Gambler's Ruin Problem

Zibo Yang

July 25, 2021

Contents

theory *Gambler-Ruin-Problem*
  imports *DiscretePricing.Infinite-Coin-Toss-Space*

begin

1 Gambler's ruin problem

In Gamblerruin.thy, we will construct the formalization of a specific random walk model coordinated with gambler's ruin problem.

1.1 Theory Infinite_Coin_Toss_Space

In order to construct the formal method in gambler's ruin problem, we start with the existing formalization in the Theory Infinite_Coin_Toss_Space which constructed the probability space on infinite sequences of independent coin. tosses.

The only concept need to be elaborated is bernoulli. 'a stream is a type of infinite sequence with all element of type 'a. The bernoulli stream is a stream measure which has the space composed of all the elements of type bool stream, measurable sets filled with all the subset of its space. under this specific measure, all the possibility of occurrence of elements in a specific set A can be described as the measure value of A if and only if A is in the measurable sets of this bernoulli stream. In fact, it was set up by producting countable measure of boolean space with measuring {True} to p and {False} to $1-p$.

1.2 Gambler model

fun (**in** *infinite-coin-toss-space*) *gambler-rand-walk-pre*:: *int* ⇒ *int* ⇒ *int* ⇒ (*nat* ⇒ *bool stream* ⇒ *int*) **where**
base: *gambler-rand-walk-pre u d v 0 w* = *v*|
step1: *gambler-rand-walk-pre u d v* (*Suc n*) *w* = ((λ*True* ⇒ *u* | *False* ⇒ *d*) (*snth w n*)) + *gambler-rand-walk-pre u d v n w*

fun (**in** *infinite-coin-toss-space*) *gambler-rand-walk*:: *int* ⇒ *int* ⇒ *int* ⇒ (*enat* ⇒ *bool stream* ⇒ *int*) **where**
gambler-rand-walk u d v n w = (*case n of enat n* ⇒ (*gambler-rand-walk-pre u d v n w*)|∞ ⇒ −*1*)

The function *gambler_ran_walk* extends the fourth parameter by adding ∞ as new input. The reason why we define it is that we found it very tough to describe the position where the specific random walk stops, for the first time, by reaching the threshold if natural number is the only allowed input as what *gambler_ran_walk_pre* defines. Since some infinite random walks will never stop, we must allocate ∞ as the output coordinated with that non-stop case and extend the type of steps from *nat* to *enat*. But if someone wants to base their further analysis on our endeavor here, please be cautious of or even avoid discussing the case that initial number and target number is negative since we map ∞ to -1. The lemma *exist* demonstrates that non-stop random walk will never succeed in reaching the target, which is the best explanation why we allocates -1 as the output of ∞ in *gambler_ran_walk*.

locale *gambler-model* = *infinite-coin-toss-space* +
fixes *geom-proc*::*int* ⇒ *bool stream* ⇒ *enat* ⇒ *int*
assumes *geometric-process*:*geom-proc init x step* = *gambler-rand-walk 1* (−*1*) *init step x*

begin

1.3 Basic functions

Here we define the all basic functions which will play an invisible role in the further probability analysis. you can just focus on the lemmas and functions where we comment

definition *reach-steps*::*int* $\Rightarrow$ *bool stream* $\Rightarrow$ *int* $\Rightarrow$ *nat set***where**
reach-steps init x target = {*step*::*nat*. *geom-proc init x step* $\in$ {*0*,*target*}}

reach_steps describes all the steps where the input random walk reaches the threshold 0, target

fun *infm*::*nat set* $\Rightarrow$ *enat* **where**
infm A = (*if A* = {} *then* ∞ *else* $\bigsqcap$ *A*)

infm A = iff A =

lemma *only-inf-infm*:
  assumes*A* $\neq$ {}
  shows*infm A* $\neq$ ∞
⟨*proof*⟩

fun *stop-at*::*int* $\Rightarrow$ *bool stream* $\Rightarrow$ *int* $\Rightarrow$ *enat* **where**
stop-at init x target = (*infm* (*reach-steps init x target*))

stop_at describes the first step in the *reach_steps* sets, which means exactly the stopping point in gambler's ruin problem. Be careful, Here the type of output has been extended to *enat*, which means stopping point will be ∞(equivalent to non-existence)

fun *success*::*int* $\Rightarrow$ *bool stream* $\Rightarrow$ *int* $\Rightarrow$ *bool***where**
success init x target = (*geom-proc init x* (*stop-at init x target*) = *target*)

success describes the random walk reaching the target number rather than ruining at stopping point

1.4 Important intermediate conclusions

1.4.1 Successful random walks never stop at ∞

Once we set target to be positive, the weird situation where random walk succeeds at ∞ will disappear

lemma *exist*:
  fixes *init*::*int* **and** *x* **and** *target*::*int*
  assumes *0* $\leq$ *init init* $\leq$ *targetsuccess init x target*
  shows *stop-at init x target* $\neq$ ∞
⟨*proof*⟩

1.4.2 The way we count never change the amount got through specific random walk

lemma *pre1*:$\bigwedge x\ n.\ snth\ x\ (n{+}1) = snth\ (stl\ x)\ n$
⟨*proof*⟩

lemma additional1 states that the reaching number doesn't change if we want to calculate from the second step

lemma *additional1*:*let* $init' = geom\text{-}proc\ init\ x\ 1$ *in*
$geom\text{-}proc\ init'\ (stl\ x)\ n = geom\text{-}proc\ init\ x\ (Suc\ n)$
⟨*proof*⟩

1.4.3 The way we count never change whether the random walk succeeds

lemma *set-up-Inf*:
fixes A **and** $a{::}nat$
assumes $\bigwedge b{::}nat.\ b \in A \Longrightarrow a \le b$ $a \in A$
shows $a = Inf\ A$
⟨*proof*⟩

lemma *Inf-property*:
fixes a **and** A
assumes $a = Inf\ A$
shows $\bigwedge b{::}nat.\ b \in A \Longrightarrow a \le b$
⟨*proof*⟩

conditional2_pre states that stopping point doesn't change if we calculate from second step

lemma *conditional2-pre*:
fixes $init'$ **and** Ar **and** Al
assumes $init' = geom\text{-}proc\ init\ x\ 1$
$Ar = reach\text{-}steps\ init\ x\ target$
$Al = reach\text{-}steps\ init'\ (stl\ x)\ target$
$0 < init$
$init < target$
shows $stop\text{-}at\ init'\ (stl\ x)\ target + 1 = stop\text{-}at\ init\ x\ target$

⟨*proof*⟩

conditional2 states that whether a random walk succeeds or not doesn't change if we calculate from second step

lemma *conditional2*:
fixes $init\ x\ target$
assumes $init' = geom\text{-}proc\ init\ x\ 1$
$0 < init$
$init < target$
shows $success\ init'\ (stl\ x)\ target \longleftrightarrow success\ init\ x\ target$
⟨*proof*⟩

1.4.4 The change of initial number

if first step is true, then we add 1 to initial number

lemma *fst-true-plus-one*:
  fixes *init x target*
  assumes $init' = geom\text{-}proc\ init\ x\ 1shd\ x = True$
  shows $init' = init + 1$
⟨*proof*⟩

if first step is False, then we reduce 1 to initial number

lemma *fst-true-plus-one-false*:
  fixes *init x target*
  assumes $init' = geom\text{-}proc\ init\ x\ 1shd\ x = False$
  shows $init' = init - 1$
⟨*proof*⟩

1.4.5 The way we count never change the successful random walk set

the set where all random walks in it succeeds and their first step are True doesn't change if we calculate from second step

lemma *conditional-set-equation*:
  fixes *init target*
  assumes
    $0 < init$
    $init < target$
  shows
$\{x{::}bool\ stream.\ success\ init\ x\ target \wedge shd\ x = True\} =$
$\{x{::}bool\ stream.\ success\ (init{+}1)\ (stl\ x)\ target \wedge shd\ x = True\}$
⟨*proof*⟩

the set where all random walks in it succeeds and their first step are False doesn't change if we calculate from second step

lemma *conditional-set-equation-false*:
  fixes *init target*
  assumes
    $0 < init$
    $init < target$
  shows
$\{x{::}bool\ stream.\ success\ init\ x\ target \wedge shd\ x = False\} =$
$\{x{::}bool\ stream.\ success\ (init{-}1)\ (stl\ x)\ target \wedge shd\ x = False\}$
⟨*proof*⟩

1.5 Probability equation

Here we start to analyse the probability of successful random walk. To better understand this part please have a look the elaboration in front of lemma *success_measurable* *probability_of_win* is the function describing possibility of successful random walks with initial number and target number as inputs

fun *probability-of-win*::*int* $\Rightarrow$ *int* $\Rightarrow$ *ennreal***where**
probability-of-win init target = *emeasure M* $\{x \in$ *space M*. *success init x target*$\}$

1.5.1 Successful random walk set is measurable

Preimage of function snth is measurable

lemma *snth-measurable*:
fixes *n*::*nat*
shows$\bigwedge k.$ $(\lambda w.$ *snth w n*$)$ $-$‘ $\{k\} \in$ *sets M*
⟨*proof*⟩

lemma *stake-measurable-pre1*:
fixes *n w k*
assumes *length k* $>$ *n*
shows *stake* (*Suc n*) *w* = *take* (*Suc n*) *k* $\longleftrightarrow$ *stake n w* = *take n k* $\wedge$ *snth w n* = *nth k n*
⟨*proof*⟩

lemma *stake-measurable-pre*:
fixes *n*
shows$\bigwedge k.$ *length k* $\geq$ *n* $\Longrightarrow$ (*stake n* $-$‘ $\{k\}$) $\in$ *sets M*
⟨*proof*⟩

The preimage of any list over function stake is measurable

lemma *stake-measurable*:
fixes *n k*
shows(*stake n* $-$‘ $\{k\}$) $\in$ *sets M*
⟨*proof*⟩

The preimage of any list set over function stake is measurable once the set is finite

lemma *finite-stake-measurable*:
fixes *A* **and** *n*::*nat*
assumes*finite A*
shows(*stake n* $-$‘ *A*) $\in$ *sets M*
⟨*proof*⟩

The new *geom_proc* function for list

fun *geom-proc-list*::*int* $\Rightarrow$ *bool list* $\Rightarrow$ *int***where**
geom-proc-list init [] = *init*|
geom-proc-list init (*x* # *xs*) = (*case x of True*$\Rightarrow$*1*|*False* $\Rightarrow$ $-$*1*) + *geom-proc-list init xs*

lemma *reverse-construct-pre*:
fixes *init lengthx y*
shows $\bigwedge x{::}bool\ list.\ lengthx = length\ x \Longrightarrow\ geom\text{-}proc\text{-}list\ init\ (x\ @\ [y]) = geom\text{-}proc\text{-}list\ init\ x + (case\ y\ of\ True{\Rightarrow}1|False{\Rightarrow}\ -1)$
⟨*proof*⟩

lemma *reverse-construct*:
fixes *init x y*
shows $geom\text{-}proc\text{-}list\ init\ (x\ @\ [y]) = geom\text{-}proc\text{-}list\ init\ x + (case\ y\ of\ True{\Rightarrow}1|False{\Rightarrow}\ -1)$
⟨*proof*⟩

lemma *success-pre*:
fixes *init x target i*
assumes $0 < initinit < target$
shows $geom\text{-}proc\text{-}list\ init\ (stake\ i\ x) = geom\text{-}proc\ init\ x\ i$
⟨*proof*⟩

Any natural number smaller than Inf A doesn't belong to A

lemma *not-belong*:
fixes *A* **and** *a::nat*
assumes $\bigsqcap\ A > a$
shows $a \notin A$
⟨*proof*⟩

This is the most important intermediate lemma prepared for lemma $success\_measurable.Itclarifiest$

lemma *success-measurable2*:
fixes *init target* **and** *i::nat*
assumes $0 < initinit < target\ 0 \leq i$
shows$\{x{\in}\ space\ M.\ success\ init\ x\ target \wedge stop\text{-}at\ init\ x\ target = i\} = stake\ i\ -`\ \{c{::}bool\ list.\ (\forall\, k{<}i.\ (geom\text{-}proc\text{-}list\ init\ (take\ k\ c)) \notin \{0,target\}){\wedge}\ length\ c = i \wedge geom\text{-}proc\text{-}list\ init\ c = target\}$
⟨*proof*⟩

lemma *stake-space*:$stake\ n\ `\ space\ M = \{c{::}bool\ list.\ length\ c = n\}$
⟨*proof*⟩

Set of all the lists with specific length is finite

lemma *finite-length*:$finite\ \{c{::}bool\ list.\ length\ c = n\}$
⟨*proof*⟩

lemma *finite-image*:$finite\ \{c{::}bool\ list.\ (\forall\, k{<}i.\ (geom\text{-}proc\text{-}list\ init\ (take\ k\ c)) \notin \{0,target\}){\wedge}\ length\ c = i \wedge geom\text{-}proc\text{-}list\ init\ c = target\}$
⟨*proof*⟩

Sets of all successful random walk with specific stop is measurable

lemma *success-measurable3*:

fixes *init* **and** *target* **and** *i::nat*
assumes $0<init$ $init<target$ $0 \leq i$
shows$\{x\in$ *space* M. *success init x target* $\wedge$ *stop-at init x target* $=$ *enat i*$\}$ $\in$ *sets M*
⟨*proof*⟩

Any successful random walk must stop at specific position described by natural number

lemma *success-measurable1*:
fixes *init target*
assumes $0 <$ *init* *init* $<$ *target*
shows $\{x\in$ *space* M. *success init x target*$\}$
$= (\bigcup i::nat.\ \{x\in$ *space* M. *success init x target* $\wedge$ *stop-at init x target* $= i\})$
⟨*proof*⟩

Here we need to elaborate about this most difficult lemma we've met during this model formalization. lemma $success\_measurable$ asserts that successful random walks set under assumption "$0 \leq initialnumber \leq targetnumber$" is measurable set for measure M. On the one hand, since the probability theory has been set up based on the measure theory, every specific set must be proved to be measurable with respect to fixed measure before we calculate the probability of the set, which severely hinders most of scholars and experts from formalizing the security analysis related to the probability since it's extremely difficult to prove why your set is measurable. That is exactly why our endeavor matters to provide the first example to overcome the difficulty. On the other hand, we are willing to briefly explain the way we prove this lemma since it's nontrivial even for pen-and-paper proof. lemma $finite\_stake\_measurable$ states that for the function (λw. stake n w) taking the first n steps of random walk, the preimage of a finite sets is measurable for measure M. lemma $finite\_image$ states that sets filled with all bool list of fixed length n is finite. lemma $success\_measurable2$ sets up the bijection between successful random walks stopping at fixed step and preimage of successful bool list with identical length. lemma $success\_measurable1$ demonstrates that set of successful random walks is countable union of sets of successful random walks stopping at some step. Combining theses 4 lemmas together proves the set of successful random walk is measurable. If you take a closed look at the proofs of these 4 lemmas patiently, you will find it's very hard to finish. Honestly, we will never be able to finish such difficult proofs within one month without the current stochastic process theory library established just in 2021 by Mnacho Echenim, the author of theory $infinite\_coin\_toss\_space$.

lemma *success-measurable*:
fixes *init target*
assumes $0 \leq$ *init* *init* $\leq$ *target*
shows $\{x\in$ *space* M. *success init x target*$\}$ $\in$ *sets M*
⟨*proof*⟩

The set of all the random walk with first step True is measurable

lemma *success-measurable-shd*:
$\{x \in space\ M.\ shd\ x\} \in sets\ M$
⟨*proof*⟩

The set of all the random walk with first step False is measurable

lemma *success-measurable-shd-false*:
$\{x \in space\ M.\ \neg\ shd\ x\} \in sets\ M$
⟨*proof*⟩

lemma *success-measurable-final*:
fixes *init target*
assumes $0 < init\ init < target$
shows$\{x \in space\ M.\ success\ (init+1)\ (stl\ x)\ target \wedge shd\ x\} \in sets\ M$
⟨*proof*⟩

1.5.2 Probability of successful random walk with its first step True

lemma *semi-goal1*:
fixes *init target P*
assumes $0 < initinit \leq target \bigwedge x.\ P\ x = success\ (init+1)\ (stl\ x)\ target \wedge shd\ x$
shows $emeasure\ M\ \{x.\ P\ (t\ \#\#\ x)\}$
$= (case\ t\ of\ True \Rightarrow 1|False \Rightarrow 0) * emeasure\ M\ \{x.\ success\ (init+1)\ x\ target\}$
⟨*proof*⟩

lemma *semi-goal21*:
fixes *p1 e::real* **and** *f*
assumes $m = measure\text{-}pmf\ (bernoulli\text{-}pmf\ p1) \bigwedge t.\ f\ t = ennreal\ e * (case\ t\ of\ True \Rightarrow 1|\ False \Rightarrow 0)$
shows *simple-function m f*
⟨*proof*⟩

lemma *semi-goal21-false*:
fixes *p1 e::real* **and** *f*
assumes $m = measure\text{-}pmf\ (bernoulli\text{-}pmf\ p1) \bigwedge t.\ f\ t = ennreal\ e * (case\ t\ of\ True \Rightarrow 0|\ False \Rightarrow 1)$
shows *simple-function m f*
⟨*proof*⟩

lemma *sum-rephrase*:
fixes $f::ennreal \Rightarrow ennreal$ **and** e
assumes $0 \neq e$
shows $sum\ f\ \{0,e\} = f\ 0 + f\ (e)$

⟨*proof*⟩

lemma *semi-goal22*:
fixes *p1 e*::*real*
assumes $m = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p1) 0 \le p1 p1 \le 1$
$\bigwedge t.\ f\ t = \textit{ennreal}\ e * (\textit{case}\ t\ \textit{of}\ \textit{True} \Rightarrow 1 |\ \textit{False} \Rightarrow 0)$
$e > 0$
shows $\textit{integral}^S\ m\ f = p1 * e$
⟨*proof*⟩

lemma *semi-goal22-false*:
fixes *p1 e*::*real*
assumes $m = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p1) 0 \le p1 p1 \le 1$
$\bigwedge t.\ f\ t = \textit{ennreal}\ e * (\textit{case}\ t\ \textit{of}\ \textit{True} \Rightarrow 0 |\ \textit{False} \Rightarrow 1)$
$e > 0$
shows $\textit{integral}^S\ m\ f = (1 - p1) * e$
⟨*proof*⟩

lemma *semi-goal23*:
fixes *p1 e*::*real* **and** *f*
assumes $0 \le p1$
$p1 \le 1$
$e > 0$
$m = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p1)$
$\bigwedge t.\ f\ t = \textit{ennreal}\ e * (\textit{case}\ t\ \textit{of}\ \textit{True} \Rightarrow 1 |\ \textit{False} \Rightarrow 0)$
shows $\int^+ t.\ (f\ t)\ \partial m = \textit{integral}^S\ m\ f$
⟨*proof*⟩

lemma *semi-goal23-false*:
fixes *p1 e*::*real* **and** *f*
assumes $0 \le p1$
$p1 \le 1$
$e > 0$
$m = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p1)$
$\bigwedge t.\ f\ t = \textit{ennreal}\ e * (\textit{case}\ t\ \textit{of}\ \textit{True} \Rightarrow 0 |\ \textit{False} \Rightarrow 1)$
shows $\int^+ t.\ (f\ t)\ \partial m = \textit{integral}^S\ m\ f$
⟨*proof*⟩

lemma *semi-goal2*:
fixes *p1 e*::*real* **and** *f*
assumes $0 \le p1$
$p1 \le 1$
$e \ge 0$
$m = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p1)$
$\bigwedge t.\ f\ t = \textit{ennreal}\ e * (\textit{case}\ t\ \textit{of}\ \textit{True} \Rightarrow 1 |\ \textit{False} \Rightarrow 0)$
shows $\int^+ t.\ (f\ t)\ \partial m = p1 * e$

$\langle proof \rangle$

lemma *semi-goal2-false*:
  fixes *p1 e*::*real* **and** *f*
  assumes $0 \leq p1$
  $p1 \leq 1$
  $e \geq 0$
$m = measure\text{-}pmf\ (bernoulli\text{-}pmf\ p1)$
$\bigwedge t.\ f\ t = ennreal\ e * (case\ t\ of\ True \Rightarrow 0|\ False \Rightarrow 1)$
shows $\int^{+} t.\ (f\ t)\ \partial m = (1-p1) * e$
$\langle proof \rangle$

lemma *semi-goal2-final*:
  fixes *p1*::*real* **and** *e*::*ennreal* **and** *f*
  assumes $0 \leq p1$
  $p1 \leq 1$
  $e \neq top$
$m = measure\text{-}pmf\ (bernoulli\text{-}pmf\ p1)$
$\bigwedge t.\ f\ t = e * (case\ t\ of\ True \Rightarrow 1|\ False \Rightarrow 0)$
shows $\int^{+} t.\ (f\ t)\ \partial m = p1 * e$
$\langle proof \rangle$

lemma *semi-goal2-final-false*:
  fixes *p1*::*real* **and** *e*::*ennreal* **and** *f*
  assumes $0 \leq p1$
  $p1 \leq 1$
  $e \neq top$
$m = measure\text{-}pmf\ (bernoulli\text{-}pmf\ p1)$
$\bigwedge t.\ f\ t = e * (case\ t\ of\ True \Rightarrow 0|\ False \Rightarrow 1)$
shows $\int^{+} t.\ (f\ t)\ \partial m =(1 - p1) * e$
$\langle proof \rangle$

lemma *fun-description-pre*:
  fixes *init target t*
  assumes $0 < initinit < target$
  shows
$emeasure\ M\ \{x \in space\ M.\ t\ \#\#\ x \in \{x \in space\ M.\ success\ (init+1)\ (stl\ x)\ target \wedge shd\ x\}\}$
$= (case\ t\ of\ True \Rightarrow 1|False \Rightarrow 0) * (emeasure\ M\ \{x \in space\ M.\ success\ (init+1)\ (x)\ target\})$
$\langle proof \rangle$

lemma *fun-description-pre-false*:
  fixes *init target t*
  assumes $0 < initinit < target$

shows
emeasure M $\{x \in$ *space M*. *t ## x* $\in \{x\in$ *space M*. *success* (*init*−*1*) (*stl x*) *target* $\wedge \neg$ *shd x*$\}\}$
= (*case t of True* $\Rightarrow$ *0*|*False* $\Rightarrow$ *1*) $*$ (*emeasure M* $\{x\in$ *space M*. *success* (*init*−*1*) (*x*) *target*$\}$)
⟨*proof*⟩

term*emeasure-stream-space*

The lemma *semi_goal_true* is the second difficulty we've overcome during the model formalization. It asserts that probability of sets of successful random walk with first step True is equal to probability of sets of random walk times probability of sets of successful random walk with initial number plus 1. Thanks to the lemma *emeasure_stream_space* provided by Mnacho Echenim, the author of $infinite\_coin\_toss\_space$, we could finally use the integral rather than tediously break down the countable product to calculate the probability

lemma *semi-goal-true*:
fixes *init target*
assumes *0* < *initinit* < *target*
shows *emeasure M* $\{x\in$ *space M*. *success* (*init*+*1*) (*stl x*) *target* $\wedge$ *shd x*$\}$
= *emeasure M* $\{x \in$ *space M*. *shd x*$\}$ $*$ *emeasure M* $\{x\in$ *space M*. *success* (*init*+*1*) (*x*) *target*$\}$
⟨*proof*⟩

lemma *semi-goal-false*:
fixes *init target*
assumes *0* < *initinit* < *target*
shows *emeasure M* $\{x\in$ *space M*. *success* (*init*−*1*) (*stl x*) *target* $\wedge \neg$ *shd x*$\}$
= *emeasure M* $\{x \in$ *space M*. $\neg$ *shd x*$\}$ $*$ *emeasure M* $\{x\in$ *space M*. *success* (*init*−*1*) (*x*) *target*$\}$
⟨*proof*⟩

1.5.3 Final goal: establish the recursive probability equation

The final probability equation we want to formalize:

$$P_n = pP_{n+1} + (1-p)P_{n-1}$$

lemma *Recursive-probability-equation*:
fixes *init target*
assumes *0* < *init init* < *target*
shows*probability-of-win init target* = *p* $*$ (*probability-of-win* (*init* + *1*) *target*) + (*1* − *p*) $*$ (*probability-of-win* (*init* − *1*) *target*)
⟨*proof*⟩

end
end