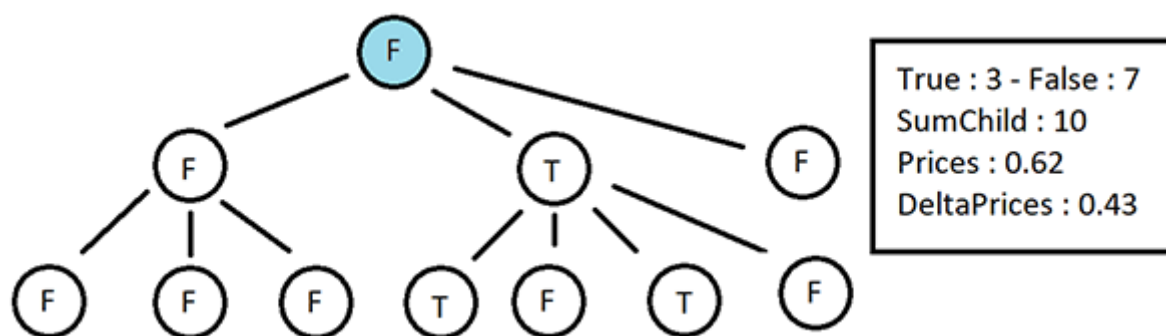
Mục tiêu của bài toán là làm cho độ lệch này nhỏ nhất có thể phụ thuộc vào chủ thể sẽ hỏi những ai .

Độ lệch này thể hiện khả năng đoán đúng và độ uy tín của cộng đồng . Khi chủ thể hỏi những người uy tín , ở trong những cộng đồng chứng khoán uy tín , tỉ lệ thành công sẽ cao hơn rất nhiều so với các trường hợp còn lại .

1.2 Khởi tạo quần thể ban đầu

Quần thể khởi tạo bao gồm 40 người , chia làm 4 cộng đồng nhỏ , mỗi cộng đồng có 10 người , tương ứng với 4 trường hợp chủ thể bài toán ở trong cộng đồng nào , đi hỏi những ai .

Mỗi cộng đồng được biểu diễn theo mô hình cây (Tree) , thể hiện thông tin sau việc đi hỏi và đưa ra quyết định như hình dưới đây .



Quyết định được đưa ra dựa vào số người trả lời là mua (true) lớn hơn hay bán (false) lớn hơn .

1.3 Chọn lọc tự nhiên

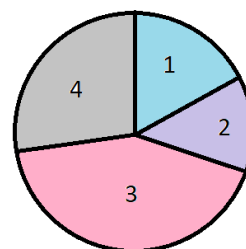
Những cộng đồng sau khi khởi tạo , trả về những thông tin như sau :

| Tree | TrueMinusFalse | SumChild | Prices |
|---------------------------------|--------------------|--------------------------|--------------------|
| 0 | -4.0 | 10.0 | 0.6200510377447751 |
| 1 | 6.0 | 10.0 | 1.537049566998035 |
| 2 | 2.0 | 10.0 | 1.197375320224904 |
| 3 | -2.0 | 10.0 | 0.802624679775096 |
| DeltaPrices | 1/DeltaPrices | Percent | |
| 0.4299073372131049 | 2.326082654189036 | 15.29745112792527 (%) | |
| 0.48709119204015505 | 2.053003659974951 | 13.501550814353067 (%) | |
| 0.14741694526702398 | 6.7834806791624125 | 44.611468977610606 (%) | |
| 0.24733369518278403 | 4.043120769537616 | 26.589529080111056 (%) | |
| GlobalPrices : 1.04995837495788 | | | |
| Sum : 15.205687762864015 | | | |

Việc chọn lọc tự nhiên sẽ được thực hiện dựa trên kết quả chênh lệch giá .

Giá chênh lệch càng ít thì cá thể càng tinh túy và cần được giữ lại để lai ghép .

Cá thể tinh túy sẽ chiếm nhiều tỉ lệ % hơn trong vòng quay chọn lọc và đồng nghĩa với xác xuất được giữ lại là cao hơn .



Ta mong muốn độ chênh lệch giá (DeltaPrices) càng nhỏ , và bằng 0 là tối ưu , nên (1 / DeltaPrices) càng lớn sẽ càng tốt . Cá thể nào có (1 / DeltaPrices) càng lớn thì càng chiếm nhiều tỉ lệ % trong vòng quay .

Sau khi đã hoàn thành Chọn lọc tự nhiên , ta "tái tạo" lại quần thể ban đầu gồm 4 cá thể tinh túy nhất .

1.4 Lai ghép

Tiến hành lai ghép các cặp cá thể (1 và 2) , (3 và 4) để tạo thành 4 cá thể mới mang gen của cả bố và mẹ

Đồng thời hi vọng rằng quần thể con sẽ xuất sắc hơn quần thể trước khi lai tạo .

Mô hình được triển khai bằng cách ngắt cây con trái của cây này gắn vào bên phải của cây kia .

1.5 Nghiệm thu

Cuối cùng là tính lại thông số và tiếp tục lặp lại việc Chọn lọc - Lai ghép cho đến khi chênh lệch không quá 5% so với Giá thực tế hoặc các cộng đồng giống hết nhau thì dừng lại .

Kết quả thu được là một cộng đồng uy tín , cho ra kết quả Giá dự đoán gần sát nhất so với Giá thực tế .

Nếu chủ thể hỏi đúng những hàng xóm này , họ sẽ có chiến lược và quyết định đúng đắn để đạt lợi nhuận tối ưu nhất .

2 Kiểm thử chức năng

Hàm main nhận đầu vào là giá trị của 4 cây cộng đồng trong quần thể khởi tạo .

Dầu ra là độ chênh lệch Giá dự đoán so với Giá thực tế .

Kết quả đầu ra với mỗi lần chạy là khác nhau do việc lai tạo có yếu tố ngẫu nhiên , bài toán được thực hiện trên cơ sở cố gắng tối ưu nhất có thể và hi vọng rằng sẽ đạt được kết quả thực nghiệm tốt .

Nếu đầu ra đạt được nhỏ hơn 0.25 (25%) đồng nghĩa với ca kiểm thử thành công .

Nếu ca kiểm thử thất bại , ta cần kiểm thử lại một vài lần nữa .

Thực nghiệm cho thấy xác suất đạt được kết quả tốt rơi vào 60 (%) .

Nên không cần kiểm thử quá 3 lần để ca kiểm thử có thể thành công .

```
@Test
public void testMain() throws Exception {
    boolean[] tree0 = { false, false, false, false, true, true, false, true, false, false };
    boolean[] tree1 = { true, true, false, true, true, true, true, true, true, false };
    boolean[] tree2 = { true, false, true, false, false, false, false, true, true, true };
    boolean[] tree3 = { true, true, true, true, false, false, false, false, false, false };

    assertTrue(" Chưa tối ưu . Hãy thử lại 1 lần nữa . ", Main.main(tree0, tree1, tree2, tree3) < 0.05);
}
```

2.1 Kiểm thử giá trị biên

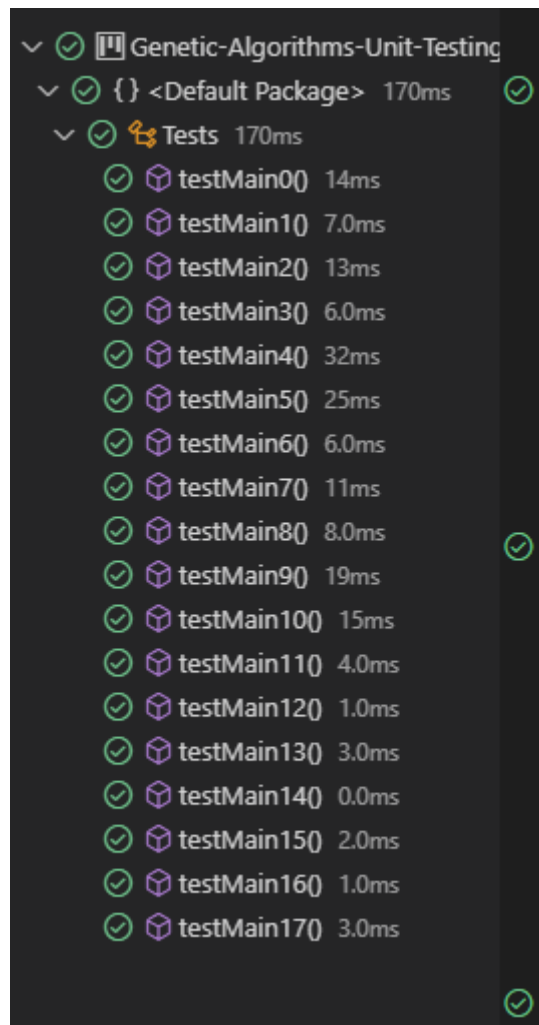
Các giá trị đầu vào : 4 bảng kiểu Boolean , mỗi mảng chứa

[(x) giá trị True và (10 - x) giá trị False ($0 \leq x \leq 10$)]

Khi đó ,

$\min(x) = 0$, $\max(x) = 10$, $\min(x)+ = 1$, $\max(x)- = 9$, $\text{nom}(x) = 5$

Số lượng ca kiểm thử cần thực hiện : ($4 \times 4 + 1 = 17$ ca)



Kiểm thử giá trị biên mạnh không được sử dụng trong báo cáo lần này vì min- và max+ không tồn tại như các nút trên cây và không có giá trị True / False để gán .

*> Kết quả triển khai :

Xác suất 80% tất cả các ca kiểm thử đều hoàn thành và cho kết quả tối ưu .

```

// nom - nom - nom - nom

@Test
public void testMain3() throws Exception {
    boolean[] tree0 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree1 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree2 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree3 = { true, true, true, true, true, false, false, false, false, false };

    assertTrue(" Chưa tối ưu . Hãy thử lại 1 lần nữa . ", Main.main(tree0, tree1, tree2, tree3) < 0.25);
}

// nom - nom - nom - max-

@Test
public void testMain4() throws Exception {
    boolean[] tree0 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree1 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree2 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree3 = { true, true, true, true, true, true, true, true, true, false };

    assertTrue(" Chưa tối ưu . Hãy thử lại 1 lần nữa . ", Main.main(tree0, tree1, tree2, tree3) < 0.25);
}

// nom - nom - nom - max

@Test
public void testMain5() throws Exception {
    boolean[] tree0 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree1 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree2 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree3 = { true, true, true, true, true, true, true, true, true, true };

    assertTrue(" Chưa tối ưu . Hãy thử lại 1 lần nữa . ", Main.main(tree0, tree1, tree2, tree3) < 0.25);
}

// nom - nom - min - nom

@Test
public void testMain6() throws Exception {
    boolean[] tree0 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree1 = { true, true, true, true, true, false, false, false, false, false };
    boolean[] tree2 = { false, false, false, false, false, false, false, false, false, false };
    boolean[] tree3 = { true, true, true, true, true, false, false, false, false, false };
}

```

2.2 Kiểm thử phân hoạch tương đương

Như đã trình bày ở mục bên trên , bài toán có miền xác định $0 \leq x \leq 10$.

Ta phân chia x làm 3 miền giá trị tương ứng với trạng thái của cây :

- x thuộc $[0,4]$ \Leftrightarrow TrueMinusFalse $< 0 \Leftrightarrow$ Giá cổ phiếu giảm , $P(t) < 1$

- x thuộc 5 \Leftrightarrow TrueMinusFalse $= 0 \Leftrightarrow$ Giá cổ phiếu không đổi , $P(t) = 1$

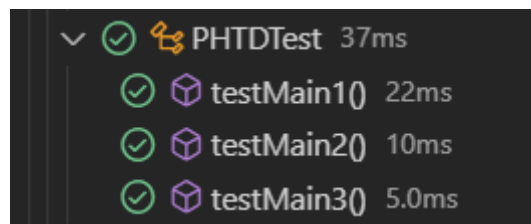
- x thuộc $[6,10]$ \Leftrightarrow TrueMinusFalse $> 0 \Leftrightarrow$ Giá cổ phiếu tăng , $P(t) > 1$

Nếu kiểm thử phân hoạch tương đương mạnh , ta cần đến $4*4*4 = 64$ (ca Kiểm thử)

Vì thế để đơn giản hóa bài toán , ta chỉ kiểm thử phân hoạch tương đương yếu .

Số ca kiểm thử : 3 (ca)

Lựa chọn ca kiểm thử : (3,5,9,10) , (8,2,1,5) , (5,7,5,4)



3 Tổng kết

Báo cáo ứng dụng kiểm thử hộp đen (Kiểm thử chức năng) để sinh ca kiểm thử cho chương trình bài toán Sử dụng Giải thuật Di truyền trong việc Tối ưu hóa dự đoán giá cổ phiếu.

Sử dụng 2 kĩ thuật :

Kiểm thử Giá trị biên và Kiểm thử phân hoạch tương đương .

Rất mong báo cáo của em lần này có thể gây ấn tượng với cô và các bạn .

4 Tài liệu

- [1] Bài giảng lớp Kiểm thử và Đảm bảo chất lượng phần mềm - Cô Nguyễn Thị Thu Trang <https://courses.uet.vnu.edu.vn/course/view.php?id=8435>.