形式化方法 lab 6 实验小作业

- 棋盘更改后的复现

PB22111599 杨映川

1 更改后的棋盘

```
Definition start : board
  := [ [Black | White | Black] |
       [White | White | White] |
       [White | Black | Black] ].

Definition target : board
  := [ [Black | Black | Black] |
       [White | Black | White] |
       [Black | Black | White] ].
```

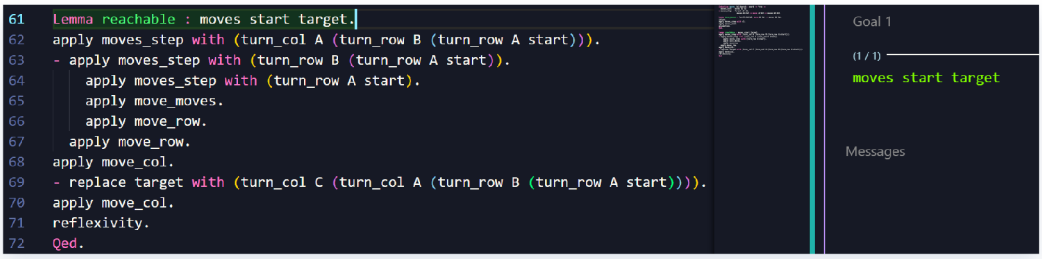
通过观察可以发现，一种解法为：start -> turn_row A -> turn_row B -> turn_col A -> turn_col C -> target

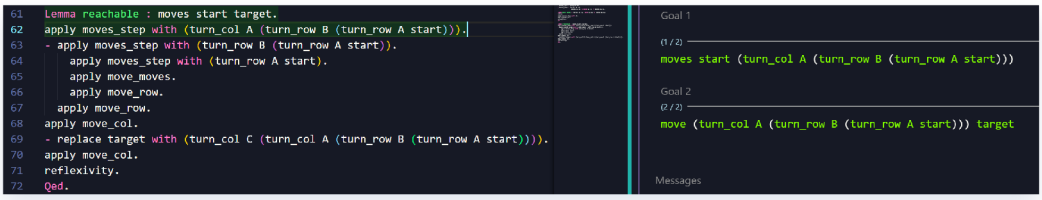
2 代码实现

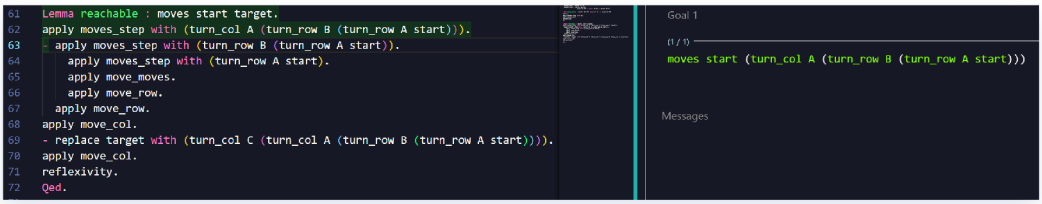
```
Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
    apply moves_step with (turn_row A start).
    apply move_moves.
    apply move_row.
  apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.

```

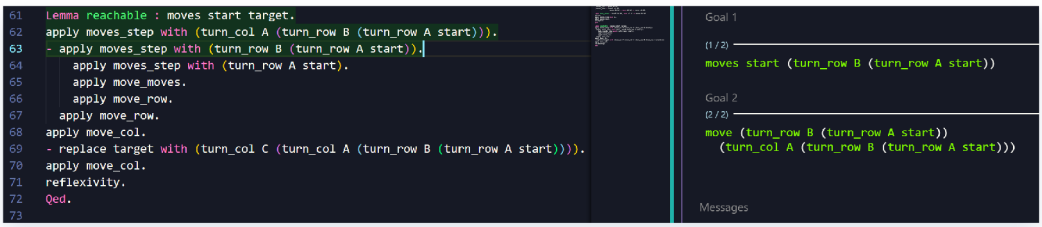
仅修改了 `Lemma reachable` 的部分，借助预先发现的一条路径反推出证明

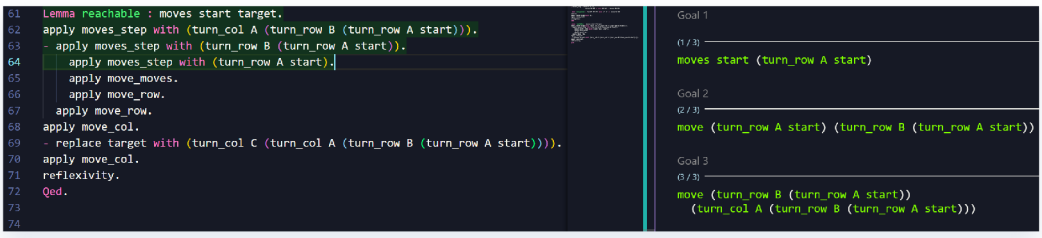
3 运行过程

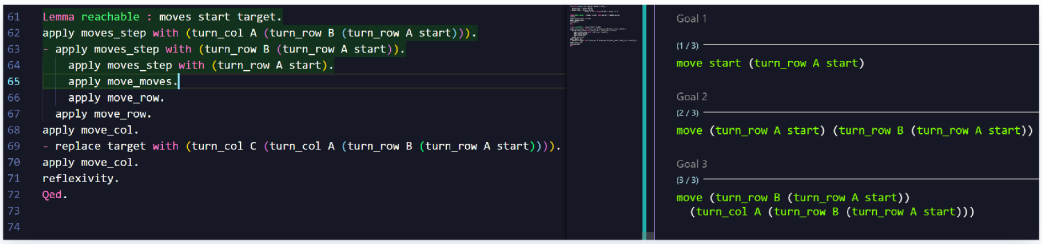

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 1)
moves start target
Messages


Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 2)
moves start (turn_col A (turn_row B (turn_row A start)))
Goal 2
(2 / 2)
move (turn_col A (turn_row B (turn_row A start))) target
Messages

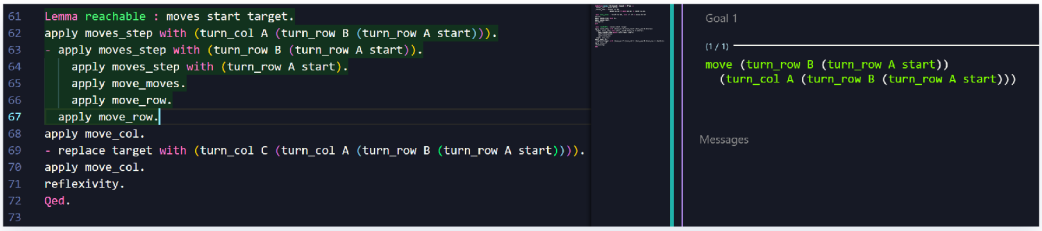

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 1)
moves start (turn_col A (turn_row B (turn_row A start)))
Messages

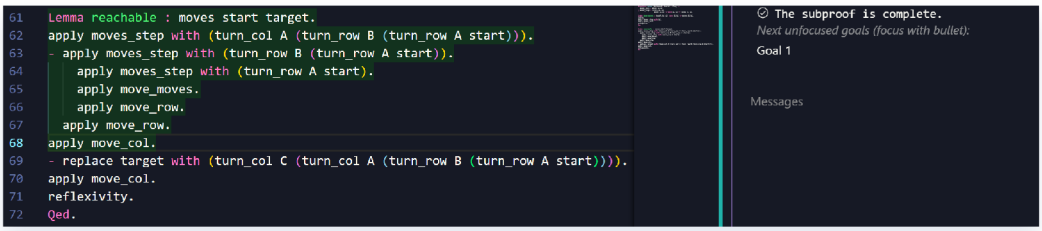

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 2)
moves start (turn_row B (turn_row A start))
Goal 2
(2 / 2)
move (turn_row B (turn_row A start))
(turn_col A (turn_row B (turn_row A start)))
Messages

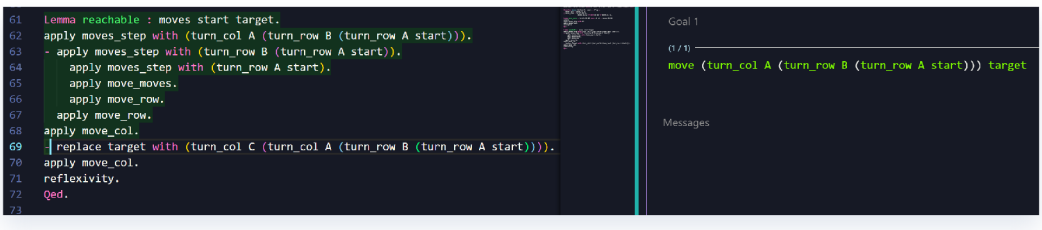

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 3)
moves start (turn_row A start)
Goal 2
(2 / 3)
move (turn_row A start) (turn_row B (turn_row A start))
Goal 3
(3 / 3)
move (turn_row B (turn_row A start))
(turn_col A (turn_row B (turn_row A start)))

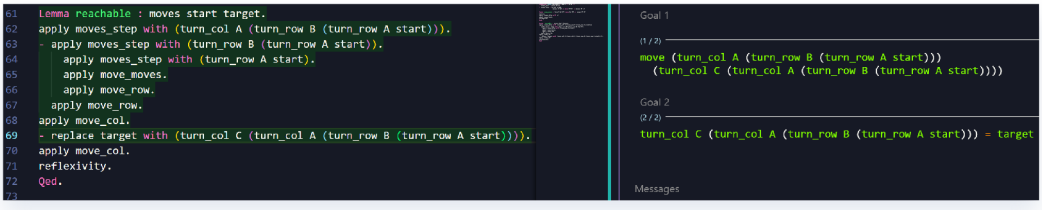

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 3)
move start (turn_row A start)
Goal 2
(2 / 3)
move (turn_row A start) (turn_row B (turn_row A start))
Goal 3
(3 / 3)
move (turn_row B (turn_row A start))
(turn_col A (turn_row B (turn_row A start)))

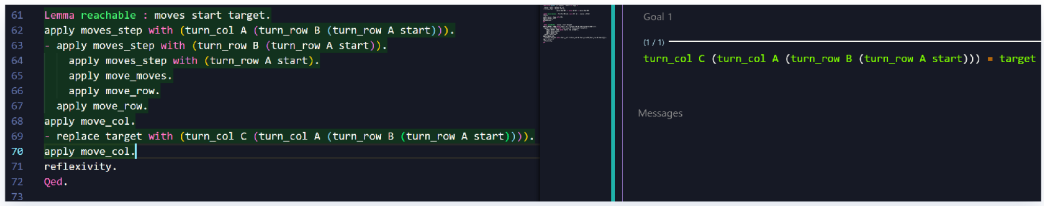

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 2)
move (turn_row A start) (turn_row B (turn_row A start))
Goal 2
(2 / 2)
move (turn_row B (turn_row A start))
(turn_col A (turn_row B (turn_row A start)))
Messages

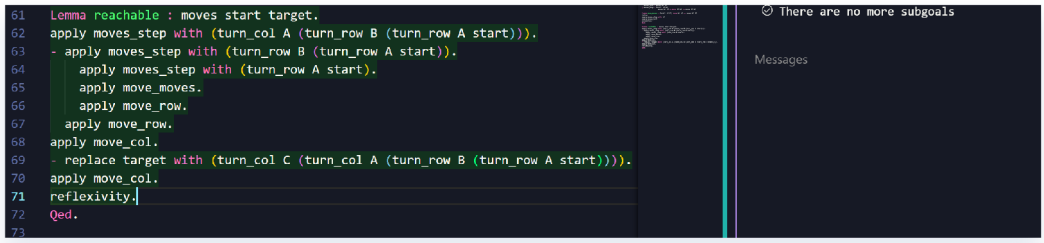

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 1)
move (turn_row B (turn_row A start))
(turn_col A (turn_row B (turn_row A start)))
Messages

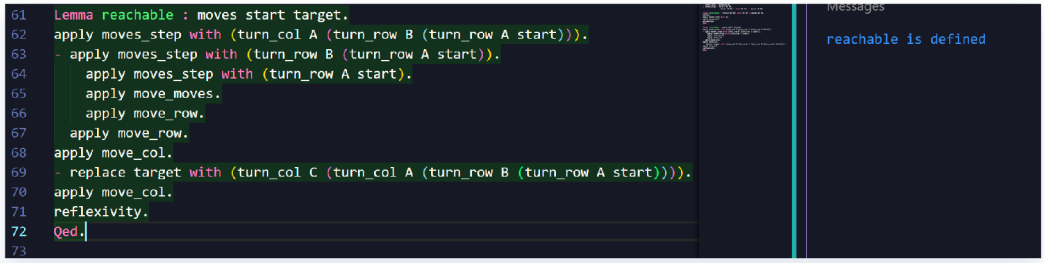

Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
The subproof is complete.
Next unfocused goals (focus with bullet):
Goal 1
Messages


Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 1)
move (turn_col A (turn_row B (turn_row A start))) target
Messages


Lemma reachable : moves start target.
apply moves_step with (turn_col A (turn_row B (turn_row A start))).
- apply moves_step with (turn_row B (turn_row A start)).
apply moves_step with (turn_row A start).
apply move_moves.
apply move_row.
apply move_row.
apply move_col.
- replace target with (turn_col C (turn_col A (turn_row B (turn_row A start)))).
apply move_col.
reflexivity.
Qed.
Goal 1
(1 / 2)
move (turn_col A (turn_row B (turn_row A start)))
(turn_col C (turn_col A (turn_row B (turn_row A start))))
Goal 2
(2 / 2)
turn_col C (turn_col A (turn_row B (turn_row A start))) = target
Messages







4 完整代码

```
Inductive color : Type := White | Black.
Inductive pos : Type := A | B | C.
Inductive triple M := Triple : M → M → M → triple M.
Set Implicit Arguments.

Notation "[ x | y | z ]" := (Triple _ x y z).

Definition triple_x M (m:M) : triple M := [m | m | m].

Definition turn_color (c: color) : color :=
  match c with | White ⇒ Black | Black ⇒ White end.

Definition triple_map M f (t: triple M) : triple M:=
match t with (Triple _ a b c) ⇒ [(f a)|(f b)|(f c)] end.

Definition triple_map_select M f p t : triple M :=
  match t with (Triple _ a b c) ⇒
    match p with | A ⇒ [ (f a) | b | c ]
                 | B ⇒ [ a | (f b) | c ]
                 | C ⇒ [ a | b | (f c) ]
    end
  end.
```

```
23
24  Definition board := triple (triple color).
25
26  Definition start : board
27    := [ [Black | White | Black] |
28         [White | White | White] |
29         [White | Black | Black] ].
30
31  Definition target : board
32    := [ [Black | Black | Black] |
33         [White | Black | White] |
34         [Black | Black | White] ].
35
36  Definition turn_row (p: pos) : board → board :=
37      triple_map_select (triple_map turn_color) p.
38
39  Definition turn_col (p: pos) : board → board :=
40      triple_map (triple_map_select turn_color p).
41
42  Definition move1 (b1 b2: board) : Prop :=
43          (exists p : pos, b2=turn_row p b1)
44       \/ (exists p : pos, b2=turn_col p b1).
45
46  Inductive move (b1:board) : board → Prop :=
47  move_row : forall (p:pos), move b1 (turn_row p b1) | move_col :
    forall (p:pos), move b1 (turn_col p b1).
48
49  Inductive moves (b1:board): board → Prop :=
50    moves_init : moves b1 b1
51  | moves_step : forall b2 b3,
52                   moves b1 b2 → move b2 b3 → moves b1 b3.
53
54  Lemma move_moves : forall b1 b2, move b1 b2 → moves b1 b2.
55  intros.
56  apply moves_step with b1.
57  apply moves_init.
58  assumption.
59  Qed.
60
61  Lemma reachable : moves start target.
62  apply moves_step with (turn_col A (turn_row B (turn_row A
    start))).
63  - apply moves_step with (turn_row B (turn_row A start)).
64      apply moves_step with (turn_row A start).
65      apply move_moves.
66      apply move_row.
67    apply move_row.
68  apply move_col.
69  - replace target with (turn_col C (turn_col A (turn_row B
    (turn_row A start)))).
70  apply move_col.
71  reflexivity.
```

```
72  Qed.
73
74
```